일체형원자로 MMIS 설계에 적용하기 위한 소프트웨어 개발 계획

서용석° 장귀숙 박근옥 이종복 김동훈
한국원자력연구소 일체형원자로MMIS설계기술개발
{yssuh, gsjang, gopark, jblee, dhkim4}@kaeri.re.kr

A Software Development Plan for Integral Reactor Man-Machine Interface System Design

Yong-Suk Suh° Gwi-Sook Jang Geun-Ok Park Jong-Bok Lee Dong-Hoon Kim
SMART MMIS Design, Korea Atomic Energy Research Institute

요 약

디지틀 중심의 원자로 제어시스템 설계에서 소프트웨어 안전성이 중요한 현안으로 부각되고 있다. 컴퓨터기반의 디지틀시스템으로 설계되는 일체형원자로 MMIS에 적용하기 위한 소프트웨어 개발 계획은 이러한 현안을 만족하기 위해 개발할 필요가 있다. 본 논문은 소프트웨어 개발 계획을 소프트웨어 수명주기 설정, 정형화 기법 적용, 위해도 분석 수행, 소프트웨어 시험 방법을 중심으로 제시하였다. 본 논문에서 제시된 소프트웨어 개발 계획은 고품질의 소프트웨어 생산을 보장하며, 원자력규제기관에서 요구하는 소프트웨어 안전성 보장 계획에 대한 규제사항을 만족한다. 본 논문의 소프트웨어 개발 계획을 바탕으로 추후 구체적인 수행방법, 지침, 절차, 문서화 등이 점차적으로 개발되어 일체형원자로 MMIS 소프트웨어 개발 시 적용할 예정이다.

1. 서론

한국원자력연구소(KAERI)는 90년대 중반부터 전력생산과 담수화 목적을 위해 330MWt급의 열출력을 생산할 수 있는 일체형원자로인 SMART(System-integrated Modular Advanced Reactor)를 설계해 오고 있다. SMART MMIS(Man-Machine Interface System)는 원자로로부터 일반인에게 방사능이 누출되지 않도록 원자로를 보호, 제어, 감시하는 컴퓨터기반의 제어시스템이다[1]. 우리나라 기존의 원자력발전소 제어시스템은 아나로그기반으로 설계되었으나, 최근에는 디지틀기반으로 설계하려는 추세이다. 그러나, 원자력규제기관(KINS)은 디지틀기반의 제어시스템을 원자력발전소에 적용하는 것에 대해 아직까지 완전한 동의를 보이고 있지 않다. 그 이유중의 하나가 디지틀시스템을 동작시키는 소프트웨어가 잠재적으로 내포하고 있는 불안전성 때문이다. 소프트웨어의 고장 원인과 그에 대한 대처가 불완전하다는 것이 그들의 입장이다. 소프트웨어의 안전성이 현재까지의 소프트웨어공학 기술로는 완전히 증명될 수 없는 실정이나, 규범화된 소프트웨어 개발 프로세스 수립과 이행으로 인해 그러한 불확실성을 다소나마 해소할 수 있다는게 소프트웨어공학관점에서의 일반적인 이해이다. 따라서, 본 논문은 원자력규제기관의 요구사항을 어느 정도 만족하고, 고품질의 SMART MMIS 소프트웨어를 생산하기 위한 소프트웨어 개발 계획을 제시한다.

2. 관련연구

ISO, IEC, IEEE를 중심으로 소프트웨어 프로세스에 대한 표준 및 지침이 개발되고 있다. 또한, 산업체에서는 소프트웨어 프로세스 능력 측정 및 평가 모형으로써 CMM, SAM, Trillium, Bootstrap 등을 활용하고 있다. 국내에서는 ISO 표준인 SPICE를 채택한 KSPICE가 소프트웨어 프로세스 능력 측정 모델로써 활용되고 있다. 소프트웨어 프로세스의 대표적인 표준으로써 IEEE/EIA 12207[2]과 세부지침인 IEEE/EIA 12207.0~12207.2를 들 수 있다. 이 표준은 소프트웨어 프로세스의 기본틀을 제시하고 있으며, 무엇보다 미국 국방성(DoD)에서 그동안 군용 소프트웨어 개발에 적용해왔던 MIL Std 498을 폐기하고 대신에 IEEE/EIA 12207 적용을 권고하고 있는 것으로 그 신뢰성이 입증되고 있다[3]. 한국전력기술기준(KEPIC)의 원자력품질보증기술기준 가운데 소프트웨어의 품질보증요건도 IEEE/EIA 12207 범주 안에 든다. 한국원자력연구소에서도 소프트웨어 확인 및 검증을 위한 계획, 절차, 평가에 대한 총체적인 노력이 계속되고 있다[4]. SMART MMIS 제어시스템의 소프트웨어를 안전중요도에 따라 안전-필수, 안전-관련, 비안전으로 분류하여 각각의 확인 및 검증활동을 <표 1>과 같이 나타내었으며, 이들을 소프트웨어 수명주기 설정, 정형화 기법 적용, 위해도 분석 수행, 소프트웨어 시험 방법을 중심으로 분류하여 다음절에서 제시한다.

