Method for Prototyping and Testing the Information Minimalism Concept of Nuclear Power Plants

Gwi-sook Jang*, Gee-yong Park
I&C and HFE Research Division, Korea Atomic Energy Research Institute, 989-111, Daedeok-daero, Yuseong-gu, Daejeon 305-353, Republic of Korea
* gsjang@kaeri.re.kr

***Abstract*—The information structure and visualization design in nuclear power plant is based on careful analysis and understanding of the work domain of the operator. Piping and instrumentation diagram (P&ID) based layouts tend to require more display space. Overuse of mimic layouts can result in visual clutter. P&ID based layouts of controls are usually less easily operated configurations than those provided by other array conventions. Thus, a display method is emerging to minimize P&ID based display. The information minimalism concept is a monitoring and controller display method for each operation mode based on log analyses of operator actions. This method provides the monitoring information and control means necessary for the operator, according to operation mode, to perform a specific operation. This method can reduce the time spent searching for information and the transition between display pages. The aim of this paper was to verify the feasibility of implementing new concepts by establishing an information minimalism prototype. The validity of the function, performance, and operational test methods were verified by testing one such information minimalism prototype.**

In this paper, we describe prototyping and testing methods for the information minimalism concept for NPPs. In the future, this concept is to be added to the concept of operator support with the existing display configuration and navigation, and thereby extend its application range while actually utilizing it.

***Keywords*—Monitoring System, Human Factors Engineering, Human-Machine Interface**

I. Introduction

The information display in a digital based control room of a Korean NPP (Nuclear Power Plant) consists of a hierarchically structured piping and instrumentation (P&ID)-based system, components and function-oriented pages. The operator obtains the monitoring and controller information necessary for particular operation modes by navigating information that is hierarchically distributed over five or more display pages. Due to the distraction from these additional tasks, reaction time is delayed and it is difficult to maintain the cognitive context, which can cause cognitive error in emergencies.

Therefore, it is necessary that the structure used for presentation of information and visualization should be designed to minimize the cognitive burden of recognition, memory, and judgment on the relationship between complex systems when an unexpected event occurs. The information structure and visualization design should be based on careful analysis and understanding of the work domain of the operator.

To solve the above problem, the monitoring system designer has to decide how to reduce the amount of information to be displayed on the screen so that the operators can work more easily. It is also important to provide a means for the operator to move easily among the information display pages and to make a natural cognitive transition to display page with a high degree of information hierarchy.

Two years ago, we proposed a monitoring and controller display method for each operation mode based on log analyses of operator actions to overcome the above problems [1, 2]. This method minimizes the existing information navigation by providing the monitoring and control means necessary for the operator to perform a specific operation mode on one or two display pages. The method provides the monitoring and controller information for each operation mode of an NPP to operators based on understanding of their work domains.

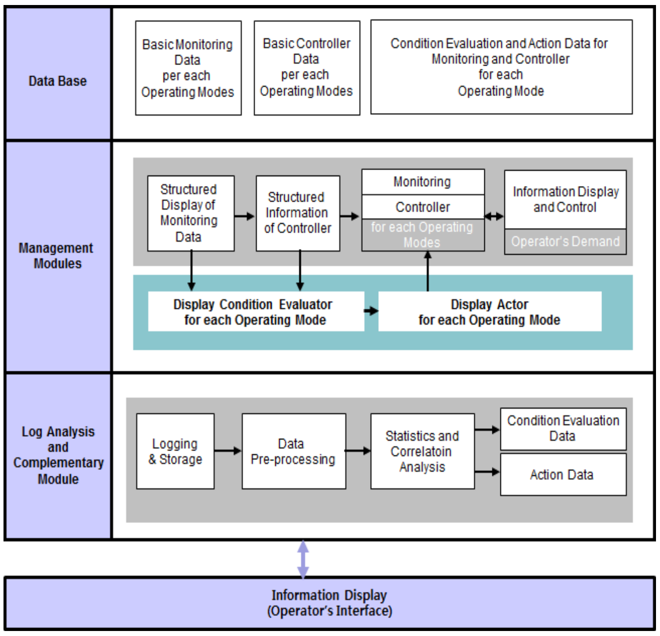
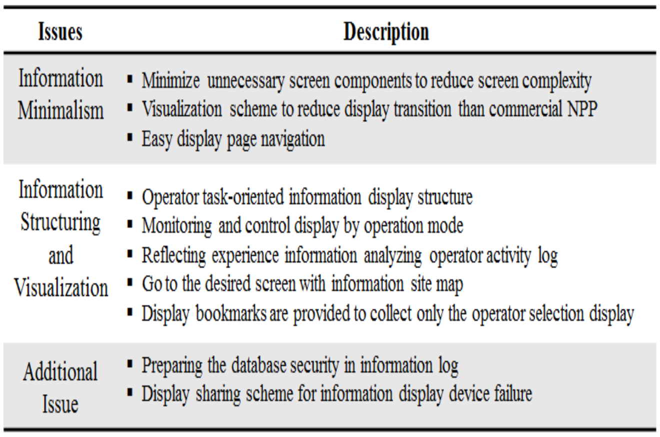

