# 원자력발전소 안전-필수 소프트웨어 개발 방법론

한국전력기술 장훈선 · 장영우 · 하재홍 · 김재학

## 1. 서 론

원자력발전소는 안전성의 확보가 가장 중요한 관심사이기 때문에 다양하고 다중의 보호개념으로 설계되고 있으며 고장시의 안전조건(Fail Safe)을 고려하여 충분한 설계여유를 갖도록 설계된다. 이에 따라 신뢰도를 확보하기 위하여 기존의 원전에 적용되어 사용되고 있는 설비 및 설계기술을 가급적 유지하려는 설계방식이 채택되어 왔다. 그러나, 계측제어분야는 최근 전기, 전자 및 컴퓨터 산업의 급속한 발달로 아날로그 기기 및 릴레이의 생산 감소 및 중단에 따라 기존 원전의 노후화된 계측제어 기기의 계측제어설비가 디지털 설비로 교체되고 있으며, 신규 원자력발전소의 계측제어계통 설계 시에도 최신 디지털 기술의 적용이 증대되고 있다.

계측제어계통에 디지털 기기를 적용할 경우에 기대되는 효과는 예비부품의 확보 및 부품 단종의 문제점 해결, 기기 노후화로 인한 계측기의 드리프트 제거뿐만 아니라 자기진단 및 자동시험의 구현으로 보수 및 정기시험에 소요되는 인력감소 및 작업시간의 단축 등의 장점이 있다. 그러나 대부분의 제어 및 보호기능이 하드웨어에 내장된 소프트웨어로 구현되므로 계측제어계통 기기의 품질 및 신뢰도를 확보하고, 기능을 보장하기 위한 높은 신뢰도의 소프트웨어 개발이 중요하다.

디지털 발전소안전계통은 4 개의 다중채널로 구성된다. 이러한 4 개의 채널방식의 보호계통은 하나의 기기가 의도된 안전기능 수행능력을 상실하는 단일고장이 발생하더라도 계통이나 기기에 대한 최소 다중성을 확보한다. 이 설계원칙을 단일고장 기준(Single Failure)이라고 하며 이러한 설계적용으로 발전소 이용율을 향상시킨다[1]. 보수나 시험 또는 한 채널의 감지기가 고장이 발생할 경우에도 정상운전이 가능하도록 우회될 수 있다. 트립채널 우회는 운전원이 보수시험반 및 운전원 모듈을 이용하여 요구할 수 있으며, 이 경우 트립 논리는 2/4 동시논리에서 2/3 동시논리로 변환된다.

또한 다중의 안전기능을 일시에 불능상태로 만드는 소프트웨어 공통유형고장(Common Mode Failure)을 배제하기 위한 노력도 기울여야 한다. 아울러 안전기능의 손실이 없도록 하기위해서 다양성(Diversity)을 갖는 심층방어(Defense-in-Depth) 개념을 가지고 설계되어야 한다. 심층방어 개념은 어떤 주어진 기능을 달성하기 위해 대체 또는 보조 수단을 설치하거나 또는 동일한 목표를 달성하기 위하여 다른 기능을 마련하거나 조치를 강구해서 의도된 기능을 달성하게 하는 설계 원칙이다.

따라서 안전계통에 적용하는 안전-필수 소프트웨어의 개발은 설계, 확인 및 검증, 전자파 장해에 대비한 설계 등 원자력발전소의 안전성 및 신뢰도에 영향을 주지 않도록 수행되어야 한다.

## 2. 본 론

본 논문에서는 디지털 안전계통에 안전-필수 소프트웨어를 적용함에 하드웨어 사항 및 소프트웨어 측면을 고려한 특성, 소프트웨어 개발체계, 규제지침 및 기술기준에 따른 문서작성, 시험 체계 등에 대해 기술하고자 한다.

### 2.1 소프트웨어 개발체계

소프트웨어 개발은 규제지침에서 제시하는 폭포수(Waterfall)모델 방법을 사용하고 있다[2].

그림 1에서 보여주는 바와 같이 소프트웨어 개발은 소프트웨어 수명주기에 따라 다음과 같이 단계별로 개발된다.

