Vol. 3, No. 1, pp. 38-46, June 2004

INTERNATIONAL JOURNAL OF SAFETY
www.kosos.or.kr

The Software Verification and Validation Tasks for a Safety Critical System in Nuclear Power Plants

Se Woo Cheon, Kyung Ho Cha, and Kee Choon Kwon

Korea Atomic Energy Research Institute, 150 Deokjin, Yuseong, Daejeon, 305-353 Korea

(Received February 10, 2004; Accepted May 30, 2004)

Abstract : This paper introduces the software life-cycle V&V (verification and validation) tasks for the KNICS (Korea nuclear instrumentation and control system) project. The objectives of the V&V tasks are mainly to develop a programmable logic controller (PLC) for safety critical instrumentation and control (I&C) systems, and then to apply the PLC to developing the prototype of an engineered safety features-component control system (ESF-CCS) in nuclear power plants. As preparative works for the software V&V, various kinds of software plans and V&V task procedures have been developed according to the software life-cycle management. A number of software V&V tools have been adopted or developed to efficiently support the V&V tasks. The V&V techniques employed in this work include a checklist-based review and inspection, a requirement traceability analysis, formal verification, and life-cycle based software testing.

Key words: nuclear power plants, safety critical systems, software, verification and validation

1. Introduction

Safety critical systems are those in which a failure can have serious and irreversible consequences. For the past two decades, digital technology has been applied rapidly to the safety critical instrumentation and control (I&C) systems for railways, airplanes, vehicles, communication networks, and so on. In the nuclear industry this attempt has also been attempted by applying the digital technology to nuclear I&C systems. The programmable logic controller (PLC) based platforms (e.g., Teleperm XS, Common Q, and Tricon) have been prototyped and evaluated for nuclear safety applications [1].

Because the localization of the qualified PLCs for the safety critical I&C systems have not been accomplished in Korea, we are carrying out the KNICS (Korea nuclear instrumentation and control system) project which is focused on realizing the localization of qualified PLCs for the safety critical I&C systems and applying these PLCs to developing the prototype of an engineered safety features-component control system (ESF-CCS). The PLC has been developed to provide various communication networks, a strict real-time performance and a high reliability. The safety functions in the ESF-CCS are implemented as safety critical software, where those functions require a high reliability and quality.

This paper introduces the software verification and validation (V&V) tasks for one of the safety critical systems (i.e., ESF-CCS) in nuclear power plants (NPPs), according to the software life-cycle process of NUREG-0800 [2]. The main activities of the V&V process are the preparation of the software plans and V&V task procedures, verification of the software requirement specification (SRS), software design specification (SDS), codes, and the testing of the integrated software and system. In order to support the V&V tasks efficiently, tools such as the SIS-RT and NuSCM have been proprietarily developed, and commercial tools such as the Statemate MAGNUM, Statemate ModelChecker and Statemate ModelCertifier, Cantata++ and McCabe Test have been adopted.

The V&V techniques which have been employed to the life-cycle V&V task phases (i.e., requirement, software design, implementation, and integration) include a checklist-based review and inspection, a requirement traceability analysis, formal verification by introducing a specification of the simulation model and model check-

*Corresponding author: swcheon, khcha,kckwon@kaeri.re.kr

ing, software testing, software safety analysis, and software configuration management. Various kinds of V&V tools have been used in carrying out the V&V tasks. A lot of software V&V checklists are derived from the software V&V criteria and the requirements from the BTP HICB-14 [3], IEEE Std 7-4.3.2 [4], IEEE Std 1012 [5], IEEE Std 1028 [6] and IEC-60880 [7].

2. Brief Descriptions of the Safety Critical System Prototype

2.1. The System Architecture

The ESF-CCS initiates several emergency actuations to prevent the plant from a hazardous state in the event of a reactor trip [8]. The actuations include a safety injection, containment isolation, main steam line isolation, auxiliary feedwater injection, and a containment spray actuation. The ESF-CCS mainly consists of group controllers (GCs), loop controllers (LCs), the communication interface and test processor (CITP), the cabinet operator module (COM) and the intra-division network (IDN).

The ESF-CCS is designed as a PLC-based architecture with four redundant divisions (i.e., A, B, C, and D), as shown in Fig. 1. The software part of the prototype is implemented on the qualified PLCs (called POSAFE-Q) via the proprietarily developed pSET engineering tool.

Figure 2 shows the developed PLC prototype, which mainly consists of power modules, a processor module, communication modules, and I/O modules. Table 1 compares the features of the prototype PLC with the features of other PLCs.