Data Base
Basic Monitoring Data per each Operating Modes
Basic Controller Data per each Operating Modes
Condition Evaluation and Action Data for Monitoring and Controller for each Operating Mode
Management Modules
Structured Display of Monitoring Data
Structured Information of Controller
Monitoring
Controller
for each Operating Modes
Information Display and Control
Operator's Demand
Display Condition Evaluator for each Operating Mode
Display Actor for each Operating Mode
Log Analysis and Complementary Module
Logging & Storage
Data Pre-processing
Statistics and Correlatoin Analysis
Condition Evaluation Data
Action Data
Information Display (Operator's Interface)

Fig. 1. Method for Monitoring and Controller Display for each Operation Mode based on Log Analyses of Operator Actions

Moreover this method complements the monitoring information and the means of control for each operation mode through analysis of the operation behavior logs obtained during operation, to improve the efficiency of display for each operation mode (see Fig.1).

In this paper, we propose more detailed functions and design issues of the information minimalism concept than in the previous paper (see Fig.2). We also propose methods for prototyping and testing to verify it.



Issues	Description
Information Minimalism	▪ Minimize unnecessary screen components to reduce screen complexity ▪ Visualization scheme to reduce display transition than commercial NPP ▪ Easy display page navigation
Information Structuring and Visualization	▪ Operator task-oriented information display structure ▪ Monitoring and control display by operation mode ▪ Reflecting experience information analyzing operator activity log ▪ Go to the desired screen with information site map ▪ Display bookmarks are provided to collect only the operator selection display
Additional Issue	▪ Preparing the database security in information log ▪ Display sharing scheme for information display device failure

Fig. 2. Design Ideas and Issues

II. PROTOTYPE DEVELOPMENT STRATEGIES

The development principles of the prototype needed to verify the information minimalism concept are as follows:

- Standardization and modularization of the development process by improving development productivity and reuse
- Establishment of a prototype product sharing system
- Establishment of prioritized development items and step-by-step development
- Identification of common risk factors and design security for safe software development
- Performing objective risk analysis and testing of all prototyping outputs

Satisfying the development principles described above, required the following prototype development strategies.

A. Component Based Development (CBD)

The process of developing S/W components and component-based systems differs in many significant ways from the "classical" development process of software systems. The main difference is in the separation of the development process of components from the development process of systems. The primary idea of the component-based approach is the reuse of the existing components instead of adding new ones, thereby building systems from already existing components, whenever possible [3]. Fig. 3 shows a detailed V-shape development process for CBD. Fig. 4 and Fig. 5 show components and a component diagram, respectively, for this prototype according to the CBD method.