1) 개념 단계
2) 요건 단계
3) 설계 단계
4) 구현 단계
5) 시험 단계
6) 현장 설치 단계
7) 운전 및 보수 단계

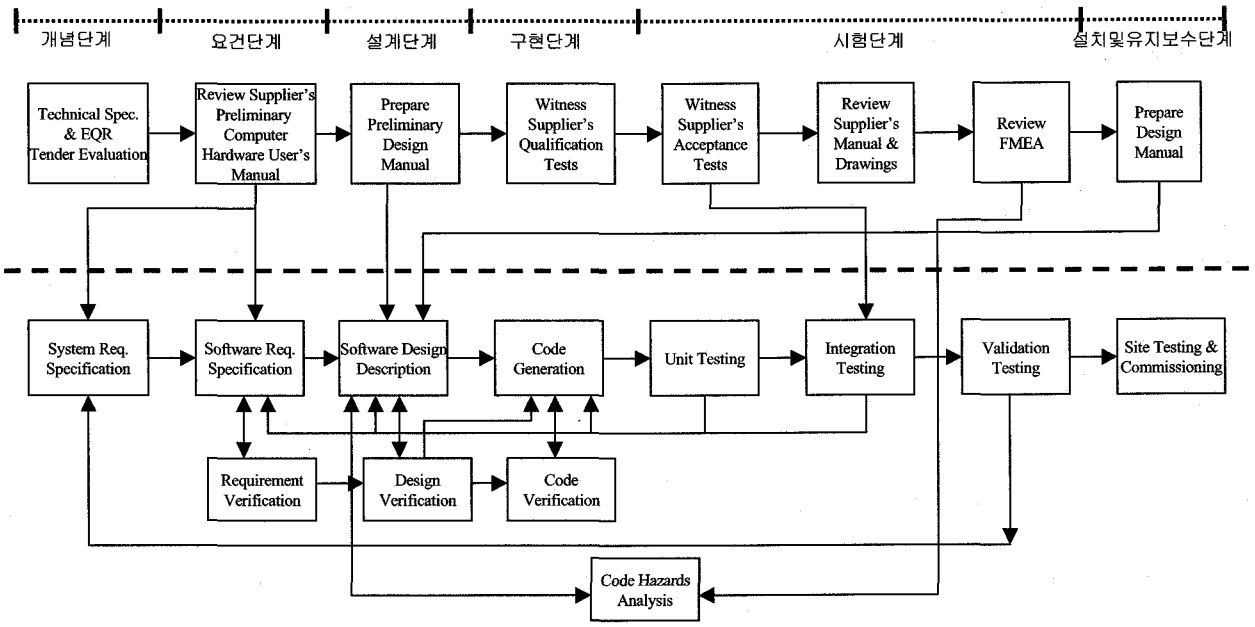



그림 1 소프트웨어 개발 수명주기(Typical)

## 2.2 소프트웨어 단계별 주요 업무

소프트웨어 설계의 각 단계는 업무에 대한 규제 요건과 승인된 기술기준에 따른 지침을 만족하도록 요구되고 있다. 이에 따른 중요한 수행업무 및 결과물은 다음과 같다.

### 2.2.1 개념 단계

개념단계에서는 소프트웨어 관리계획, 품질보증 등의 계획서들이 결과물로 생산된다.

#### 2.2.1.1 소프트웨어 관리계획서

소프트웨어 개발에 있어서의 관리적인 측면을 기술하며 전반적인 소프트웨어의 관리계획과 프로젝트 엔지니어링 계획의 일부를 기술한다.

#### 2.2.1.2 소프트웨어 개발계획서

프로젝트 개발에서의 기술적인 부분을 담당한다. 또한 각 수명주기 단계에서의 목표와 전체적인 프로젝트의 목표를 포함하여 기술한다.

#### 2.2.1.3 소프트웨어 품질보증계획서

소프트웨어의 개발 및 사용과 관련하여 그 과정과 실행부분을 기술하며, 소프트웨어의 등급분류, 소프트웨어 개발과정, 소프트웨어 관리, 문서화 규격, 관용적 절차, 검토요건, 문제점 보고 등을 포함하는 소프트웨어 품질과 관련된 내용들을 포함한다.

#### 2.2.1.4 소프트웨어 형상관리 계획서

소프트웨어 생산물, 소프트웨어 변경사항의 관리 및 구현, 그리고 변경사항 구현상태의 기록과 보고에 관한 내용을 정의하여 기술한다. 또한 형상관리 대상 항목의 관리, 상태보고 및 검토에 관한 내용을 포함한다.

#### 2.2.1.5 소프트웨어 확인 및 검증계획서[2]

소프트웨어의 전 수명주기를 통하여 소프트웨어 계통의 요구사항들이 최종 소프트웨어 산출물에 정확히 구현되었는지를 평가하도록 하는 문서이다. 각각의 안전성 관련 소프트웨어 모듈들은 개발팀과는 독립적인 조직의 검증팀에 의해서 확인 및 검증을 받아야 한다. 소프트웨어 확인 및 검증계획은 확인 및 검증 업무, 단계별의 입출력, 요구사항 추적관리표 및 체크리스트, 수명주기내의 업무분장과 책임사항이 기술되어야 한다.

#### 2.2.1.6 소프트웨어 안전성계획서[3]

공중의 건강과 안전에 중대한 결과를 초래하는 소프트웨어 결함을 예방하기 위하여 필수안전 소프트웨어에 대하여 수행되는 업무를 기술하고 있다. 또한 조직, 책임, 요원자격, 그리고 훈련문서 요건 등을 기술하고 있다. 수명주기별 안전성 분석 업무를 기술하고 있다.