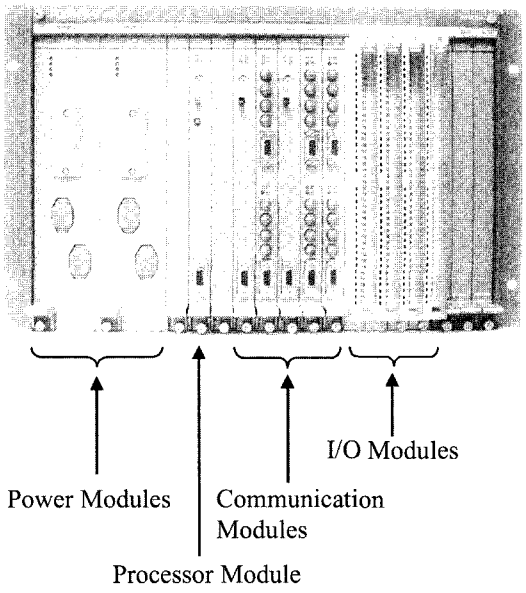


Fig. 2. The safety critical PLC prototype.

2.2. The Software Classification

The safety components in Table 2 are implemented for the safety functions both in the PLC and the ESF-CCS. These components are categorized as safety critical (SC), safety related (SR), and none safety (NS) components. The engineering tool, pSET, is used for developing the functional block diagrams (FBDs) and for downloading the FBD-based programs into the POSAFE-Q PLCs via RS-232C.

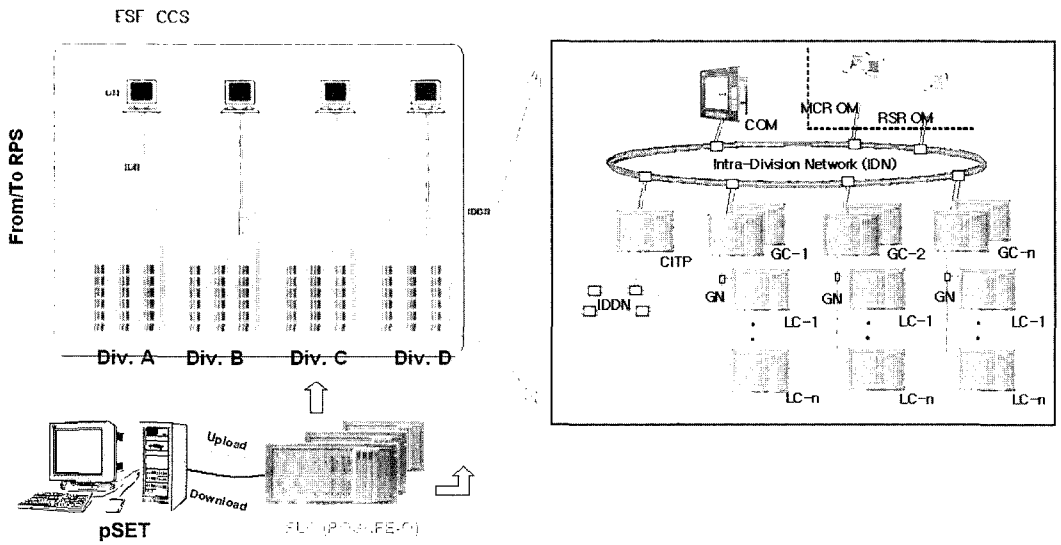


Fig. 1. The overall architecture of the ESF-CCS prototype.

Table 1. The main PLC features

Category \ PLC		Advant AC-160	Teleperm XS	Triconex	POSAFE-Q
Processor Module	CPU	MC68360	Intel 80486 DX 32 bit	Motorola MPC860	TI DSP SMQ320C32PCM 32bit
	Clock (MIPS)	33 MHz (6 MIPS)	33 MHz (20 MIPS)	50 MHz (66 MIPS)	60 MHZ (30 MIPS)
	Floating Point	Very Low	Low (30M FLOPS)	High	High (60M FLOPS)
Analog Input Module	Accuracy	±0.2%	±0.2%	±0.15%	±0.1%
	Update Time (Resolution)	40 ms(12 bit)	2 ms(16 bit)	40 ms(12 bit)	40 ms(12 bit)
Pulse Count Module	Input Max. Fq.	100 KHz	25 KHz	20 KHz	100 KHz
Safety Comm. Module	Processor	MC68360 QUICC 25 MHz	V25 20 MHz	None	EC1 48 MHz
	Protocol (Speed)	High Speed Link (3.1 MBPS)	Profibus-FDL (1.5 MBPS)	None	Profibus-FDL (5 MBPS)

Table 2. Software classification for the PLC and ESF-CCS

Systems	Software Components		Classification
PLC	Processor Module RTOS	pCOS	SC
		System Task	SC
	Communication Module	Profibus-FDL1	SC
		Profibus-FDL2	SC
		Profibus-FMS	SR
	I/O Module	Analog Input OS (PIAOS)	SC
		Analog Output OS (PIAOS1)	SC
	pSET	Interpreter, Linker	SR
		Compiler, Loader	SR
		Editor, Debugger	NS
		Simulator	NS
ESF-CCS	Group Controller(GC) Loop Controller (LC)	Coincidence logic (Control module)	SC
	Comm. Interface & Test Processor(CITP)	On-line automatic periodic test logic, Interface	SR
	Cabinet Operator Module (COM)	Communication	SR
		Man-machine Interface	SC

