'99 추계학술발표회 논문집
한국원자력학회

안전필수 소프트웨어의 확인 및 검증
Safety Critical Software Verification and Validation

김희철, 백영우, 윤청
충남대학교
대전시 유성구 궁동 220
김복렬
한국원자력안전기술원
대전시 유성구 구성동 373-1

요약

원자력 산업계에서 컴퓨터 기술의 적용은 다른 응용 분야와 마찬가지로 요구되는 시대적 변화라 할 수 있다. 그러나 현재까지 원자력 분야에 컴퓨터 적용이 용이하지 않았던 주된 요인은 시스템 제어가 소프트웨어로 이루어지는 경우 하드웨어를 사용하는 것보다 오류를 발견하기 어려워 안전 필수 시스템의 경우 소프트웨어를 사용하는 것에 많은 저항이 있어 왔다. 이러한 안전성 및 신뢰성을 확보하는 문제가 디지털 계측제어 시스템의 경우 중요한 쟁점으로 떠오르고 있다. 이러한 문제를 해결하기 위해 원자력 발전소 계측제어 소프트웨어에 요구되는 품질보증 활동에 대한 연구가 수행되어 왔다. 소프트웨어의 품질을 보증하는 중요한 수단으로서 확인 및 검증은 소프트웨어 시스템 개발 과정 중 모델링과 분석결과에 대하여 잠재적인 위험이 발생될 확률과 심각성을 판단할 수 있으며, 오류를 제거하고 위험을 제거할 수 있다. 본 연구는 안전필수 소프트웨어의 생명주기 전반에 걸쳐 이루어지는 확인 및 검증의 계획 및 절차에 대하여 설명하고자 한다.

Abstract

Application of computer technology in nuclear power plant is also a necessary transformation as in other industry fields, Bur until now, application of software technology was not wide-spread because of its potential effect to safety in nuclear field, Insurance for safety and reliability is a issue of digital instrumentation and control systems, To solve these problems, there must are researches of assurance of instrumentation and control software quality in nuclear field, As primary method to assurance software quality, software verification and validation can decide probability and seriousness of potential hazard and can remove error and hazard, This research illustrates software verification and validation plan and process through software development.

1. 서론

시스템 제어가 기존의 하드웨어 중심으로부터 소프트웨어 중심으로 발전하고 있다. 소프트웨어

시스템은 오류를 발견하기 어렵고, 특히 안전 필수 소프트웨어(예를 들면, 원자력발전제어시스템, 항공시스템)인 경우 소프트웨어에 대한 요구가 더욱 높다. 이러한 이유로 계측제어 시스템에 필요한 소프트웨어의 품질을 보증하고 안전성 및 신뢰성은 반드시 확보하여야 하며 이를 달성하기 위해 소프트웨어 개발 생명주기에 따른 개발체계를 확립하고 소프트웨어에 대한 확인 및 검증을 수행함으로서 보다 높은 안전성 및 신뢰성을 확보하고 품질을 보증할 수 있다.

본 논문은 안전필수 소프트웨어의 개발 생명주기에 따른 소프트웨어 확인 및 검증의 계획 및 절차에 대해 설명하고자 한다.

2. 소프트웨어 확인 및 검증(Software Verification and Validation)

검증(validation)이란 소프트웨어가 고객의 요구사항을 만족시키는가의 여부를 밝히는 활동이고, 확인(verification)이란 소프트웨어가 지정된 기능에 대해 정확히 수행되는가의 여부를 밝히는 활동이다. 소프트웨어 시험도 검증과 확인 과정이라 볼 수 있다.

2.1 확인(Verification)

확인은 개발주기상에서 현 단계의 산출물이 바로 이전단계의 산출물과 일치하는가를 결정하는 과정으로, 사용자가 바라는 대로 소프트웨어가 만들어졌음(설계 및 구현됨을 의미)을 확신할 수 있도록 수행하는 활동이다. 즉, 사용자가 원하는 대로 소프트웨어가 제대로 만들어지고 있는지, 요구사항 명세서에 기술되어 있는 사용자 요구사항대로 시스템이 동작하는지 검사하는 것이다. 확인은 설계, 구현, 시험 단계에서 이루어지며, 시스템 개발의 각 단계에서 그 이전 단계의 결과를 올바르게 반영하고 있는지, 전 단계에서 정해진 사양과 일치하는지 검토함으로써 시스템이 정확히 구축되는지 검사하는 활동이다. 즉, 설계 과정에서는 분석 과정에서 규명된 요구사항이 설계에 제대로 반영되었는지, 구현 단계에서는 설계 명세서대로 프로그래밍이 이루어 졌는지 확인한다.

안전필수 소프트웨어인 경우, 어떤 시스템 또는 구성요소에 대한 요건이 완전하고 정확한지, 각 개발단계의 제품이 바로 직전 단계에서 부여된 요건을 충족시키는 지를 결정해야 한다.