#### 2.2.1.7 소프트웨어 통합계획서

소프트웨어의 통합과정 및 하드웨어/소프트웨어 통합과정의 전반적인 내용과 그러한 과정들의 목적을 기술하고 있다.

#### 2.2.1.8 코딩 지침서[4]

소프트웨어 코딩에 필요한 표준절차 및 형식을 기술

하고 있다. 이러한 목적은 정해진 구조 및 절차에 의한 가독성, 관리유지 효율성 및 각 단계별로 연계되는 신호들의 일관성을 유지하기 위한 목적으로 하고 있다.

#### 2.2.2 요건 단계

소프트웨어 개발의 가장 중요한 소프트웨어 요건의 이해와 문서화가 이루어진다. 이 단계의 중요문서는 다음과 같은 문서로 나누어진다.

- 소프트웨어 요건 명세서
- 확인 및 검증 요건 요약보고서

#### 2.2.3 설계 단계

소프트웨어 개발의 소프트웨어 요건을 바탕으로 소프트웨어를 설계하는 것이다. 이 단계에서 필요로 하는 업무는 다음과 같다.

- 하드웨어 및 소프트웨어 구조 설계
- 소프트웨어 설계 명세서
- 확인 및 검증 설계 요약보고서

#### 2.2.4 구현 단계

소프트웨어 설계가 실제적인 코드로 변환되는 것을 의미한다. 이 단계에서는 소스코드를 컴파일, 링크작업을 통하여 실행 화일을 생산하는 것이다. 이 구현 단계의 중요업무는 다음과 같은 문서로 나누어진다.

- 소스코드 리스팅
- 실행화일 생산
- 단위시험 수행
- 확인 및 검증 구현 요약보고서

#### 2.2.5 시험 단계

이 단계에서는 하드웨어와 소프트웨어가 통합된 환경에서 실시된다. 통합 시험절차서는 기능 요건서에 기록된 모든 요건이 빠짐없이 시험되는가를 확인하는 것이다. 이 단계에서의 중요 업무는 다음과 같은 문서로 나누어진다.

- 통합시험 절차서
- 통합시험 보고서
- 확인 및 검증 시험 요약보고서
- 코드 인증서 발행

#### 2.2.6 현장설치 단계

이 단계에서는 실제로 현장 환경에 맞게 기기가 설치되는 작업을 수행한다. 이 단계에서의 중요 업무는 다음과 같은 문서로 나누어진다.

- 현장시험 절차서(SATP)
- 현장시험 보고서(SATR)
- 최종 확인 및 검증 보고서
- 소프트웨어 매뉴얼

### 2.3 소프트웨어 특성

안전-필수 소프트웨어의 설계는 소프트웨어 공학의 원리에 따라 모듈화, 단순화, 서브루틴의 최소화 사용, 인터럽트의 배제 등을 기본으로 하며 설치 후의 보수를 용이하도록 하여야 한다.

소프트웨어의 품질특성은 기능특성과 개발공정 특성으로 나누어질 수 있다. 기능 특성은 안전계통 소프트웨어가 가져야할 특성이고, 개발 수행 특성은 소프트웨어가 요구된 동작들을 수행을 보증하는데 기여하는 소프트웨어 개발공정에 관한 특성이다. 다음은 안전-필수 소프트웨어를 구현하는데 필요한 요건들을 제시하였다.

#### 2.3.1 기능특성

- 정확도

센서와 운전원의 입력에서 오류가 발생하지 않는 정도이며, 정확도의 요건은 각 입력과 출력변수에 대한 데이터 형태와 데이터 크기에 대한 요건을 포함해야 한다.

- 기능성

각 운전모드에서 수행되어야 할 동작들이 완전하게 명시되어야 한다. 기능에 대한 서술은 해당 기능의 입력, 기능에 의해 실행되는 변환, 기능에 의해서 발생되는 출력 등을 포함해야 한다.

- 신뢰도

고장-내성(Fault-Tolerance)과 고장모드들에 대한 요건들이 각 운전 모드에 대해서 명백하게 기술되어야 한다. 하드웨어 및 소프트웨어 요건이 컴퓨터 시스템의 고장분석과 고장으로부터의 복구에 관한 요건과 함께 제시되어야 한다. 온라인 가동 중 시험과 진단기능에 대한 요건들이 제시되어야 한다.

- 강인성

부정확하고, 모호하고, 부적당한 입력, 하드웨어 이상한 현상 및 소프트웨어 현상이 있을 때 소프트웨어가 정확하게 동작하도록 설계되어야 한다. 특히 그러한 상황에서도 소프트웨어는 건전성을 유지하며 부정확한 출력을 만들지 않아야 한다.