*SC: Safety Critical, SR: Safety Related, NS: None Safety

3. Preliminary Works for the Software V&V Tasks

3.1. Related Codes and Standards

One of the main purposes of the software V&V is to acquire a license from the regulatory authority. It is crucial for the V&V process to meet the regulatory requirements as well as the design goals. The software V&V criteria and requirements are based on the codes and standards including BTP HICB-14 [3], Reg. Guide 1.170 [9], IEEE Std 1012 [5], IEEE Std 7-4.3.2 [4], IEEE Std 1008 [10], IEEE Std 829 [11], IEEE Std 1028 [6], etc.

3.2. Implementation of the Software Planning Documents

To successfully carry out the V&V tasks, we have developed various kinds of software plans, as shown in Fig. 3. The software V&V plan addresses the issues of team organization, master schedule, software integrity level, management of the V&V, life cycle V&V activities, V&V reporting and documentations, etc.

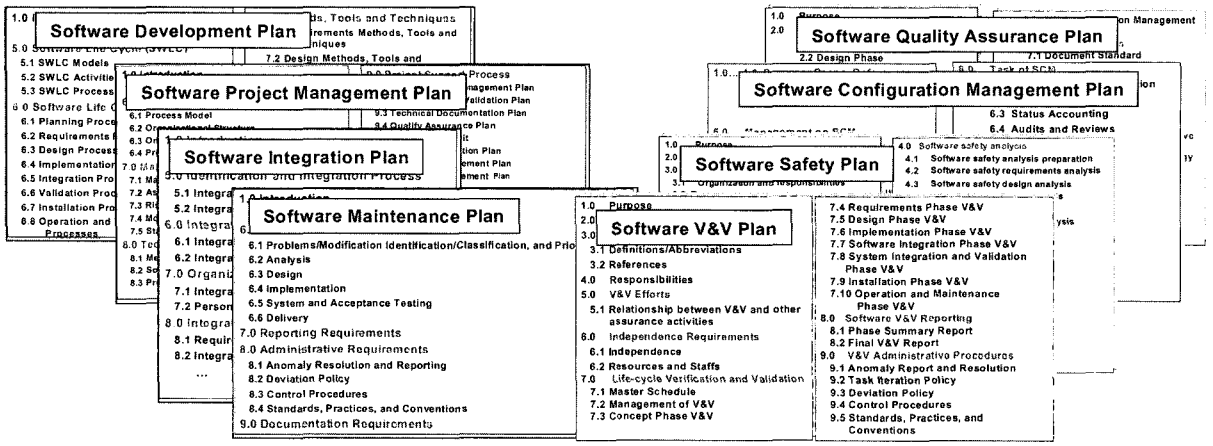


Fig. 3. Development of the software plans.

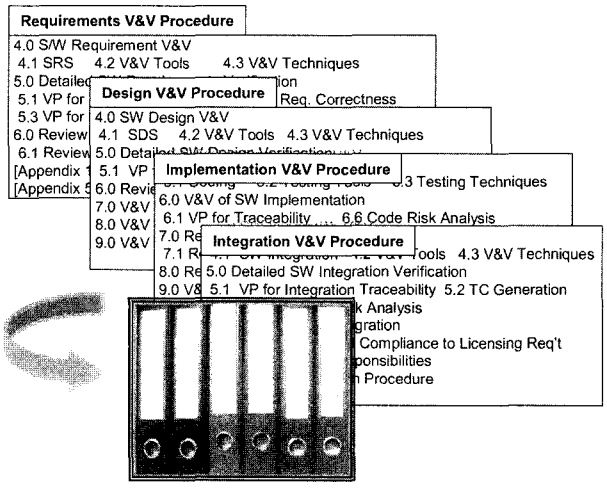


Fig. 4. Development of the software V&V procedures.

3.3. Implementation of the Software V&V Task Procedures

The V&V process provides an objective assessment of the software products and processes throughout the software life cycle. As shown in Fig. 4, we have developed software V&V procedures for the software requirement, design, implementation, and integration phases.

The V&V procedures provide specialized checklists for the life cycle V&V tasks, V&V methods employed, supporting V&V tools and the required inputs and outputs. The V&V procedures also include testing procedures.

3.4. Software V&V Tools

We have used various kinds of software V&V tools (both proprietary and commercial) for the V&V tasks, as shown in Fig. 5. The SIS-RT (software inspection supporting-requirement traceability) tool, which has been proprietarily developed for the KNICS project, is used to support a systematic review and inspection of the document-based design outputs (e.g., functional requirements (FR), SRS, SDS) and to support the traceability analysis between the source and the target documents [12].

