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

# Software Safety Lifecycle and Method of POSAFE-Q System

Jang-Soo Lee, Kee-Choon Kwon
*Korea Atomic Energy Research Institute, Dukjin 150, Yuseong, Daejeon, Korea, {jslee, kckwon}@kaeri.re.kr*

## 1. Introduction

This paper describes the relationship between the overall safety lifecycle and the software safety lifecycle during the development of the software based safety systems of Nuclear Power Plants. This includes the design and evaluation activities of components as well as the system. The paper also compares the safety lifecycle and planning activities defined in IEC 61508 with those in IEC 60880, IEEE 7-4.3.2, and IEEE 1228. Using the KNICS project as an example, software safety lifecycle and safety analysis methods applied to the POSAFE-Q are demonstrated. KNICS software safety lifecycle is described by comparing to the software development, testing, and safety analysis process with international standards. The safety assessment of the software for POSAFE-Q is a joint Korean German project. The assessment methods applied in the project and the experiences gained from this project are presented.

## 2. Safety Lifecycles in IEC and IEEE Standards

The safety assessment of the software for the KNICS RPS and PLC is an ongoing joint Korean German project. In the cases where the documents have been evaluated by KAERI, ISTec has checked the results of the evaluation supplemented by spot checks of the development documents according to the following IEC and IEEE standards.

- IEC 61508-1, Functional safety of electrical/electronic/programmable electronic safety-related systems –Part 1:General requirements
- IEC 61508-2, Functional safety of electrical/electronic/programmable electronic safety-related systems –Part 2: Requirements for electrical/electronic/programmable electronic safety-related systems
- IEC 61508-3, Functional safety of electrical/electronic/programmable electronic safety-related systems –Part 3: Software requirements
- IEC 60880, Nuclear Power Plants – I&C systems important to safety – Software aspects for computer-based systems performing category A functions
- IEC 61513, Nuclear Power Plants – Instrumentation and control for systems important to safety – General requirements for systems
- IEEE Std. 7-4.3.2-2003, IEEE Standard Criteria for Digital Computers in Safety Systems of Nuclear Power Generating Stations
- IEEE Std. 1228-1994, IEEE Standard for Software Safety Plan

In order to follow both frameworks of standards, IEC and IEEE, it is needed to compare the safety lifecycle, and identify the differences of the frameworks. It was compared the safety lifecycles for the general safety electronic systems in IEC 61508 and for the instrumentation and control system of a nuclear power plants in IEC 61513. We identified the differences of the safety lifecycles in IEC 60880, IEC 61513, IEEE 7-4.3.2 and IEEE 1228.

Most of the IEC and IEEE standards consist of three main phases, planning phase, realization phases according to the plan, and the validation phase. The safety lifecycles for the industry specific standards, for example, IEC 62279 for railway, IEC 61513 for nuclear power plants, inherit the definition of phases from the generic IEC standard of IEC 61508. However, the detail phases of the safety lifecycles for the specific industries are different from IEC 61508. The safety lifecycles in IEEE standards require a direct safety analysis in each phase of the lifecycle.

## 3. Software Safety Lifecycle for KNICS

In KNICS project, we developed the software safety lifecycle based on the IEEE 1228, IEEE 7-4.3.2, IEC 61513, and IEC 60880. The software safety lifecycle is tightly coupled with the reliability process. Software failures cannot be treated as random events and probabilities for software failures cannot be derived using historical data [1]. Although attempts have been made to apply a quantitative probability for software [2], this approach is still controversial. For that reason the standards being used to develop and assess safety critical software for nuclear power plants establish a software safety lifecycle and dedicated requirements to ensure safe and high reliable software. The software safety life cycle of IEC 61513 is given in fig 1.

The software development process is split into consecutive phases. Each phase produces its own set of documents. The change-over from one phase to the next one includes appropriate verification activities. After system integration the system validation shall demonstrate the system meets the system requirements

specification. The application of standards will give a solid basis for high quality software.