- 안전성

시스템의 안전성에 직접 영향을 미치거나 이와 상호연관되는 소프트웨어 시스템의 특성을 말한다. 안전성 기능들은 잘 정의되고 엄격히 통제된 인터페이스를 유지해야 한다.

- 보안

불필요한 무단침투나 안전에 영향을 주는 침투를 방

지하기 위한 능력을 말하며 안전관련 기능에 영향을 주는 보안위협 사항들을 감지, 예방하기 위해 소프트웨어가 취해야 할 조치사항이 고려되어야 한다.

• 타이밍

정해진 타이밍 제약조건 내에서 동작해야 하는 기능들이 확인되어야 하며, 각 기능에 대한 타이밍 기준이 정해져야 한다. 타이밍 요건은 안전성 동작에 대한 입력과 출력간의 시간지연이 정상 및 예상된 고장 조건들 하에서 결정론적으로 서술되어야 한다.

### 2.3.2 개발 프로세스 특성

• 완전성

컴퓨터시스템의 모든 운전모드와 모든 변수들의 가능한 값에 대해 컴퓨터시스템이 요구하는 모든 작업들을 완전하게 기술해야 한다. 소프트웨어가 감시 및 제어해야 할 기능 요건들이 완전하게 기술되어야 한다.

• 일관성

하나의 소프트웨어의 요건명세서의 내용들이 설계, 구현 테스트단계까지 관련된 여러 가지 종류의 문서들과 요소들 간에 서로 상반된 것이 없는 것을 말한다.

• 정확성

컴퓨터시스템에서 요구되는 동작들을 기술할 때에 결함이 없어야 하고, 다른 어떤 요건들이 서술되어서는 안된다. 요구되는 기능들의 개시방법, 요구되는 입력/출력, 그 기능에 요구되는 업무순서, 동작, 사건 및 종결조건과 시스템의 상태 등을 정확하게 기술해야 한다.

• 스타일

계획문서, 수행문서, 설계결과물 등과 같은 문서의 체제, 형태 및 구조를 말한다. 또한 이해성, 판독성, 수정성 등의 효과를 증진시키기 위하여 프로그래밍의 제약사항 및 코딩관례에 관한 지침을 제시한다.

• 추적성

수명주기 단계의 설계요소들과 코드요소들 간의 양방향 추적이 가능하도록 해야 한다. 아울러 그 요소가 정확하게 구현되었는지를 확인하기 위해 요소들이 후속 또는 선행단계까지 추적될 수 있는 정도를 말한다.

• 명확성

소프트웨어의 요건 및 요소나, 서로 합쳐진 모든 요소들이 한가지의 의미만을 가지는 것을 말한다.

• 확인성

각 요건이 만족되었는지를 확인하기 위한 특정한 분석, 검토 또는 시험이 가능하도록 해야 한다.

### 2.3.3 소프트웨어 설계요건

안전-필수 소프트웨어에서 구현되어야 할 기능은 비안전계통과 달리 여러 가지 설계요건을 고려하여야 한다. 소프트웨어 구현에 고려하여야 할 중요한 요건은 다음과 같다.

#### 2.3.3.1 운영체제의 건전성

실행 소프트웨어는 계속적 및 자동적으로 CPU, 메모리, 입출력카드, 및 통신 등을 진단을 수행한다. 만약 멈춤, 명령어 오류, 메모리 오류 같은 치명적인 에러가 CPU에 의해서 탐지되면 소프트웨어의 오류가 운전원에게 통지되어야 하며 재가동 없이 정지되어야 한다.

#### 2.3.3.2 메모리 건전성

메모리의 건전성을 점검하기 위하여 일정한 시간마다 순환잉여검사(Cyclic Redundancy Checking) 또는 메모리의 XOR 기법을 통하여 메모리의 건전성을 점검한다.

#### 2.3.3.3 소프트웨어 흐름의 건전성

응용 소프트웨어가 모든 모듈을 완전하게 수행하는지를 알기 위하여 일정한 시간 내에 완료를 알리는 신호를 점검한다.

#### 2.3.3.4 시스템 건전성

시스템 프로세서의 동작이 정지되면 감시타이머(Watchdog)로 가는 박동신호(Heartbeat)의 출력이 차단되며 감시타이머(Watchdog timer)는 디지털 출력 모듈의 전원을 차단하여 작동신호가 발생하도록 설계한다.

#### 2.3.3.5 하드웨어 건전성

아날로그 입력신호 카드 및 디지털 입력신호 카드의 건전성을 감시하는 소프트웨어를 구현한다.

#### 2.3.3.6 시스템 프로세서 부하(load) 제한

프로세서가 이용하는 메모리, 입.출력 등의 자원의 이용율이 일정량을 초과하지 않도록 한다.