The statechart-based tool, Statemate MAGNUM, can rapidly design and validate complex systems level products through a unique combination of graphic modeling and simulation [13]. The Statemate ModelChecker and Statemate ModelCertifier are statechart-based formal verification tools [14]. These tools allow systems designers to validate their Statemate models, ensure that they follow good design practices, and prove that they meet user defined critical properties, such as safety. It allows system engineers to ensure that their designs perform correctly under all circumstances, both those that are expected, as well as those that are unexpected. These tools are used for specifying the software requirements of the ESF-CCS and verifying the statechart-based formal specifications.

Cantata++ is a commercial tool for automating software testing tasks and is to be applied to the dynamic software testing [15]. This tool has been designed around the requirements of the C/C++ languages to produce a tool which allows developers to efficiently perform component and integration testing.

The McCabe Test is also a commercial software tool for automating testing tasks and is to be applied to the static testing [16]. The tool provides comprehensive test coverage; penetrating the complexity of software systems to focus, monitor, and document software testing processes.

Finally, NuSCM (i.e., nuclear software configuration management tool) is proprietarily developed to manage

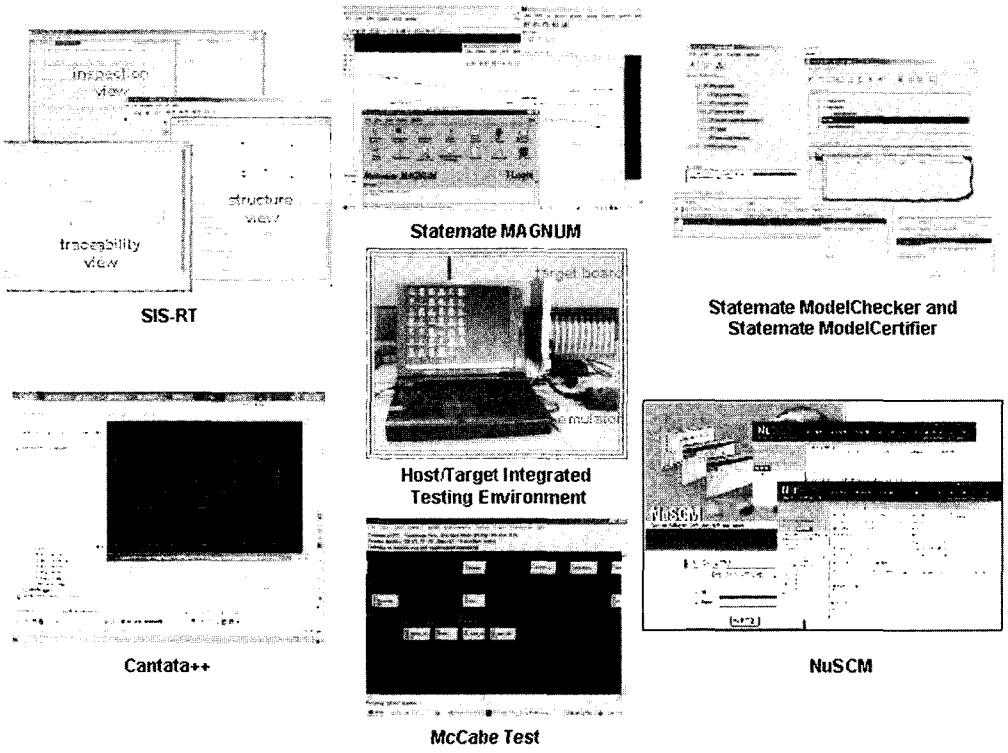


Fig. 5. Software V&V tools used in KNICS.

the software configuration activities in the KNICS project [17]. NuSCM is based on the systematic management of the software design documents and source codes based on the projects/activities.

4. Software V&V Tasks for the ESF-CCS Prototype

Figure 6 shows the overall software life-cycle V&V tasks which have been carried out for the software V&V. Software V&V techniques include a review and inspections, a traceability analysis, formal verification by introducing a specification of the simulation model and model checking, software testing, software safety analysis, and software configuration management. Various kinds of V&V tools have been used in carrying out the V&V tasks.

4.1. Software Requirement Phase

The review and inspections are popular V&V techniques. The checklist-based review and the formal Fagan [18] inspection method are applied to the overall phases of the software life-cycle. The Fagan inspection process consists of seven steps, i.e., planning, overview, preparation, inspection (analysis) meeting, rework, and a follow up. Inspection team is composed of a moderator, a recorder, a reader, an author, and an inspector (or a tester).