이러한 확인 과정은 개발자들에 의해 이루어질 수 있으며 시험 과정에서 고객에 의해 이루어지기도 한다. 경우에 따라서는 독립적인 확인자(independent verifier)를 두어 확인하기도 한다. 만약 시스템이 이에 합당하게 설계 또는 구현되지 못했을 때 이의 수정이 요구된다. 이때 설계 명세서 또는 프로그램의 수정이 불가피하므로 요구사항을 만족시켜야 하는 것은 개발자의 의무이다. 설계 문서, 또는 시스템이 요구사항을 제대로 반영되어 만들어졌는지 확인하기 위해 여러 기법이 사용되고 있다.

2.2 검증(Validation)

요구사항 분석 단계에서의 산출물인 요구사항 명세서에 고객의 요구를 제대로 반영하여야 한다. 개발자와 사용자는 실제 개발에 들어가기 전에 요구사항 분석 결과에 대하여 자신감과 높은 신뢰도를 가져야 한다. 검증이란 사용자가 원하는 올바른 제품을 만들었는지, 요구사항 분석의 결과가 사용자가 바라는 요구를 반영한 만족할만한 것인지 검사하는 것이다. 따라서 만들어지는 시스템이 프로젝트 초기 단계에서 정의된 목표와 요구사항에 부합하는지 검토하여 사용자가 원하는 올바른 시스템이 구축되었는지 검증한다. 요구사항 분석의 결과인 요구사항 명세서는 설계, 구현

등 앞으로 예상되는 개발 활동의 근거가 된다.

안전필수 소프트웨어는 그 자체가 가지고 있는 특성으로 하여, 최종 시스템 또는 구성요소가 규정된 요건들을 정확히 준수할 것을 요구하고 있다.

검증을 위해 사용되는 여러 기법이 있으며 시제품 개발(prototyping), 기술적인 검토(technical review), 시뮬레이션(simulation) 등을 포함한다. 시제품 개발은 개발자와 사용자가 필요로 하는 시스템에 대한 올바른 시각을 가지고 있는지 검사하기 위해 시스템의 초기 버전을 만들어 보는 것이다. 기술적인 검토는 개발자와 사용자가 만나서 프로젝트의 문서인 요구사항 명세서, 설계명세서, 프로젝트 계획서 등을 검토하는 것이다. 시뮬레이션은 개발에 들어가기 전에 기능 이외의 요구사항이 완전하고 일관성이 있는지, 논리적으로 모순이 없는지, 주어진 시간 내에 작동하는지 알기 위해 시스템을 모델링 해보는 것이다. 일반적으로 많은 CASE도구들이 요구사항 명세서를 시뮬레이션할 수 있도록 기능을 제공하고 있다.

요구사항 명세서는 고객의 요구사항을 정확히 반영하고 있어야 한다. 실제로 프로젝트를 수행하다 보면 개발자와 고객이 시스템에 대하여 서로 다른 생각을 가질 수 있으며, 요구사항 명세서를 포함한 문서 및 시스템의 수정이 요구된다.

확인 및 검증은 만들어질 시스템이 가지고 있는 위험요소를 초기에 발견할 수 있도록 하고, 신뢰성과 유지보수성 및 확장성을 얻을 수 있도록 하는데 그 목적이 있다.

3. 소프트웨어 확인 및 검증 계획(Software Verification and Validation Plan)

소프트웨어 확인 및 검증을 수행하기 전에 우선 확인 및 검증 계획을 세우는 것이 필요하다. 소프트웨어 확인 및 검증 프로세스는 소프트웨어 확인 및 검증 계획을 확립한 후 수행하며, 타스크의 계획 및 보고, 개괄보고, 예외보고 및 소프트웨어 확인 및 검증 최종보고 등으로 이루어진다.

소프트웨어 확인 및 검증 계획은 소프트웨어 개발주기의 요구사항 분석 단계로부터 시작하며, 설계, 구현, 통합, 검증, 설치, 운전 및 유지보수 등 전반 과정을 거쳐 확인 및 검증 타스크들은 수행된다. 각 단계에서는 그에 대응하는 확인 및 검증 보고서가 작성된다. 이러한 보고서는 각 단계의 산출물의 일부가 되며, 위에서 설명한 문서들을 포함할 수 있다.

소프트웨어 확인 및 검증 계획은 시스템의 요구사항을 만족시켜야 할뿐더러 또한 소프트웨어 요구사항도 만족시켜야 한다. 소프트웨어 시험 계획은 소프트웨어 요구사항 분석과 동시에 이루어질 수 있다. 이러한 소프트웨어 확인 및 검증 계획은 프로젝트 개발 과정에서 수요에 따라 세부적으로 변경될 수도 있다. 소프트웨어 시험 계획은 그 개발과정에서 소프트웨어 요구사항의 오류를 발견할 수 있다.

소프트웨어 확인 및 검증을 수행함에 있어서 일반적으로 확인 및 검증 팀을 두는 것이 바람직하며, 확인 및 검증 팀은 개발팀과 따로 존재함으로써 독립성을 보증하여야 한다. 확인 및 검증 팀의 독립성은 다음과 같은 방면에서 표현된다.