#### 2.3.3.7 수명주기 동안의 변수 명 일치유지

수명주기 동안에 사용되는 변수는 확인 및 검증업무를 효율적으로 수행하기 위하여 변수명의 형식, 크기 등을 가능한 한 일치되도록 유지하여야 한다.

### 2.3.4 상용제품 인증(Commercial Grade Item Dedication)

상용등급으로 사용되는 운영체계 및 응용프로그래밍 작성 도구 소프트웨어를 안전계통에 사용하기 위해서는 상용제품 인증절차를 통해 검증되어야 한다. EPRI

TR-106439에서는 4개의 방법을 권고하고 있다[5].

- Special Tests and Inspections
- Commercial Grade Survey
- Source Verification
- Acceptable Supplier/Item Performance Record

이중 방법 4는 다양한 산업 환경에서 얻어진 운전이력 및 성능데이터를 바탕으로 사용될 제품의 성능을 신뢰할 수 있도록 해주는 상용인증절차로서 운전 이력 자료는 제품의 특성 및 사용목적에 부합되는 산업 환경에서 수집된 성능데이터를 바탕으로 작성되어야 한다. 또한 제작자는 설계 및 공정관리에 대한 조치를 적절히 수행하고 실사를 통해 확인받아야 한다.

### 2.3.5 소프트웨어 예비 위험도 분석

안전계통에 대해서 수행되는 소프트웨어 예비 위험도 분석은 소프트웨어 수명주기 초기단계에서 소프트웨어 실패로 영향을 받을 가능성이 있는 위험요소를 작성한다. 위험요소는 참고문헌 6에 언급한 바와 같이 기능특성을 기본으로 하여 따라 표 1과 같이 분류하여 제시되고 있다[6]. 예비 안전성 분석보고서와 고장모드 및 영향분석(Failure Mode and Effect Analysis:FMEA)을 기반으로 각 위험요소를 지침구문으로 정하여 소프트웨어 실패원인, 탐지방법, 잠재적 실패결과, 위험도 영향, 실패경감방법 및 위험도 제어검증방법을 작성한다. 표 2는 소프트웨어 예비 위험도 분석표를 보여주고 있다.

표 1 기능특성에 따른 위험요소 목록(예)

| 특 성 | 형 상 | 적용단계 | 위험요소[지침구분] |
|---|---|---|---|
| 정확도 | 센 서 | RADC | 범위 안에 있으나 잘못됨① |
| | 작동기 | RADC | 아무 곳에서 Stuck |
| 용 량 | 타이밍 | RADC | 입력신호가 늦게 발생 |
| | 메시지 | RADC | 전송율이 비규칙적② |
| 기능성 | | RA | 기능이 명시된 대로 수행안함③ |
| 신뢰도 | | RA | 오류가 예상되지 않은 곳으로 파급 |
| 강인성 | | RA | 필요시 재가동에 실패④ |
| 안전성 | | RA | 위험상태에서 안전 상태로 이동하는데 실패 |
| 보 안 | | RA | 불인정 소프트웨어 변경 발생 |

표 2 소프트웨어 예비 위험도 분석표(Typical)

| 실 패 형 태 (지침구문) | 실 패 원 인 | 탐 지 방 법 | 잠재적 실패 결과 | 잠재적 위험영향 | 경감실패 | 위험 제어확인 방법 |
|---|---|---|---|---|---|---|
| ① | 입력에러 변환에러 | 시각점검 채널비교 | 채널 알람 | 계통 트립영향 | 4채널방지 | 소프트웨어 테스팅, 코드검토 |
| ② | 입력에러 프로그래밍 에러 | 시각점검 채널비교 | 부정확값 사용됨 | 계통 트립영향 | 4채널방지 | 소프트웨어 테스팅, 코드검토 |
| ③ | 입력에러 프로그래밍 에러 | 시각점검 채널비교 | 트립 데이터 전달 안됨 | 계통 트립영향 | 4채널방지 워치독 작동 | 소프트웨어 테스팅, 에러보고 |
| ④ | 부적절 요건 부적절 설계 부정확 검토 | 채널경보 에러보고 트립 | 채널경보 트립 | 없음 | 표준 소프트웨어 개발 절차 확인 | 설계검토 테스팅, 에러추적 |

### 2.3.6 소프트웨어 등급분류

원자력발전소에 적용되는 소프트웨어는 중요도에 따라 4개의 등급으로 분류될 수 있으며 특히 안전기능을 직접적으로 수행하는 안전-필수(Safety Critical)등급으로 분류되어 가장 수준 높고 엄격한 확인 및 검증과정이 적용된다. 표 3은 소프트웨어 등급 분류를 보여주고 있다.