3. 소프트웨어 수명주기 설정

소프트웨어 개발을 위한 수명주기를 계획, 요건분석, 설계, 구현, 시험, 통합, 확인 및 검증, 설치, 운전 및 유지보수 단계로 구분한다. 계획단계에서 품질보증, 관리, 개발, 통합, 설치, 유지보수, 교육, 운전, 안전성보장, 확인 및 검증, 형상관리에 대한 계획을 수립한다. 요건분석단계에서 요건서를, 설계단계에서 설계서와 구조를, 구현단계에서 원시코드를, 통합단계에서 시스템통합문서를, 설치단계에서 설치보고서를, 운전단계에서 운전절차서를, 유지보수단계에서 유지보수절차서를 생산한다. 소프트웨어 개발 수명주기의 각 단계마다 안전성을 분석하며, 확인 및 검증 활동을 수행하며, 형상관리하에 생산물을 관리한다. 소프트웨어 개발 모델로써 IEEE/EIA 12207.2를 참조하여 폭포수(waterfall), 점진형(incremental), rapid prototyping 모델을 준용한다. SMART MMIS 설계 전반적인 흐름은 기본적으로 폭포수 모델을 선택한다. 그러나 SMART MMIS 설계가 디지틀기반으로 이루어지기 때문에 디지틀 특성에 따른 기능 및 성능을 rapid prototyping을 통하여 확인할 필요가 있다. 또한, 확인과정에서 요건의 수정 또는 개선이 발생할 수 있으므로 어느 정도 점진형의 성격도 있다고 할 수 있다. 따라서, SMART MMIS 소프트웨어 설계는 폭포수모델을 근간으로 하되 rapid prototyping과 점진형이 혼합된 hybrid 모델을 적용한다.

4. 정형화 기법 적용

정형화 기법(formal method)은 시스템 요건 및 기능을 수학적으로 표현하거나, 참과 거짓을 판명할 수 있는 논리적인 방법으로 표현하는 기법이다. 이는 명세서를 강력한 규범에 따라 표현하게 함으로써 일관된 해석을 유지하게 하고 시험을 통해 밝힐 수 없는 오류까지 명세서 검증과정에서 찾아낼 수 있도록 하기 위함이다. 정형화 기법을 적용하는데는 세 가지의 문제점이 있다. 첫 째는 기존의 프로그래머가 어떤 문제에 대해 정형적으로 생각하고 정형적 표현기법으로 명세서를 작성하는데 익숙치 않다. 따라서 개발자가 정형화 기법에 익숙하기 위해서는 많은 교육, 훈련 및 시간이 필요하다. 둘 째는 복잡한 시스템의 기능을 정형화 기법으로 모두 표현하는데 한계가 있으며 심지어 불가능할 수도 있다. 셋 째는 과거의 잘 사용해왔던 표현방법에 비해 정형적 표현방법이 효과적이다는 평가가 원자력계에서는 아직 보편화되어 있지 못하다. 현재 정형화 기법이 활발하게 사용되는 곳은 미국 NASA이며 학계와 산업계에서도 많은 정형화 기법 도구를 제시하고 있다. 앞으로 NASA의 사례[5]를 연구하여 정형화 기법의 효율과 효과에 대해 연구할 필요가 있다. 일체형원자로 안전에 치명적 영향을 주는 안전-필수 제어시스템의 명세서는 정형화 기법을 적용하여 작성하며 이에 대해 원자력규제기관에서도 긍정적으로 평가하고 있는 추세이다.

반 정형화 기법(semi-formal method)은 기능블럭도, 상태흐름도 등과 같은 도식적 표현(graphical representation)이다. CASE(Computer-Aided Software Engineering)도구는 반 정형화 도구라 할 수 있으며 원자력계에서는 반 정형화 기법에 더 익숙해 있다. 안전-관련 및 비안전 제어시스템 및 감시시스템의 명세서와 소프트웨어는 반 정형화 기법을 적용하여 작성된다.

