DEVELOPMENT METHODOLOGIES FOR A NUCLEAR SAFETY DISPLAY SYSTEM

J.H.Kim[1], S.M.Baek[2], J.H.Cho[3,] C.H.Kim[4], S.D.Sohn[5] and J.H.Yoon[6]
I&C System Engineering Department, KEPCO-E&C, Daejeon, Korea
jhkim10[1],smbaek[2], jhcho[3], kimch[4], sdsohn[5], jhyun[6]@kepco-enc.com

Abstract

The safety display of a nuclear safety system of Advanced Power Reactor 1400(APR1400) has been rated as important to safety grade (SIL 3: Safety Integrity Level 3) [1]. Recently, the regulatory agencies are taking stronger position on safety requirements for safety display system (SDS). To satisfy these requirements, it is necessary to develop a safety critical (SIL 4) grade display system.

This paper presents the result of a feasibility study on two methodologies for safety critical display software development based on industrial personal computer platform. One methodology is an SDS with safety critical grade certifiable operating system and display application developed using commercial development tools that depend on designated graphic libraries. The other is a SDS with safety critical grade certifiable operating system and display software that is independent of Open-GL (Graphic Library) or other vendor specific graphic libraries.

In addition, this paper describes the pros and cons of two development methodologies for securing a nuclear power plant safety critical display system.

1. Introduction

SDS is a graphical user interface for Plant Protection System, Reactor COre Protection System (RCOPS), Qualified Indication and Alarm System, and Engineered Safety Features - Component Control System of Shin-Kori PWR Nuclear Power Plant Units 5 & 6 in Korea.

The safety grade of SDS has been SIL-3 which is distinguished from SIL-4 of Programmable Logic Controller (PLC) based algorithm processing subsystem. For example, the safety grade of Operator Module and Maintenance & Test Panel of RCOPS is SIL-3 while that of the PLC sub-system of RCOPS is SIL-4.

Although these different safety grades have been used and accepted for legacy safety systems, the need to develop a SIL-4 grade SDS has arisen for safety critical display functions to meet strict safety requirement for a new plant.

This paper describes the safety requirements for the design of SDS from a software platform standpoint, and the feasible safety display system development methodologies.

2. Safety System Requirements

The purpose of the feasibility study for the SDS is to see whether the development methodologies could be used to implement its required safety display functions. These development methodologies are designed in accordance with the following IEC 61508 and IEC60880 safety requirements:

2.1 IEC 61508

The IEC 61508 describes a generic approach to all lifecycle activities for safety systems comprised of electrical/electronic and/or programmable electronic (E/E/PE) elements that are used to perform safety functions.

The development of safety system requires processes to ensure the system's safety function. These processes are a) hazard and risk analysis, b) functional safety assessment, c) verification activities, d) validation activities, e) configuration management, and f) incident reporting and analysis (61508-1).

The hazard and risk analysis, verification and validation as well as configuration management have been performed in the feasibility study for SDS. One of things we have focused on is to make sure the developed software is fully testable through verification and validation.

A system, typically has two types of failure: Namely, the systematic failure and random failure (61508-4).

The systematic failure is a failure that happens in a deterministic way due to a certain cause while the random failure is the one that occurs randomly or by an unknown cause.

The former should be defended by fault avoidance and fault control, while the latter should be defended by fault control.

The standard requires the "application of QM (Measures to avoid systematic faults) measures" to avoid failures in the different phases during the cycle of a product. The requirement to apply these measures is dependent on the level of SIL. The failure avoidance measures can be Project managements, Documentation, Separation of safety function from non-safety functions, Semi-formal methods, Checklists, Computer aided specification tools, or Formal methods [2].

The main failure avoidance approach in the feasibility study is to make well defined documentation and using CASE tool through the whole software lifecycle.

2.2 IEC 60880

IEC 60880 provides requirements for the software of nuclear power plants performing functions of safety category A defined in IEC 61226. IEC 60880 corresponds to IEC 61508, part 3 for the nuclear application sector [3].

The requirements are defined based on the software safety lifecycle, where the software can be categorized into system software and application software. The system software is the system kernel including communications, I/O management, standard functions, self-supervision, etc., and the application software is any kind of application specific algorithms like functional logic, display

software, alarm logic, etc. There is no distinction between the requirements for system software and application software.