The software requirements should satisfy the acceptance criteria according to BTP HICB-14 [3]. As shown in Fig. 7, the criteria is divided into two sets: i.e., the functional characteristics (i.e., accuracy, timing, safety, security, robustness, reliability and functionality) and the process characteristics (i.e., completeness, consistency, correctness, style, traceability, unambiguity and verifiability). The right side of Fig. 7 shows a sample of the checklist-based review and inspection table, in which

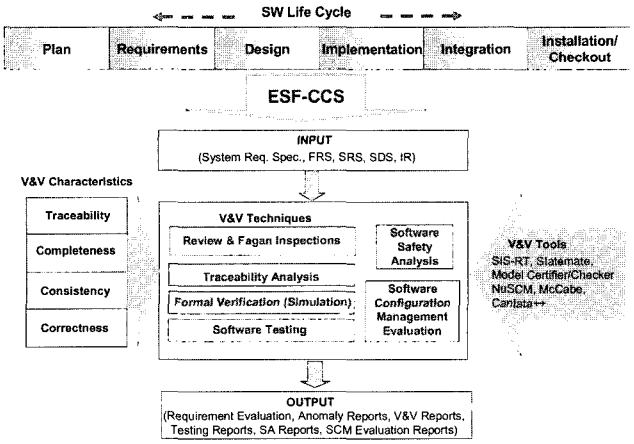
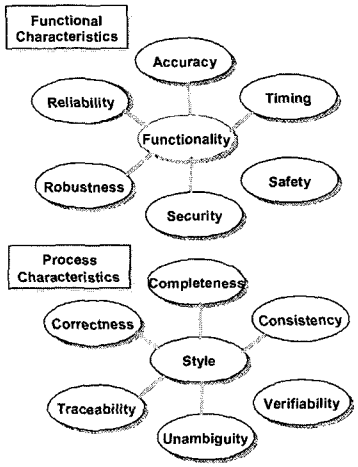


Fig. 6. Overall software life-cycle V&V tasks.



Categories	Criteria	Checklists		Review Comments
Completeness for definitions of SW function	BTP HICB-14 Ch.3.3a	Are software functions and their behaviors specified for each operating mode?	Normal Operating Mode	* Fig. 5.1 does not specify heart-beat signals. *.....
			Operating Bypass	
			Adjusting Setpoints	
	IEC 60880 4.C.1	Are change conditions between operating modes specified for each operating mode?		
Completeness for definitions of SW behavior	BTP HICB-14 Ch.3.3a	Are each functional requirement specified for task sequence, behavior and event and timing?		

Fig. 7. A sample of the checklist-based review and inspection table.

the checklists were categorized based on the fourteen items of the review characteristics.

The traceability analysis between the functional requirements of the system and the requirements from the SRS has been made by using the SIS-RT tool. The tool can compare the similarity chains between the source and the target requirements on the traceability view, as shown in Fig. 8. We have performed the inspection and traceability analysis by focusing on three important properties: i.e., completeness, correctness and consistency (i.e., 3Cs).

The software requirements are formally specified by using the statechart method. The Statemate MAGNUM tool, which employs the statechart method, can create a visual, graphical specification that clearly and precisely represents the intended functions and behavior of the requirements being specified. The statechart specification can be executed, or graphically simulated, so the verifier can explore what-if scenarios to determine if the behavior and the interactions between the system elements are correct. Model checking is a method for formally verifying the finite-state concurrent systems. Figure 9 shows the examples of the statechart formal specification for the ESF-CCS.

4.2. Software Design Phase

The V&V activities of the software design phase are almost similar to those of the software requirement