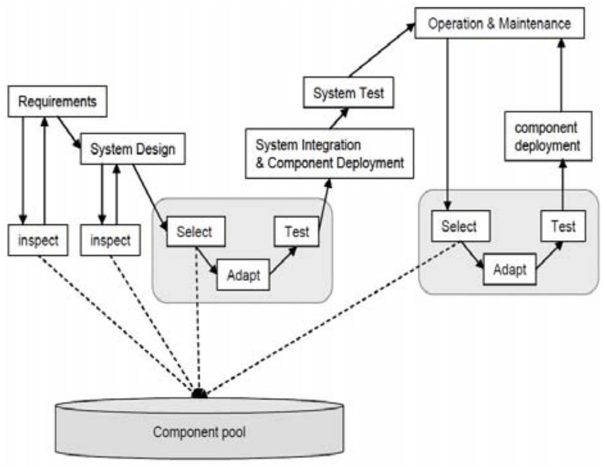


Fig. 3. Detailed V-shape Development Process for CBD

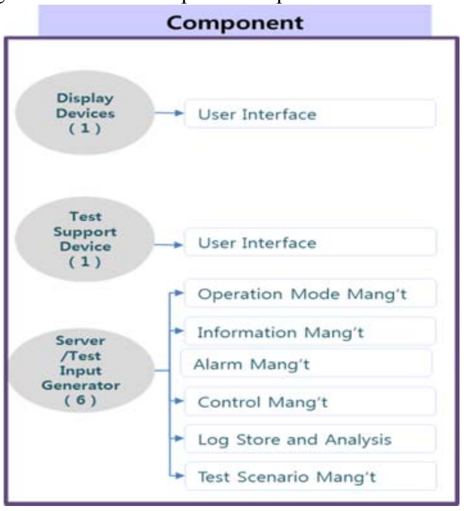


Fig. 4. Components

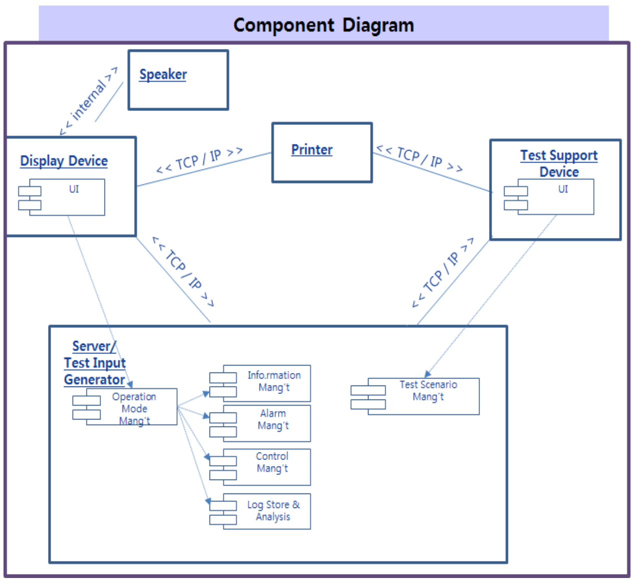


Fig. 5. Prototype Component Design

Because this CBD-based prototyping aims for increased reusability of existing components, the efforts needed for their implementation decreases, and the effort for system verification increases. The components of the prototype will be used as inputs for future human factor engineering validation and verification mock-up of a new Korean reactor.

B. Seven Touchpoints of Secure Software

The control and monitoring system of a nuclear power plant is operated in a closed network within the external network. However the non-safety monitoring system is linked to the general business network to provide information about operation of the nuclear power plant to a large number of users. In addition, non-safety monitoring systems use commercial H/W and S/W, so there is a high probability of malicious and non-malicious cyber infestation.

Therefore, in this paper, we propose a secure software development process for the non-safety monitoring system of NPP through this prototyping.

The secure SDLC (Software Development Life Cycle) process ensures that security assurance activities such as penetration testing, code review, and architectural analysis are an integral part pf the development effort. Software security touchpoints are based on good software engineering practice and involve explicitly pondering security throughout the software lifecycle. This means knowing and understanding common risks (including language-based implementation bugs and architectural flaws), designing for security and subjecting all software artifacts to thorough, objective risk analyses and testing. “Software Security Touchpoints” specifies one set of touchpoints and shows how software practitioners can apply them to the various software artifacts produced during software development [4]. This secure method consists of seven major points in the S/W SDLC (see Fig. 6).