5. 위해도 분석 수행

소프트웨어 위해도 분석(hazard analysis)은 소프트웨어 설계 시 소프트웨어에 의해 발생될 수 있는 비정상 상태와 원인을 분석하고 그에 대한 대책을 최대한 마련하기 위해 수행되는 분석행위이다. 주요 기법으로는 고장유형영향분석(failure modes and effects analysis-FMEA)과 고장수목분석(fault tree analysis-FTA)이 있다. 고장유형영향분석은 테이블형식으로 작성된다. 테이블의 항목에는 소프트웨어요소, 원인, 영향, 발견방법, 대처방안 등이 나열되며 각 항목별로 구체적인 데이터를 입력한다. 고장유형영향분석은 상향식(bottom-up)접근방식이다. 테이블의 고장요소는 최소 부품단위부터 출발한다. 현재 하드웨어 부품의 고장요소 및 원인에 대한 정보는 성숙되어 있으나 소프트웨어에 대한 고장요소를 밝히는 작업은 아직 시작단계이다. 따라서 소프트웨어 고장요소를 세분화하는 연구가 필요하다. 고장수목분석은 더 이상 하위 고장요소가 발견되지 않을 때까지 고장요소를 세분화 하는 분석행위이다. 이 방식은 AND, OR, Inhibit 게이트(gate) 심볼 등을 이용하여 고장요소를 수목형태의 모양으로 분석해 나가는 하향식(top-down)접근방식이다. 일반적으로 시스템의 고장을 분석하는 과정에서 고장영향분석과 고장수목분석을 병행하여 진행한다. 이것은 고장요소를 테이블과 수목형태로 작성하는 과정에서 시각적으로 많은 경우를 발견할 수 있기 때문이다. 소프트웨어의 고장수목분석은 소프트웨어 구문요소별(if, for, while 문 등)로 템플릿(template)을 만들어 원시코드 전체를 고장수목으로 나타내려는 시도가 있다[6]. 원자력규제기관은 안전-필수 제어시스템에 대한 하드웨어 및 소프트웨어 위해도 분석을 수행한 결과보고서 생산을 강력히 요구하고 있다.

6. 소프트웨어 시험 방법

소프트웨어 시험은 개발그룹에서 작성한 소프트웨어를 독립적인 시험그룹에서 수행하는 시험을 의미한다. 독립적인 소프트웨어 시험의 목적은 개발자가 발견하지 못한 소프트웨어 오류를 객관적으로 발견하기 위함이다. 또한, 시험을 통해 소프트웨어 고장율 데이터를 취득할 수 있으며 이 데이터는 소프트웨어 신뢰도(*reliability*) 계산에 중요한 입력으로 사용된다. 구체적인 소프트웨어 시험으로써 원시코드 정적분석과 동적시험을 수행한다. 원시코드 정적분석은 직접검토(peer review)방식이며 IBM의 Fagan이 제시한 분석방법을 수행한다[7]. 동적시험은 구조적(structured)시험, 기능적(functional)시험, 과부하(stress)시험, 통계적(statistical)시험, 회귀(regression)시험으로 구분하여 수행한다. 구조적시험은

원시코드 내부의 실행흐름을 확인하는 경로시험과 데이터의 정당성을 확인하는 데이터시험으로 구분하여 수행한다. 구조적시험은 white-box시험이며 기능적시험은 black-box시험이다. 기능적시험은 원시코드 내부구조에는 관심이 없으며 소프트웨어가 만족하여야 할 기능적 요건을 확인하는 시험으로써 외부로부터 입력 데이터를 주입하여 소프트웨어가 출력하는 결과값을 분석하는 시험이다. 과부하시험은 소프트웨어가 수행되는 환경에 과부하 조건을 만들어 소프트웨어의 성능을 확인하는 시험이다. 통계적시험은 주로 난수(random number)를 장시간 입력함으로써 소프트웨어에 오류가 발생하는지 또는 잘못된 결과를 출력하는지 관찰하는 시험이다. 회귀시험은 수정된 소프트웨어가 자신에게 수행되었던 과거의 테스트케이스(test case) 가운데 다시 시험할 필요가 있는 부분을 골라 수행하는 시험이다. 시험의 계획, 테스트케이스, 절차, 결과, 평가 등 모든 시험과 관련된 사항은 문서화 한다. 이에 대한 정보는 IEEE-829를 참조한다 [8].