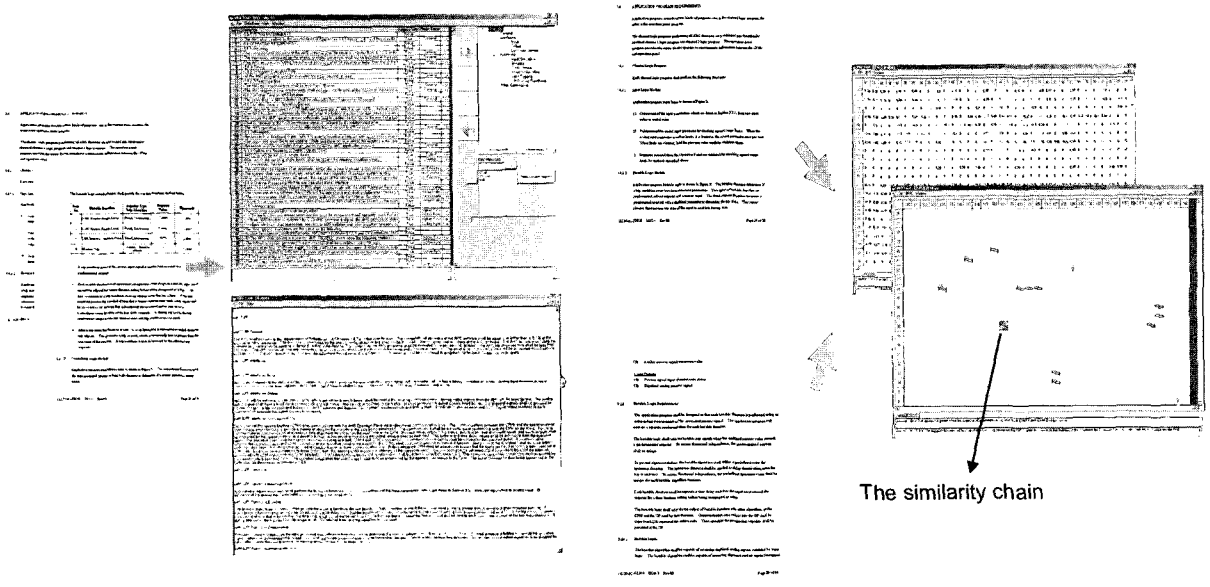
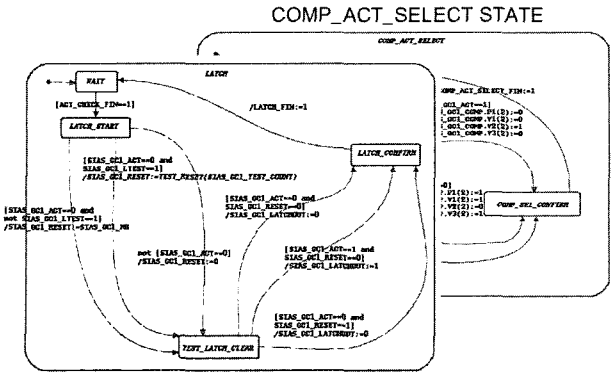


Fig. 8. The examples of the SIS-RT functions.



statechart simulation table

Statechart	No	Trace No	Characteristics	Scope of Statechart	Transition Test Cases	Transition Coverage Paths	Pass/Fail Criteria	Monitor Value	Pass/Fail	Remarks	Recomendations
LATCH	1	Fig 5.1-18	Correctness, Completeness	LATCH	NOT(SIAS_GC1_ACT=0) AND SIAS_GC1_ACT=1 AND SIAS_GC1_RESET=0	WAIT -> LATCH_START -> TEST_LATCH_CLEAR -> LATCH_CONFIRM -> WAIT	SIAS_GC1_RESET=0 AND SIAS_GC1_LATCHOUT=1	0 1	P		Design Change
	2	Fig 5.1-18	Correctness Completeness	LATCH	SIAS_GC1_ACT=0 AND SIAS_GC1_LTEST=1 AND SIAS_GC1_RESET=0	WAIT -> LATCH_START -> TEST_LATCH_CLEAR -> LATCH_CONFIRM -> WAIT	SIAS_GC1_LATCHOUT=1	1	P		

Fig. 9. An example of the statechart formal specification.

phase. Formal verification adopts FBDs to specify the software design. We have developed a technique to translate the FBDs into the input languages of Statemate Model Checker. By using this technique, we have performed the model checking in the software design phase, as shown in Fig. 10.

4.3. Implementation Phase

In the implementation phase, we have also performed the review and inspections, and traceability analysis. However, in this phase, the main activity is software testing, specifically component testing. Software testing is the final barrier to the release of an application into production. The thoroughness of the testing effort directly affects the quality and stability of the program.

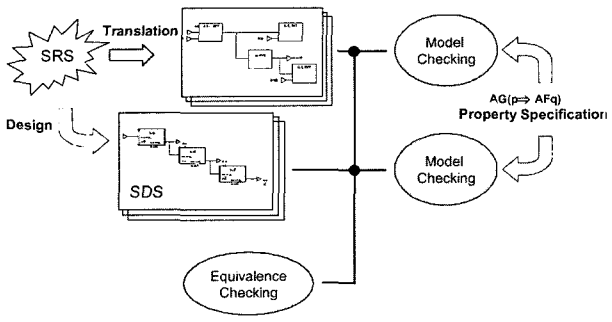


Fig. 10. The schematic diagram of the model checking process.

Component testing is conducted to verify the correct implementation of the software design and its compliance with the software requirements for one software element (i.e., unit and/or module) or a collection of software elements. Component testing of the ESF-CCS software is challengeable because the testing techniques for the FBDs are not mature as yet and the ESF-CCS software is implemented on the PLC by using FBDs. We have developed a component testing technique by defining the test coverage criteria on the FBDs. By using the technique we are deriving the test cases for the component testing of the ESF-CCS software.

In Fig. 11, each testing should follow the software test life cycle tasks (e.g., test plan generation, test design generation, test case generation, test procedure generation, and test execution) [5]. To automate the testing tasks, the McCabe Test and Cantata++ are used.