1) Code review
2) Risk analysis
3) Penetration testing
4) Risk-based security test
5) Abuse cases
6) Security requirements
7) Security operation

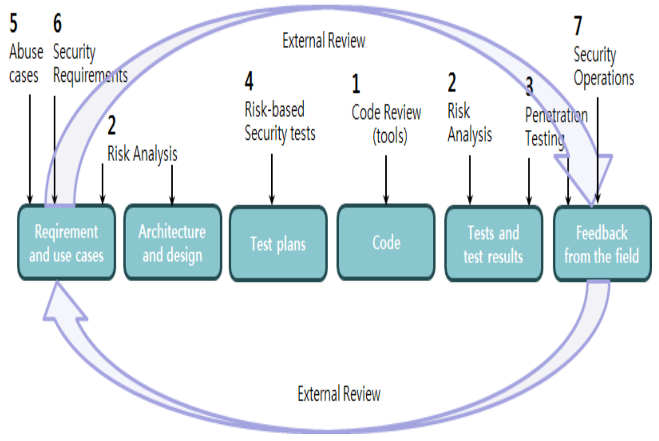


Fig. 6. Seven Touchpoints of Secure Software

TABLE I shows a secure SDLC plan for this prototype based on the seven touchpoints of secure software.

TABLE I
A Secure SDLC Plan for the Prototype

SDLC	Secure Plan	Result
Requirement and use cases	• Converting both overt functional security(e.g., use of applied cryptography) and emergent security propertied (as revealed by abuse case and attack patterns) • Describe the system's behavior when under attack to clarify what areas and components of the software-based system need to be protected, from which threats, and for how long	S/W Req'ts Spec.
Architecture and Design	• Includes documenting assumptions and identifying possible attacks, and uncovering and ranking architectural flaws for mitigation. Recurrent risk tracking, monitoring, and analysis should be ongoing throughout the life cycle	S/W Design Spec.
Test Plans	• Includes testing of security functionality using standard functional testing techniques, and risk-based security of the software as a whole, with test scenario based on attack patterns	Test Plan & Proced ure
Code	• Entails use of static analysis tools to detect common vulnerabilities	
Test and Test Results	• With the architectural risk analysis results driving the selection and implementation of “canned” black-box tests offered by automated application security and penetration testing tools	Test Report
Feedback from the Field	• After development, this includes monitoring the behavior of the fielded software system for indications of attacks and exploits against the software. Knowledge gained through monitoring attacks and exploits should be cycled back into the other touchpoints	S/W Req'ts Spec.

III. PROTOTYPING METHOD

A. Hardware and Software Configurations of the Prototype

The H/W configuration of the prototype follows in Fig. 7 and the S/W configuration of the prototype follows in Fig. 8.

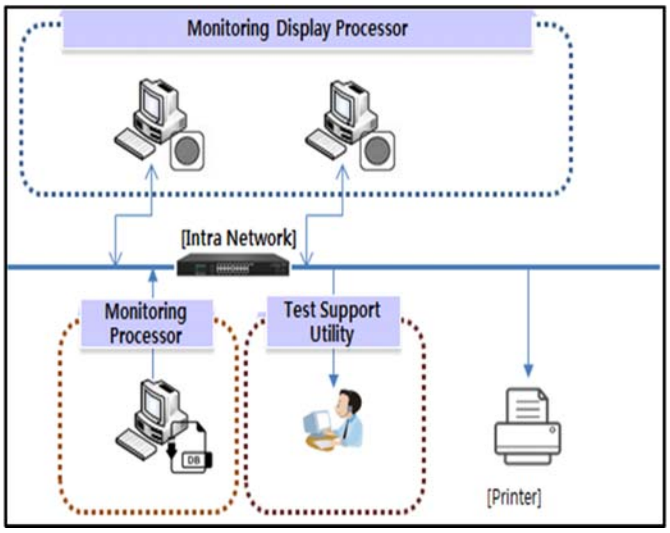


Fig. 7. Hardware Configuration

	Server	Client
H/W	Monitoring Processor	Display Processor Test Support utility
OS	UNIX Ubuntu	Windows 10
S/W	Apache Web Server Database (MySQL)	Web Browser
Development Tools	JSP	JAVASCRIPT HTML SCRIPT