표 3 소프트웨어 등급분류(IEEE Std. 1012: Typical)

| Criticality | Description | Integrity Level |
|---|---|---|
| 안전-필수 등급 (Safety Critical) | 소프트웨어의 기능이 원자로 보호계통의 개시작동, 공학적 안전설비 제어작동, 안전정지 제어작동을 직접적으로 수행하기 위하여 필요한 것. | 4 |
| 안전중요 등급 (Important to Safety) | 소프트웨어의 기능이 보조 보호계통의 제어 작동에 필요하거나, 또는 감시 및 시험을 위해 요구되는 소프트웨어, 또는 발전소의 중요 안전기능을 감시하기 위한 소프트웨어 | 3 |
| 이용중요 등급 (Important to Availability) | 가동 중인 원전을 유지하는데 필수적인 발전소 시스템 및 장비의 유지 보수를 위해 요구되는 소프트웨어 | 2 |
| 일반 등급 (General Purpose) | 상기 분류 이외의 다른 목적을 수행하기 위한 소프트웨어. 본 등급으로 분류된 소프트웨어는 상기 분류의 소프트웨어를 개발하는데 필요한 개발도구를 포함한다. | 1 |

### 2.3.7 관련 규제지침 및 기술기준

안전계통의 기능을 위해 소프트웨어를 사용할 경우 이를 위한 규제지침은 그 소프트웨어가 적합한 소프트웨어 개발 계획에 따라 개발되었음을 확증하고, 그 계획이 관련 규제지침 및 기술기준에 의한 소프트웨어 수명주기를 따랐다는 증거가 있으며, 그 프로세스가 적합한 절차에 따라 설계결과물을 생산했다는 증거를 제시하는 것을 기본으로 하고 있다.

이와 같은 절차는 10 CFR 50, 부록 B의 요건에 따라 설계의 적합성을 확인 및 검증하기 위한 품질 표준들이 지정되고 설계관리 수단들이 마련되어야 함을 요구하고 있다. 또한 규제지침 1.173의 지침에 따른 소프트웨어 공학 수명주기에 따라 수행할 것을 요구하고 있다.

이에 따라 규제기관은 안전-필수 소프트웨어를 개발하는 수명주기 동안 규제지침에서 요구하는 절차대로 모든 문서들이 진행되는지에 대하여 감사를 수행하고, 질의를 통하여 지침준수 여부를 확인하며 개발기관은 질의에 대한 답변서를 제출하고 있다.

표 4는 안전계통 디지털 소프트웨어에 대한 규제지침과 승인된 기술기준을 보여주고 있다.

표 4 소프트웨어 관련 규제지침 및 기술기준

| 원자력발전소 규제지침 | 승인된(Endorsed) 기술기준 |
|---|---|
| R.G. 1.152: Criteria for Digital Computer | IEEE 7-4.3.2 : Standard Criteria for Digital Computer |
| R.G. 1.153: Criteria for Safety Systems | IEEE 603 : Criteria for Safety Systems |
| R.G. 1.1.68: Verification, Validation, Review, and Audits | IEEE 1012 : Software Verification and Validation |
| | IEEE 1028 : Software Reviews and Audits |
| R.G. 1.169: Configuration Management Plans | IEEE 828 : Software Configuration Plans |
| | IEEE 1042 : Software Configuration Management |
| R.G. 1.170: Software Test Documentation | IEEE 829 : Software Test Documentation |
| R.G. 1.171: Software Unit Testing | IEEE 1008 : Software Unit Testing |
| R.G. 1.172: Software Requirement Specification | IEEE 830 : Software Requirement Specification |
| R.G. 1.173: Developing Software Life Cycle Process | IEEE 1074 : Developing Software Life Cycle Processes |

## 2.4 확인 및 검증

안전 필수 소프트웨어 확인 및 검증은 크게 두 부분으로 나눌 수 있다. 첫 번째는 개념단계부터 설계단계까지 작성된 설계문서의 확인 및 검증 업무를 포함한다. 두 번째는 일상적으로 언급하고 있는 시험단계의 테스트를 의미한다.

### 2.4.1 설계문서 확인

계통이 요구하는 소프트웨어의 요건이 완전성, 일치성, 정확성, 및 추적성 등을 평가하고, 설계가 기술기준 지침이나, 설계 절차서에서 정의한대로 구현되었는가를 평가한다. 이러한 업무를 수행하기 위하여 각 단계마다 소프트웨어 확인 및 검증 요약보고서를 작성한다. 이 보고서의 주된 내용은 다음과 같은 형식을 가지고 있다.

- 수행된 확인 및 검증 업무
- 미완료된 업무의 기록
- 불일치 업무 및 해결된 업무의 기록
- 소프트웨어 품질 평가

각 단계마다 수행되는 확인 및 검증 요약 보고서에는 요건추적 매트릭스(Requirement Traceability Matrix) 도표 및 단계별 검검표(Checklist)를 작성하여야 한다.

점검표는 단계별로 점검되어야 할 항목을 작성하여 점검하는 형식으로 진행한다.