As one of the software requirements, the self-supervision should be done for the random failure of hardware components, erroneous behavior of software (e.g., deviations from specified software processing and operating conditions or data corruption) and erroneous data transmission between different processing units. The self-supervision is specified as fault detection and self-diagnostics in IEEE7-4.3.2[4], IEC 60880 also requires periodic testing for those components which are not covered adequately by self-supervision. For any fault detection during the diagnosis, the system should maintain the fail-safe condition for SIL-4 display system.

IEC 60880 specifies the software requirements for pre-developed software, development tools as well as security. Especially for Common Cause Failure (CCF), analysis of the potential for software CCF shall be performed and documented at the system level and/or at the level of the total I&C architecture of the I&C systems[5].

2.3 Strategy to Conform to IEC 61508/60880

To ensure the software safety, it is important to set up a well-defined quality assurance plan and make sure that the software is qualified according to the plan. As it is recognized that the reliability of a computer-based safety system cannot be demonstrated only by testing, the quality of processes for the whole system life cycle is an important factor in demonstrating the system safety [6].

For the safety system, the quality assurance plan is to assure the conformance to customer's requirement and safety requirement of the system and the software documentation, development, test, configuration management, verification and validation activities will be performed for each software life cycles to meet the quality requirement. Especially, the software has to be tested completely and successfully through the verification and validation processes to meet the requirements.

The formal method or at least semi-formal method for the whole software development lifecycle has to be adopted for SIL-4 software. The meaning of formal methods interpreted by European nuclear regulators is the use of mathematics and logic to describe systems and designs in a way that makes their mathematical or logical analysis possible [6]. The reason for introducing the semi-formal method is the limited applicability of formal methods in real design. There are limitations on mathematical analysis because it is difficult to combine all features and properties for mathematical analysis of real system, and is even more difficult when the system includes external interfaces.

The most practical method of applying the formal method is to use CASE tools that support the formalism, and any limitation of applying the formal method shall be explicitly documented. The formalism shall also be applied to the OS kernel [7,8,9] of task scheduling, memory management, interrupt handling development and test. The safety of I/O drivers can be demonstrated through the plant operating experience and vendor support documents.

A soft error is the one in a storage medium, which means the value is changed unintentionally and unpredictably. The soft error is a part of the derivation of the Safe Failure Fraction (SFF) that impacts on SIL. The IEC 61508-2010 considers soft errors for Memory Management Unit, Direct

Memory Access, CPU internal Random Access Memory (e.g., program-counter, stack pointer) and variable memories.

The measure to control soft error can be error detection (e.g., parity bit, Error Correcting Code/Error Detection Code and Cyclic Redundancy Check) and temporary redundancy of multiple executions in succession for comparison, but the best measure is the physical redundancy that keeps redundant physical mirrored memory.

3. Safety Display System

3.1 Software Platform

The most important software for safety-critical display is system software. There are two ways in securing the system software: One is to use the Commercial Off-The-Shelf (COTS) vendor-provided safety system software like Wind River's VxWorks, QNX Software system's QNX and PikeOS which obtained IEC 61508 SIL3 certification. The other is to develop a new customized system software or modify existing ones. There are pros and cons for each approach and it seems that both are feasible.

In our feasibility study we have implemented both approach and derived the following:

The merits of the former are the high feasibility, enhanced performance and capability, easiness of verification & validation, easiness of update, and CASE tool compatibility including display CASE tool. The latter has the advantage of having the copyright for the developed software which consequently brings the advantages of budget, life cycle configuration control, autonomous licensing with source code when getting a license from the regulator without getting help from vendor, easy to customize function for application purpose. In the licensing of SDS, the customized system software seems preferable because of the licensing advantage with source code accessibility and system customization.