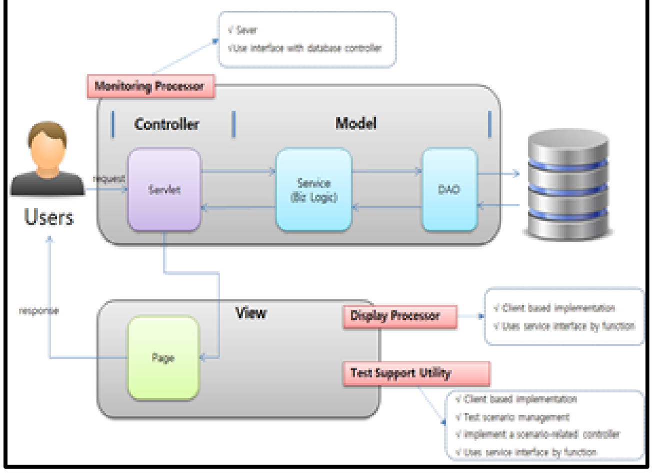

√ Sever
√Use interface with database controller
Monitoring Processor
Controller
Model
request
Servlet
Service (Biz Logic)
DAO
Users
response
View
Page
Display Processor
√ Client based implementation
√ Uses service interface by function
Test Support Utility
√ Client based implementation
√ Test scenario management
√ implement a scenario-related controller
√ Uses service interface by function

Fig.8. Software Configuration

B. Security Vulnerabilities of the Prototype

The expected security vulnerabilities of the prototype are as follows in Fig. 9. These vulnerabilities are managed according to the plans in TABLE I.

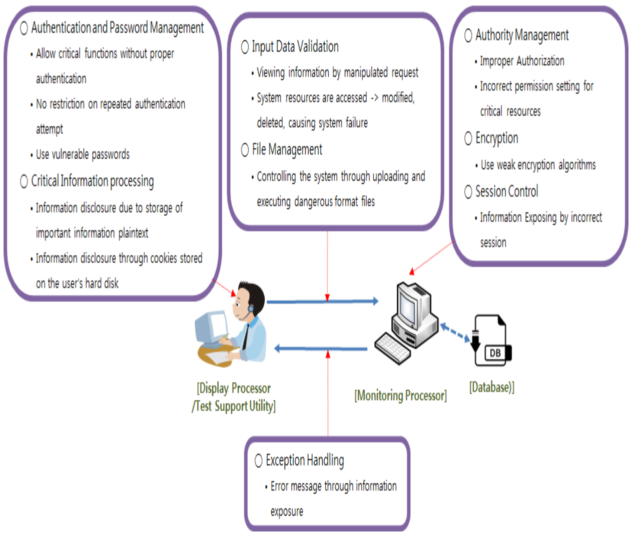

○ Authentication and Password Management
• Allow critical functions without proper authentication
• No restriction on repeated authentication attempt
• Use vulnerable passwords
○ Critical Information processing
• Information disclosure due to storage of important information plaintext
• Information disclosure through cookies stored on the user's hard disk
○ Input Data Validation
• Viewing information by manipulated request
• System resources are accessed -> modified, deleted, causing system failure
○ File Management
• Controlling the system through uploading and executing dangerous format files
○ Authority Management
• Improper Authorization
• Incorrect permission setting for critical resources
○ Encryption
• Use weak encryption algorithms
○ Session Control
• Information Exposing by incorrect session
[Display Processor /Test Support Utility]
[Monitoring Processor]
DB
[Database)]
○ Exception Handling
• Error message through information exposure

Fig. 9 Expected Security Vulnerabilities of the Prototype

C. Display Design of the Prototype

The monitoring and controller display for each mode based on log analyses of operator actions, is as follows in Fig. 10. The first page is the monitoring and controller display page of a specific operation mode. When an operator clicks a controller name in the controller list, the controller pops-up on the page. The second page is the statistics and correlation analysis page. The operator analyzes the operator activity log information page and adds or deletes monitoring and controller information for a specific operation mode set in the past.