7. 결론

현재 국내외적으로 아나로그 중심에서 디지틀 중심의 원자로 제어시스템이 설계되는 추세이다. 디지틀 중심의 원자로 제어시스템에서 소프트웨어 안전성이 중요한 현안으로 부각되고 있다. 본 논문은 컴퓨터기반의 디지틀 시스템으로 설계되는 일체형원자로 MMIS에 적용하기 위한 소프트웨어 개발 계획을 소프트웨어 수명주기 설정, 정형화 기법 적용, 위해도 분석 수행, 소프트웨어 시험 방법을 중심으로 기술하였다. 소프트웨어 수명주기는 IEEE/EIA 12207를 중심으로 설정하며, 정형화 기법 적용은 필요하나 방법 및 타당성 연구가 진행되어야 하며, 위해도 분석은 고장유형영향분석과 고장수목분석을 중심으로 수행하며, 소프트웨어 시험 방법은 원시코드 정적 분석과 white-box 시험 및 black-box 시험을 중심으로 수행한다. 본 논문에서 제시된 소프트웨어 개발 계획의 수립은 고품질의 소프트웨어 생산을 보장하며 이는 원자력규제기관에서 요구하는 소프트웨어의 안전성 보장을 만족할 것으로 판단된다. 본 논문을 통해 제시된 소프트웨어 개발 계획을 바탕으로 추후 구체적인 수행방법, 지침, 절차, 문서화 등이 점차적으로 개발되어 일체형원자로 MMIS 소프트웨어 개발 시 적용할 예정이다.

8. Acknowledgement

본 연구는 과학기술부의 원자력연구개발사업 일환으로 수행되었음.

9. 참고 문헌

[1]. KAERI/RR-1901/98, "일체형원자로 MMIS설계기술개발", 한국원자력연구소, 1999. 3.
[2]. IEEE/EIA 12207, "Information technology-Software life cycle processes", ISO and IEC, 1995.
[3]. MIL-STD-498, NOTICE 1, "Software Development and Documentation", USDoD, Mar. 27, 1998.
[4]. 이장수 외, "원전 계측제어 고신뢰도 소프트웨어 확인 / 검증 기술 현황", Journal of the Korean Nuclear Society, 26권, 4호, 1994. 12, pp. 600-610.
[5]. NASA-GB-002-95, "Formal Methods Specification and Verification Guidebook for Software and Computer Systems", Vol. I & II, NASA, 1995.
[6]. Nancy G. Leveson et al, "Safety Verification of ADA Programs Using Software Fault Trees", IEEE Software, Jul. 1991, pp. 48-59.
[7]. M. E. Fagan, "Design and Code Inspections to Reduce Errors in Program Development", IBM System Journal, Vol. 15, No. 3, 1976, pp. 182-211.
[8]. IEEE Std 829, "IEEE Standard for Software Test Documentation", IEEE, 1998.

<표 1> 일체형원자로 MMIS 소프트웨어 등급별 요구되는 확인 및 검증 활동

확인 및 검증 활동	소프트웨어 등급분류		
	안전-필수	안전-관련	비안전
개념문서 평가	○	○	○
하드웨어, 소프트웨어 및 사용자 요구사항 할당분석	○	△	-
소프트웨어 확인 및 검증계획 수립 및 이행	○	○	○
소프트웨어 요구사항 검토 및 분석	○	○	○
정형적 요건 및 설계방법의 사용	△	-	-
정형적 요건 및 설계검토	△	-	-
소프트웨어 설계 분석 및 평가	○	○	△
정형적 언어의 사용	△	-	-
정적 원시코드 분석	○	△	△
요건 추적 분석	○	△	△
필수성 분석	○	△	-
위험요소 및 위해도 분석	○	△	-
연계사항 분석	○	△	-
고장유형영향 분석	○	△	-
알고리즘 분석	△	-	-
데이터베이스 분석	○	△	-
복잡도 및 타이밍 분석	○	△	△
제어 및 데이터 흐름 분석	○	△	△
시스템 확인 및 검증 시험계획 수립	○	○	△
구조적 시험	○	○	△
기능적 시험	○	○	○
통계적 시험	△	-	-
부하(스트레스) 시험	○	△	△
회귀 시험	○	○	△
검증 시험	○	○	△

주) ○ : 반드시 수행, △ : 수행을 권고, - : 수행 안함