The following are minimum requirements included in the system software that had been applied in the feasibility study for SDS:

a) Task scheduling: Task should be scheduled using the priority-based preemptive scheduler. The Rate Monotonic Analysis should be used to select priorities, i.e., the shorter period task should have a higher priority. The deterministic execution of tasks should be supported.

b) Interrupt service routine handling: In case of interrupt service request for system abnormality (e.g., arithmetic fault, memory fault, illegal instruction, machine malfunction, data format fault, power restore, etc.), the system software should report the status so that an application can actuate a proper counteraction. The system software for SDS considers only internal interrupts. The external interrupts should not be permitted in the CPU board except for the timer interrupt.

c) Resource management: It should provide protected partition in space. A file management system that can generate and remove file/directory is required.

d) I/O handling: Safety field bus, point-to-point datalink, digital input, digital output, removable media should be supported.

e) Self-diagnostics to detect and report system fault: Memory functionality and integrity test, computer system instruction set tests, computer peripheral hardware test, computer architecture support hardware test, and communication link diagnostics should be provided. If there appears any abnormality in self-diagnostics, the system information for appropriate safety action should be provided.

3.2 Graphic Display

There are two possible approaches to securing the safety display. One is an SDS with safety grade system software and application software developed using commercial display development tools that depend on designated graphic libraries like Open GL; the other is an SDS with safety critical grade system software and display program without using any Open GL libraries. The first approach has the advantage of utilizing various graphic libraries, and the second has the advantage of avoiding the graphic card certification for the use of safety critical display.

3.2.1 Open GL

The Open GL or other pre-developed graphic libraries can be provided using a graphic card. In this approach the most difficult part is the graphic card licensibility for use in SDS. There can be an Open GL or Open SC GL (Safety-Critical GL for safety system) implementation by software only without using the graphic card, but this approach has limitations, such as display CASE tool compatibility and low processing speed compared to Open GL in graphic card.

We have used SCADE Display and SCADE Suite for the feasibility study of graphic display with Open GL. The SCADE Display and SCADE suite are SIL-3 certified software development tools that can be used in safety software developments. The graphic display is implemented using SCADE display and the logic calculation (e.g., callback functions) for display using SCADE suite.

Figure 1 shows the overall architecture of SDS configuration. The display application consists of two parts; a Control Logic and a Graphic Display. The Control Logic performs the callback functions and the Graphic Display performs graphic display on the screen. The source codes for both parts are generated automatically from the tool based on designed model and additional hand codes for I/O handling, communication, and platform-specific functions are needed.

The source codes that are generated automatically from the SCADE can be considered qualified for the safety system, but the hand codes and the graphic cards should be qualified as safety grade codes as is described in Section 2.3. The system software including Board Support Package is also needed to be qualified as safety grade.

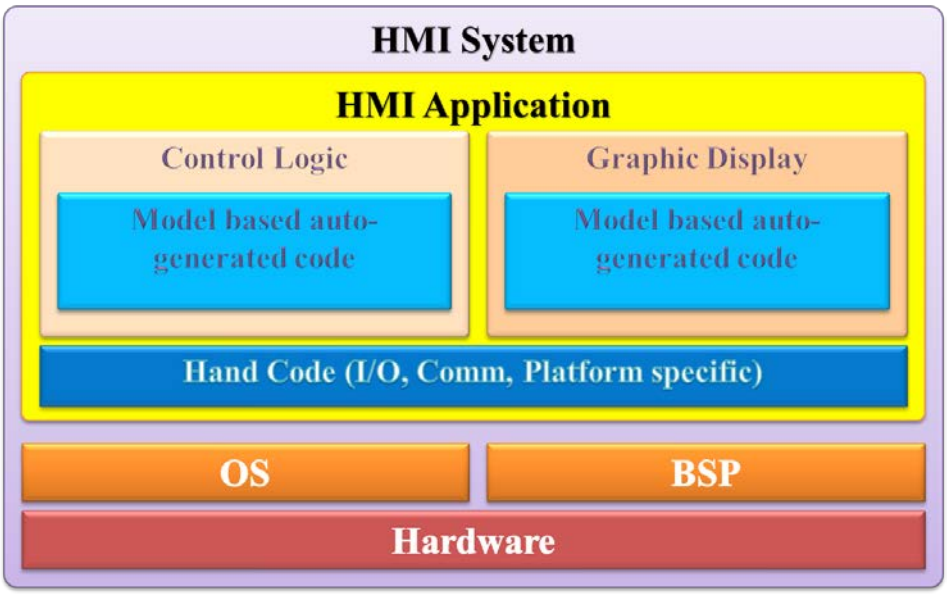

Figure 1. SCADE display application

To obtain the license from the regulatory agency, even though the source codes are qualified as safety code because they are automatically generated from safety grade tool, the source codes still need to go through the V&V activities for SIL-4 level. The verification for display application software is much more difficult compared to validation. One of the strategies to overcome this difficulty is to separate the Graphic Display of codes from Control Logic codes. This approach is simpler and more feasible because of the simple test cases by the separation of codes according to the functions.