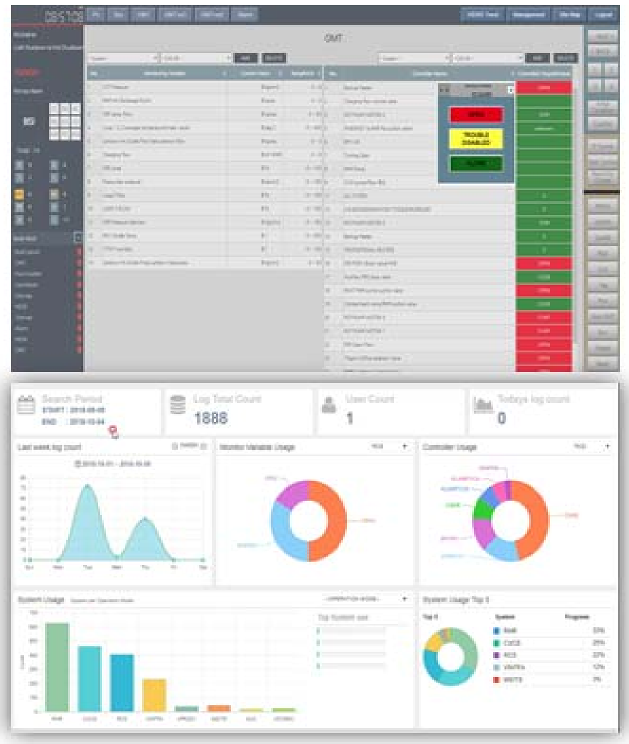
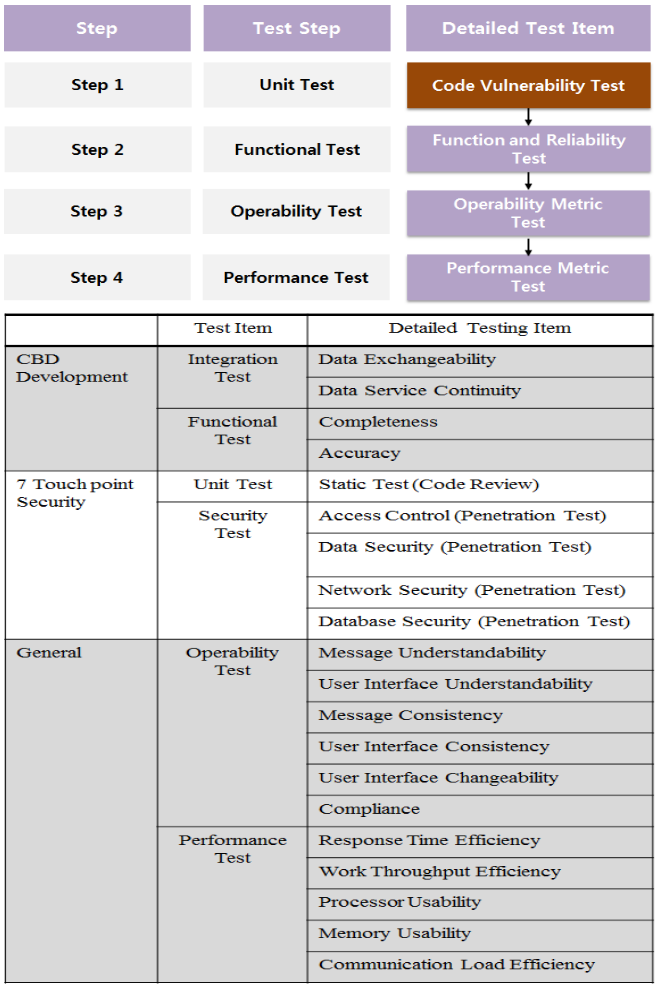


Fig.10. Display Design Examples of the Prototype

IV. TESTING METHOD

The test steps of this prototype include unit test, functional test, operability test and performance test as shown in Fig. 11. Moreover the detailed test items for each test step according to the CBD based cyber security development process are as shown in Fig. 11.