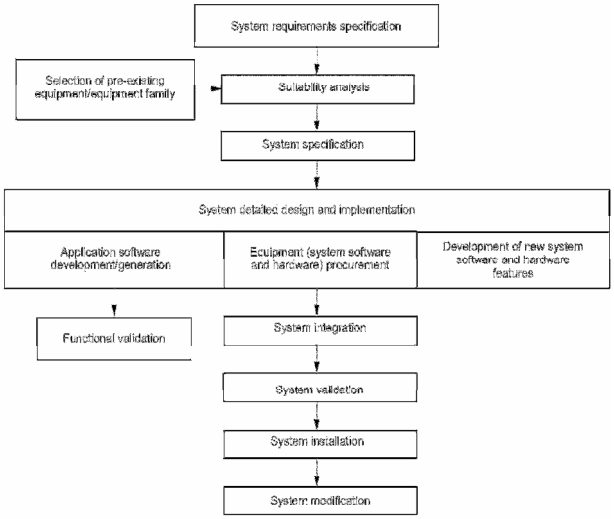


Fig. 1. Software safety life cycle (IEC 61513)

Nevertheless, software development is a complex process that may result in incorrect final products. Complete tests for all internal state conditions and input scenarios can not be performed due to time constrains. Therefore the software qualification must be complemented by safety and reliability analysis.

Several techniques for safety analysis have been used by industry for decades, and some have attracted great attention in the research community. They include Fault Tree Analysis (FTA), Failure Modes, Effects and Criticality Analysis (FMECA), Failure Propagation and Transformation Notation (FPTN), Hazard and Operability (HAZOP), and Preliminary Hazard Analysis (PHA). In Leveson's book, "Safeware" [5], there is an excellent summary on techniques for system safety and computers.

Additional failure assumptions are made. The assumed failures must be controlled by the system. Of course support mechanisms to design fault tolerant system architecture are necessary. Since no single method can prove the case of correctness of software a set of different measures gives sufficient evidence.

The bundle of different measures and activities to ensure safety and reliability comprise:

- Application of the safety life cycle, including verification and validation activities,
- Safety and reliability analysis,
- Fault tolerant system design.

In KNICS project, a safety lifecycle was developed as shown in fig. 3 with the quantitative approach of the reliability analysis for the system and hardware levels, but with the qualitative approach of the safety analysis for software.

We used the software fault tree analysis (FTA) method for the design and coding phases of the lifecycle. After creating the software fault trees using the procedure, they produced two groups of outcome from software FTA. One group is the recommendations to improve the fault tolerance, and the other is the influence on testing.

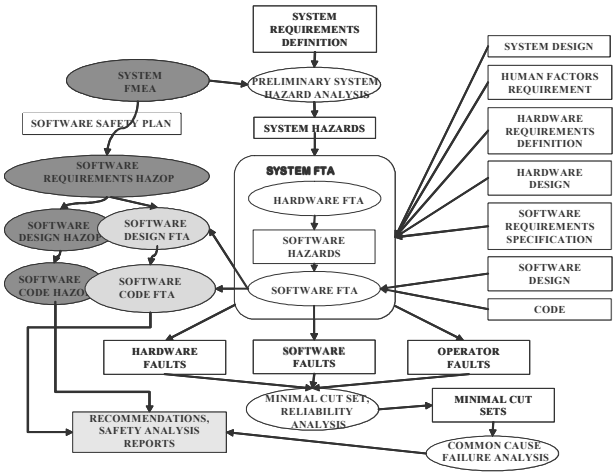


Fig. 2. Software safety lifecycle for KNICS POSAFE-Q system

## 4. Conclusions

This paper discusses the software life-cycle safety analysis tasks for the safety-critical software protection system in nuclear power plants. In order to meet the requirements from the both frameworks of standards, IEC and IEEE, the safety lifecycle have been compared, and the differences of the frameworks have been identified. The overall safety lifecycle to software safety lifecycle has been compared for developing Reactor Protection System (RPS) and its components using the KNICS PLC as an example. The differences of the software safety lifecycles of IEC 61508-3, IEC 60880, IEEE 1228-1994, and IEEE standards 7-4.3.2-2003 have been shown. The software safety lifecycle applied for KNICS RPS and PLC systems was introduced and the relationship of safety analysis and testing in software safety lifecycle was identified. Software safety assessment methods were described that have been applied for KNICS RPS and PLC systems.

## REFERENCES

[1] Leveson, N. G. and Stolzy, J. L.: Safety analysis of Ada programs using fault trees. IEEE Transactions on Reliability, Vol R-32, No-5, 1983.

[2] Ramamoorthy, C. V. Bastini, F. B.: Software reliability – Status and perspective. IEEE Transactions on Software Engineering. SE-8, 1982.

[3] Lee J. S., Kwon K. C., and Cha S. D.: Software safety analysis of digital protection system requirements using a qualitative formal method, Nuclear Technology Vol. 147, 2004.