The interfaces between Control Logic and Graphic Display are accomplished through input/output mapping between two parts. Figure 2 illustrates the mapping information concept which shows separate input/output variables for each Control Logic and Graphic Display and also mapping information for input/output of both parts.

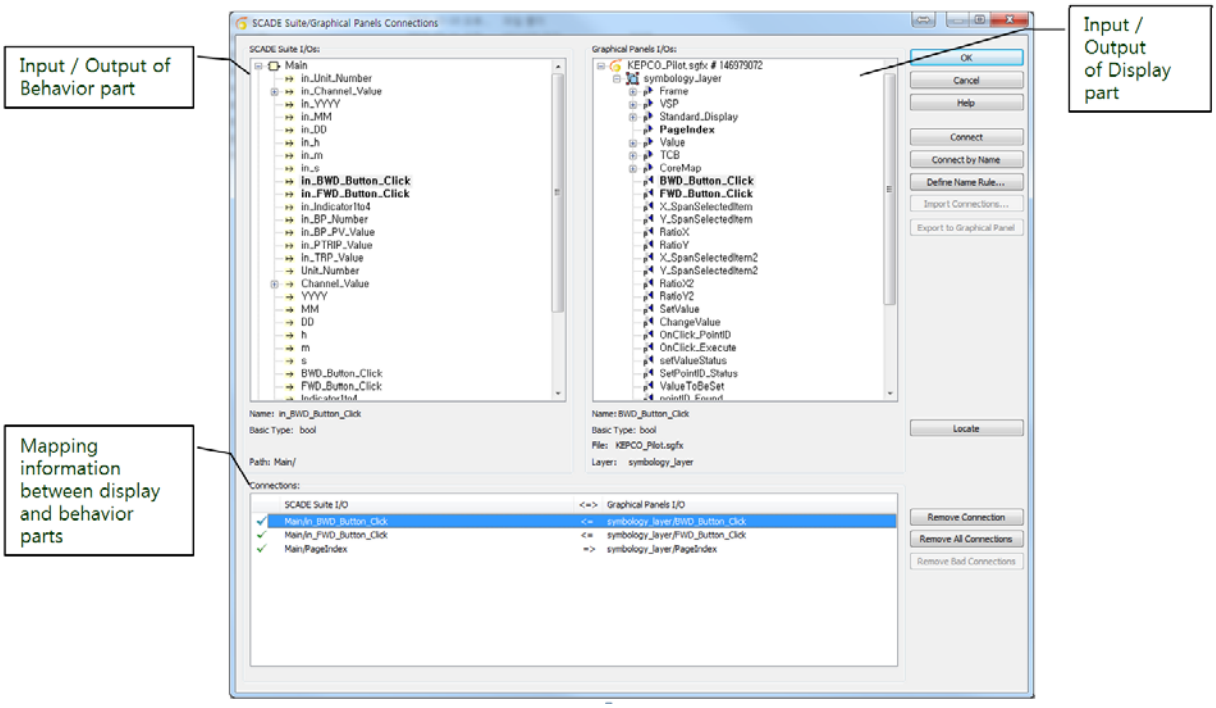

Figure 2. Mapping information between Control Logic and Graphic Display I/O

This feasibility study shows that the SIL-4 grade display with Open GL is a selectable approach, but there is one prerequisite to it – a CGID graphic card needs for SIL-4 grade nuclear display system.

3.2.2 Graphic Display without Open GL

We can consider two approaches for Human Machine Interface (HMI) display implementation without open GL. The first approach is to use a display configuration with two separated processors - one is for display server and the other is for display panel. And these work together in loosely coupled manner. The display server and display panel can communicate with each other through pre-defined graphic display protocol for display presentation. The second approach is tightly-coupled display configuration where the display server provides raw display information through Digital Visual Interface (DVI) cable to the display panel.