4.4. Integration Phase

The integration phase can be divided into the software integration phase and the system integration phase. The main V&V activity in the software integration phase is the integration testing: i.e., an orderly progression of the testing of the incremental pieces of the software systems in which the software elements are combined and tested until the software has been integrated to show a compliance with the software requirements and design specifications. System testing is the activities of testing an integrated hardware and software system to verify and validate whether the system meets its original objectives or not. Fig. 12 shows the testing of the ESF-CCS integrated hardware and software at the system integration phase.

4.5. Software Safety Analysis

Since the licensing criteria requires the safety analysis

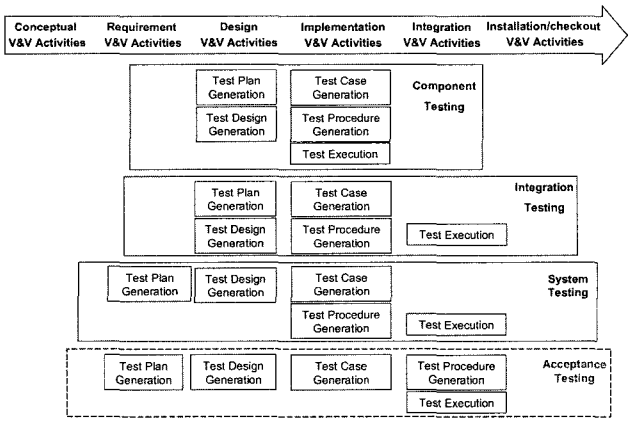


Fig. 11. The life cycle testing process.

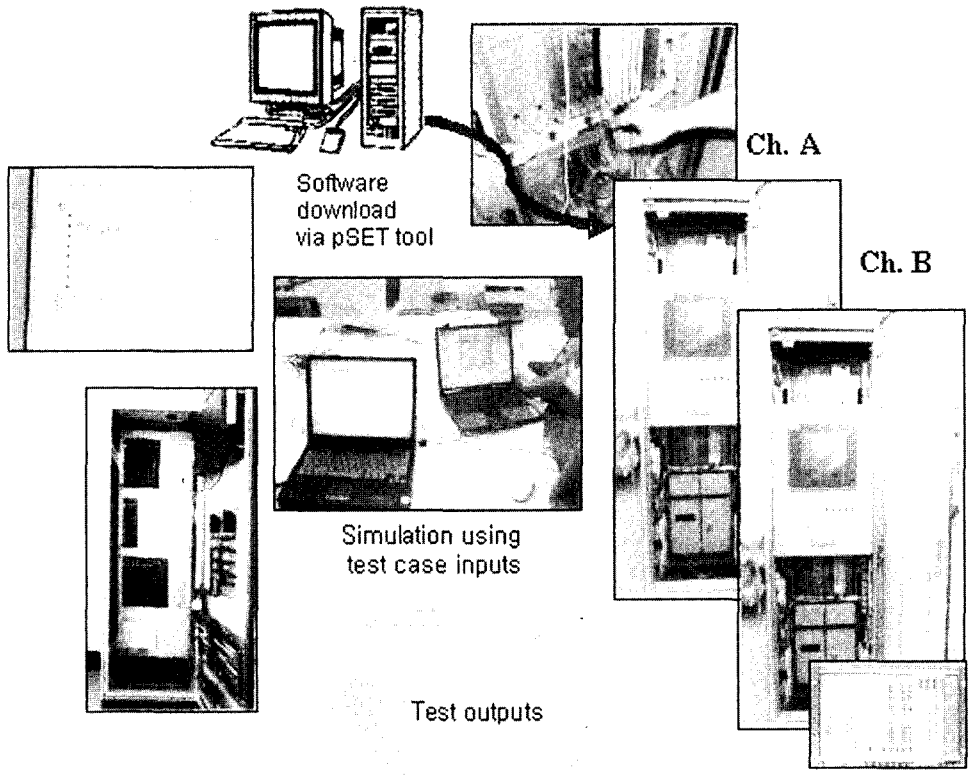


Fig. 12. The testing process.

of the product from each phase of the life cycle, the software safety analysis (SSA) procedure for each phase has been developed as shown in Fig. 13. The procedures include HAZOP (hazard operability) procedures for the requirement and design phases, and fault tree analysis (FTA) procedures for the design and implementation phases. We have developed the guide phrases, checklists and the software HAZOP procedure for the ESF-CCS software.

4.6. Software Configuration Management

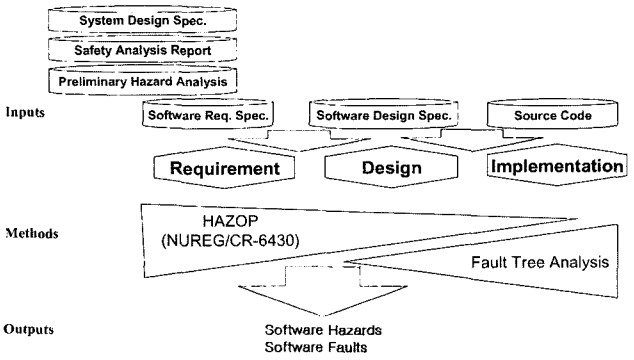


Fig. 13. The software safety analysis procedure.