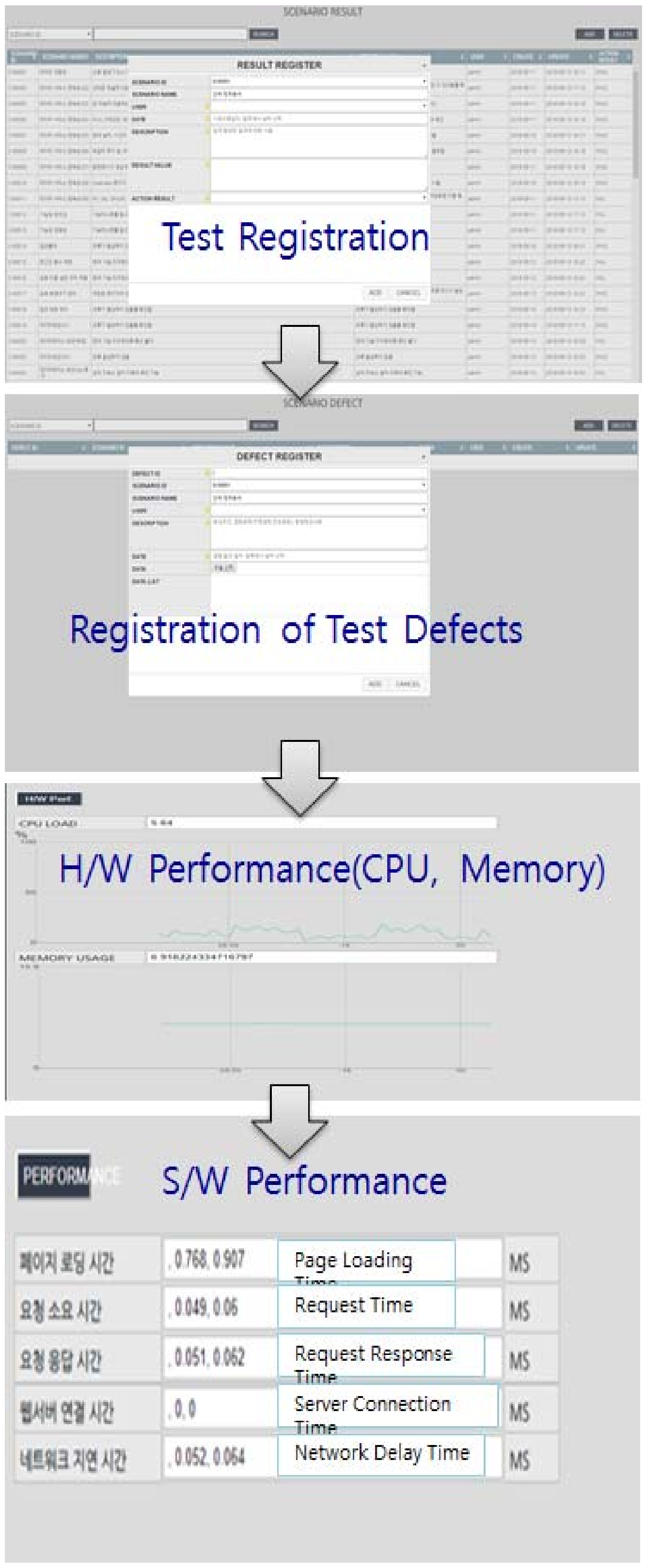
Step	Test Step	Detailed Test Item
Step 1	Unit Test	Code Vulnerability Test
Step 2	Functional Test	Function and Reliability Test
Step 3	Operability Test	Operability Metric Test
Step 4	Performance Test	Performance Metric Test

	Test Item	Detailed Testing Item
CBD Development	Integration Test	Data Exchangeability
		Data Service Continuity
	Functional Test	Completeness
		Accuracy
7 Touch point Security	Unit Test	Static Test (Code Review)
	Security Test	Access Control (Penetration Test)
		Data Security (Penetration Test)
		Network Security (Penetration Test)
		Database Security (Penetration Test)
General	Operability Test	Message Understandability
		User Interface Understandability
		Message Consistency
		User Interface Consistency
		User Interface Changeability
		Compliance
	Performance Test	Response Time Efficiency
		Work Throughput Efficiency
		Processor Usability
		Memory Usability
		Communication Load Efficiency

Fig. 11. Test Design

The test support utility supports a unit test, performance test, and system utilization analysis test of the prototype, as shown in Fig. 12.

1) *Unit Test*
   - Check the coding rules of the source code
   - Tool : LDRA
2) Performance Test
   - Check the response time
   - Tool : Wireshartk
3) System Utilization Analysis Test
   - Check CPU, memory usage
   - Tool : perfmon.msc



페이지 로딩 시간	0.768, 0.907	Page Loading Time	MS
요청 소요 시간	0.049, 0.06	Request Time	MS
요청 응답 시간	0.051, 0.062	Request Response Time	MS
웹서버 연결 시간	0, 0	Server Connection Time	MS
네트워크 지연 시간	0.052, 0.064	Network Delay Time	MS

Fig. 12. Test Support Utility