The tightly-coupled display configuration is to use a graphic support card (not graphic card) for graphic display. The display server does all necessary display processing and loads the display information into graphic memory map in the graphic support card. The card does not provide any graphic library as most commercial graphic cards do for Open or Open SC GL but just supports interface between display server and display panel.

This is relatively a favorable approach from the standpoint of safety certification because the display panel just receives the video data and presents it on the display panel without any data processing as shown in Figure 3.

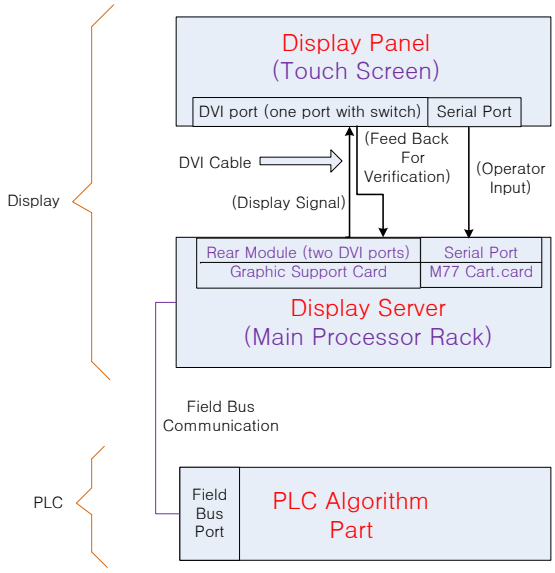

Figure 3. Tightly-coupled Display Configuration

4. Conclusion

This paper presents two methodologies for design and implementation of SIL 4 safety grade display system with the description of current status and safety requirements. One methodology is for graphic display with Open GL, and the other is for graphic display without Open GL. The feasibility study shows that both methodologies are applicable and can be selected considering the pros and cons of each approach.

The SDS with Open GL has the advantage of using commercial off the shelf design tools that provides dynamic model simulation and automatic platform independent code generation.

The SDS without Open GL has the advantage of avoiding graphic card certification and programmer friendly coding that can customize codes as is needed in V&V. The proposed configuration for SDS without Open GL is based on Power PC with compact Peripheral Component Interconnect backplane bus with tightly-coupled display configuration.

This paper is based on the feasibility study, and the future works will be the full implementation and the certification for SIL-4 display system.

5. References

[1] IEEE 1012 1998, “IEEE Standard for Software Verification and Validation”.

[2] IEC 61508 2010, “Functional safety of electrical/electronic/programmable electronic safety-related systems.

[3] Lahtinen, Jussi, et al., "Comparison between IEC 60880 and IEC 61508 for certification purposes in the nuclear domain," Computer Safety, Reliability, and Security. Springer Berlin Heidelberg, 2010. 55-67.

[4] IEEE 7-4.3.2 2010, "Standard Criteria for Digital computers in Safety Systems of Nuclear Power Generating Stations ".

[5] IEC 60880 2009, “Nuclear power plants - Instrumentation and control systems important to safety – Software aspects for computer-based systems performing category A functions”

[6] BEL V, BfS, Consejo de Seguridad Nuclear, ISTec, ONR, SSM, STUK, Common position of seven European nuclear regulators and authorized technical support organizations, Revision 2013.

[7] Baumann, Christoph, et al., "Formal verification of a microkernel used in dependable software systems," Computer Safety, Reliability, and Security. Springer Berlin Heidelberg, 2009. 187-200.

[8] Gong, ShengWen. "Modeling and Verifying the Kernel of RTOS," Information Computing and Applications. Springer Berlin Heidelberg, 2012, 610-617.

[9] Mike Medoff, Real Time Operating Systems for IEC 61508, http://www.exida.com/index.php/resources/whitepapers.

ACKNOWLEDGMENTS

This work was supported by the Korea Institute of Energy Technology Evaluation and Planning(KETEP) and the Ministry of Trade, Industry & Energy(MOTIE) of the Republic of Korea (No. 2013T100200002).