Software configuration management (SCM) is an activity which configures the form of a software system (i.e., documents, drawings and source codes) and systematically manages and controls the modifications used to compile the plans, development, and operations resulting from the software development and maintenance. The software configuration management tool, NuSCM, has been developed for the software life cycle V&V management of the KNICS project. More detailed descriptions on the NuSCM are found in [17].

5. Conclusions

This paper has discussed the software life-cycle V&V tasks for the ESF-CCS, which is one of the important safety critical systems in NPPs. The tasks includes the preparation of the software plans and V&V task procedures, a checklist-based review and Fagan inspection, the requirement traceability analysis, the formal verification methods and the life-cycle based software testing. The framework of the software V&V process has been established through the on-going KNICS project. It is envisaged to achieve the functionality, performance, reliability and safety that are the V&V objectives of the nuclear safety software in the KNICS project. In addi-

tion, the technical experience in the project is expected to be applied to other safety critical software industries.

Acknowledgments

This work has been carried out for the research entitled "Development of the Licensing Techniques for Digital I&C" as part of the KNICS project, which is under the auspices of the Ministry of Science and Technology (MOST) in Korea.

References

[1] *Proceedings of Digital Instrumentation Upgrades Workshop*, Embedded Meeting of NPIC & HMIT 2004, Columbus, Ohio, Sept. 19, 2004.

[2] NUREG-0800, *Standard Review Plan, Chapter* 7, USNRC, 1997.

[3] BTP HICB-14, Branch Technical Position HICB-14, *Guidance on Software Reviews for Digital Computer-Based Instrumentation and Control Systems*, USNRC, 1997.

[4] IEEE Std 7-4.3.2, *IEEE Standard for Digital Computers in Safety Systems of Nuclear Power Generating Stations*, 1993.

[5] IEEE Std 1012, *Standard for Software Verification and Validation Plans*, 1998.

[6] IEEE Std 1028, *Standard for Software Reviews and Audits*, 1988.

[7] IEC 60880, Software for computers in the safety systems of nuclear power stations, IEC, 1986.

[8] S. T. Kim, S. J. Lee, H. W. Chung, D. K. Chung and C. H. Cho, "The Design and Fabrication of Engineered Safety Features-Component Control System," The 3rd KNS-KIEE Joint Workshop on I&C Technology, Changwon, Korea, Nov. 14, 2003.

[9] Reg. Guide 1.170, *Software Test Documentation for Digital Computer Software Used in Safety Systems of Nuclear Power Plants*, USNRC, 1997.

[10] IEEE Std 1008, *Standard for Software Unit Testing*, 1987.

[11] IEEE Std 829, *Standard for Software Test Documentation*, 1983.

[12] Y. J. You, M. C. Kim, and P. H. Seong, "A Methodology for Improving the SIS-RT in Analyzing the Traceability of the Documents (in Korean)," The Korean Nuclear Society 2002 Spring Meeting, Gwangju, Korea, May 2002.

[13] Statemate MAGNUM, http://www.ilogix.com/.

[14] The Statemate ModelChecker and Statemate ModelCertifier, http://www.ilogix.com/.

[15] Cantata++, http://www.iplbath.com/.

[16] McCabe Test, http://www.mccabe.com/.

[17] S.W. Cheon, K. C. Kwon, C. Youn, H. C. Han and D. H. Kim, "Development of a Software Configuration Management System for Software Life Cycle Management," in Proceedings of the NPIC&HMIT 2004, Columbus, Ohio, Sept. 19-22, 2004.

[18] M.E. Fagan, "Design and Code Inspections to Reduce Errors in Program Development," IBM *Systems Journal*, 15, No. 3, 1976.

[19] S. W. Cheon, K. H. Cha, J. Y. Kim, J. S. Lee, H. S. Sohn, Y, J. Lee and K. C. Kwon, "Software Life-Cycle V&V Tasks for the KNICS Plant Protection System Prototype." 4th ANS Topical Meeting on Nuclear Plant Instrumentation, Controls and Human-Machine Interface Technologies (NPIC & HMIT 2004), Columbus, Ohio, Sept. 19-22, 2004.

[20] K. H. Cha, D. Y. Lee, J. C. Park and K. C. Kwon, "The KNICS Approach for Systematic V&V of Safety Software," 14th Pacific Basin Nuclear Conference (PBNC), Honolulu, Hawaii, March 21-25, 2004.

[21] K. C. Kwon, J. S. Lee, J. Y. Kim, H. S. Sohn, Y, J. Lee, K. H. Cha and S. W. Cheon, "Verification and Validation Process for the Safety Software in KNICS," The Enlarged Halden Program Group Meeting, Sandefjord, Norway, May 9-14, 2004.