요건추적 절차는 시스템 기능 요건명세서의 요구사항을 추적관리를 하는 표준화된 절차를 수행한다. 추적절차는 요건단계부터 시험단계까지 각 단계마다 관련된 장,절을 입력하여 요구사항의 일치성을 추적할 수 있으며 각 단계마다 검토자의 평가의견이 반영된다. 표 5는 요건추적 매트릭스 도표를 보여주고 있다.

표 5 Requirements Traceability Matrix

| 번 호 | Req. Fields | Requ./ Definition | Design | Imple/ Integration | Test/Validation |
|---|---|---|---|---|---|
| 1. | 1.xxxxxxxxxxx xxxxxxxxx<br>검토자 의견 | 해당문서의 장,절 | 해당문서의 장,절 | 해당문서의 장,절 | 해당문서의 장,절 |
| 2. | 2.xxxxxxxxxxx xxxxxxxxx<br>검토자 의견 | 해당문서의 장,절 | 해당문서의 장,절 | 해당문서의 장,절 | 해당문서의 장,절 |

### 2.4.2 테스트

안전-필수 소프트웨어 테스트는 크게 다음과 같이 분류되어진다.:

- 모듈 테스트
- 계통통합 테스트
- 계통검증 테스트
- 공장인수 테스트(FAT)
- 현장인수 테스트(SAT)

모듈테스트는 소프트웨어 모듈레벨의 평가를 수행한다. 계통통합 테스트는 모듈 통합의 특성을 조사하고 모듈간의 연계를 목표로 하고 있다. 계통검증 테스트는 계통의 기능 요건을 완벽하게 만족하는지를 테스트한다.

공장인수 테스트는 기기의 적정한 와이어링 및 기기 모듈을 확인한다. 이를 위하여 입력/출력단자의 확인, 입력카드의 조정, 모든 릴레이의 정상작동 및 기기 간의 데이터 링크 등을 점검한다.

신뢰도, 효율적 및 경제적인 테스트를 수행하기 위하여 상용 테스트 도구를 사용하는 방향으로 발전하고 있다. 그림 2는 상용 테스트 도구인 LDRA를 이용한 모듈 테스트의 수행방법을 보여주고 있다. 이 테스트 도구를 이용하여 각 모듈의 모든 실행문(Statement)과 분기문(Branch)의 커버리지(Coverage) 결과를 알 수 있다[5].

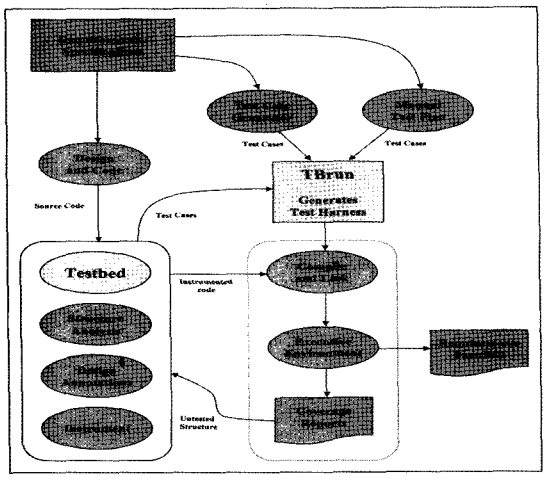



그림 2 테스트 도구(LDRA) 모듈 시험 절차도

계통통합 테스트에서는 인간기계연계(Man Machine Interface)와 응용 소프트웨어의 통합을 중점으로 프로세서, I/O 모듈, 통신 모듈 등에 테스트를 수행한다.

계통검증 테스트에서는 현장의 환경을 모사한 I/O 시뮬레이터를 이용하여 수행한다. 현장인수 테스트는 현장의 환경에 적용하기 위한 테스트를 수행한다. 이를 위하여 운전모드에 따른 테스트 및 운영 절차서를 적용한 테스트를 수행한다.

#### 2.4.3 원자력발전소 확인 및 검증과 산업계 CMMI

기존에는 원자력발전소와 산업계는 소프트웨어 개발에 각각 독립적인 지침과 기술기준을 기본으로 소프트웨어 수명주기 관리를 왔으나, 원자력 규제지침이 인정하는 IEEE Std.1012에 CMMI(Capability Maturity Model Integration) 공정과의 맵핑을 제공하고 있다. 이 도표에의해서 제공하는 CMMI 핵심영역과 상호 관련되는 중요 항목은 다음과 같다.

- 공정 관리(Process Management)
- 사업 관리(Project Management)
- 위험 관리(Risk Management)
- 요건 관리(Requirement Management)
- 기술적 해결(Technical Solution)
- 형상 관리(Configuration Management)
- 문제점 해결(Decision Analysis & Resolution)

상기에서 보는 바와 같이 대부분의 항목이 원자력 규제지침에 잘 대응하는 것을 보여주고 있다.

## 3. 결론 및 향후 전망

원자력발전소 안전-필수 소프트웨어 개발은 경제성보다는 안전성에 중점을 두고 있으며 이에 따라 확인 및 검증 업무를 수행하기 위하여 독립된 확인 및 검증 조직을 두고 있다. 규제지침과 기술기준에 따라 각 단계마다 소프트웨어 개발과 병행하여 엄격한 확인 및 검증 업무를 수행하고 있다. 향후에 디지털 계측제어 계통을 네트워크를 이용하여 정보를 송·수신 함으로 발생될 수 있는 보안위협으로부터 계측제어 계통을 보호하기 위한 절차적, 기술적 설계방안이 제고되어야 한다.

규제지침서의 확인 및 검증 업무와 산업계의 소프트웨어 개발 프로세스 분야인 CMMI와의 맵핑을 제공하고 있으며 이에 따라 프로세스 분야의 발전에 기여를 할 것으로 기대된다[2].

원전의 제어실에 제공되는 시각정보에 대한 표시방식 변화(메뉴화면) 및 관련된 정보를 효율적으로 표시하기 위하여 인간공학 확인 및 검증 측면에서의 설계방안도 연구되어야 할 것이다[8].

## 4. 요 약

본 논문에서는 안전-필수 소프트웨어 구현을 위한 수명주기를 설명하였으며 이에 따른 단계마다의 필요한 업무를 기술하였다. 또한 관련 규제 및 기술기준 등을 기술하고 이를 기반으로 단계별로 수행하여야 하는 업무를 제시하였다. 원자력발전소 안전계통의 하드웨어 및 소프트웨어의 품질 및 특성을 조사하였으며 특히 수명주기별 수행되는 확인 및 검증업무를 검토하였다. 아울러 테스트에서는 기존의 수작업에서 수행하던 업무가 소프트웨어 도구를 이용하여 실행됨을 보여주었다. 또한 각각 독립적으로 수행되던 산업계의 CMMI와 원자력발전소의 소프트웨어 수명주기를 비교, 분석하여 공통점을 제시하였다.

## 참고문헌

[1] IEEE Std 7-4.3.2-2003, "IEEE Standard Criteria for Digital Computers in Safety Systems of Nuclear Power Generating Stations".

[2] IEEE Std 1012-1998, "IEEE Standard for Software Verification and Validation".

[3] IEEE Std 1228-1994, "IEEE Standard for Software Safety Plans".

[4] NUREG/CR-6463, "Review Guidelines on Software Languages for Use in Nuclear Power Plant Safety Systems".

[5] EPRI TR-106439, "Guidance, on Evaluation and Acceptance of Commercial Grade Digital Equipment for Nuclear Safety Applications".

〔6〕 NUREG/CR-6430, "Software Safety Hazard Analysis".
〔7〕 Kwon-Ki Moon, "Module Testing Techniques for Nuclear Safety Critical Software Using LDRA Testing Tool", KNS, Autumm Seminar, 2006.
〔8〕 NUREG-0700, "Human-System Interface Design Review Guideline", Rev.01.

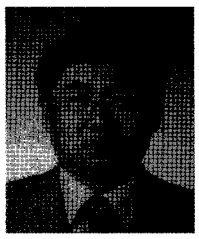
**장 영 우**

1975 서울대학교 공과대학(학사)
1983 University of Kansas(석사)
2003 한국과학기술원 원자력 및 양자공학과(박사)
2006~현재 한국전력기술(주) 근무
관심분야 : UML, 소프트웨어 확인 및 검증, 수치해석, 정수론
Email : ywchang@kopec.co.kr

**하 재 홍**

1990 경북대학교 전자공학과(학사)
1993 경북대학교 전자공학과(석사)
1990 삼성전자 근무
1992 한국원자력연구소 근무
2006~현재 한국전력기술(주) 근무
관심분야 : 신호처리기술, 소프트웨어 확인 및 검증, 공정계측 및 제어계통 설계
Email : jhha2@kopec.co.kr

**장 훈 선**



1980 명지대학교 전자공학과(학사)
1985 인하대학교 전산학과(석사)
1976 한국원자력연구소 근무
2006~현재 한국전력기술(주) 근무
관심분야 : 소프트웨어 객체지향, 소프트웨어 확인 및 검증, 임베디드 소프트웨어 개발
Email : hschang@kopec.co.kr

**김 재 학**

1983 중앙대학교 전기공학과(학사)
1985 중앙대학교 전기공학전공(석사)
1985 현대엔지니어링 입사
1993 한국원자력연구소 근무
2006 현재 한국전력기술(주) 이관근무
관심분야 : 원자력분야 소프트웨어 확인 및 검증, 소프트웨어 신뢰도 시험
Email : ghost@kopec.co.kr