- 기술적 독립(technical independence)

기술적 독립은 우선 팀 성원들이 소프트웨어의 개발에 참가하는 것을 허용하지 않는다. 이 팀 성원들은 시스템 설계에 대한 지식을 갖춰야 하거나 연관된 경험과 시스템 공학의 백그라운드를 가져 가히 시스템을 포착할 수 있어야 한다. 팀은 시스템의 요구사항을 분석하고, 시스템을 확립하는 방법을 제안하며, 또한 문제가 부닥칠 때 시스템의 개발팀에 의해 영향을 받지 말아야 한다. 기술적 독립은 팀이 소프트웨어 요구사항 분석, 설계 및 코딩과정에서의 민감한 에러를 발견하는데 있어서 결정적 작용을 한다.

기술적 독립성을 보증하기 위해 확인 및 검증 팀은 컴퓨터가 제공해주는 환경(예를 들면, 컴파일러, 어셈블러, 유틸리티)이나 제공해주는 툴을 사용할 수도 있으나 소프트웨어 시험을 수행할 때 이러한 툴자체가 에러를 덮어 감추는 것을 방지해야 한다. 팀은 자체의 시험 툴을 사용하거나 개발하여 이러한 결함을 극복할 수 있다.

– 관리적 독립(managerial independence)

관리적 독립은 확인 및 검증 팀에 할당된 책임의 프로젝트 관리 및 수행팀의 책임과의 분리를 말한다. 프로젝트 관리 및 수행팀에서 요구사항을 규정할 때 팀은 독립적으로 소프트웨어/시스템의 분석 및 시험, 기술, 타스크 수행 스케쥴 및 기타 기술적 요소를 결정지어야 한다.

–재무적 독립(financial independence)

재무적 독립은 말 그대로 확인 및 검증 팀이 예산을 유지함에 있어서 관리 및 개발팀과 분리하는 것을 말한다. 이러한 독립은 경제적 압력과 영향 혹은 자금에서의 견제 등으로 의해 V&V 타스크 수행이 지체되거나 중단됨으로 하여 제때에 결과물을 제출 못하는 것을 방지해줄 수 있다.

소프트웨어 확인 및 검증 계획은 또한 확인 및 검증 프로세스를 수행하는 일정, 소요되는 자원, 툴, 기술 등에 대해서도 설명되어야 한다.

소프트웨어 확인 및 검증 계획을 수행하기 위해 전반 소프트웨어 생명주기에 걸쳐 확인 및 검증 타스크를 수행한다.

안전필수 소프트웨어의 확인 및 검증 계획은 위에서 서술한 요소들을 포함하여야 할 뿐만아니라 소프트웨어의 관리, 이행 및 자원 특성들을 포함하여 철저한 확인 및 검증 계획을 생성하는 것이 바람직하다. 또한 확인 및 검증 조직이 준수해야 할 국제, 국가, 산업 및 회사의 표준과 지침들이 나열되어야 한다.

4. 소프트웨어 생명주기에서 확인 및 검증(Software Verification Validation through Software Life Cycle)

소프트웨어 확인 및 검증은 소프트웨어 생명주기에 걸친 소프트웨어 품질을 보증하기 위한 활동으로서 소프트웨어 확인 및 검증 계획과 동시에 진행하여야 한다. 생명주기 각 단계에서는 대응하는 소프트웨어 확인 및 검증 계획을 생성하며, 수행하여야 할 타스크들, 확인 및 검증을 수행하기 위한 입력물 및 산출물을 정의한다. 소프트웨어 생명주기에 따른 확인 및 검증 타스크들, 입력물, 산출물은 그림 1과 같다.

4.1 소프트웨어 요구사항 확인 및 검증(Software Requirement V&V)

소프트웨어 요구사항 확인 및 검증 활동은 시스템에 할당된 소프트웨어 요구사항이 적절하고 정확한지를 점검하며, 소프트웨어 요구사항이 어느 정도 제대로 서술되었는지(정확하고, 완벽하며, 모호성이 없으며, 시험할 수 있게)를 점검하는 것이다. 소프트웨어 요구사항은 반드시 소프트웨어 목표가 이루어질 수 있도록 구성되어야 한다.

안전필수 소프트웨어는 소프트웨어와 하드웨어를 포함한 전반 시스템에서 소프트웨어의 역할, 작용 등을 정확히 파악하여야 하며, 소프트웨어가 제대로 작동하지 못할 경우 시스템 안전성에 가져다 주는 영향의 심각성 등도 고려하여야 한다.

소프트웨어 요구사항 확인 및 검증은 소프트웨어 요구사항이 정확하고 완벽하고, 일관되고, 엄밀하고, 읽기 쉬우며 시험 할 수 있는지를 확인하는 것이다. 또한 소프트웨어 요구사항 확인 및 검증은 소프트웨어, 하드웨어를 포함한 전반 시스템의 요구사항도 만족시켜야한다.

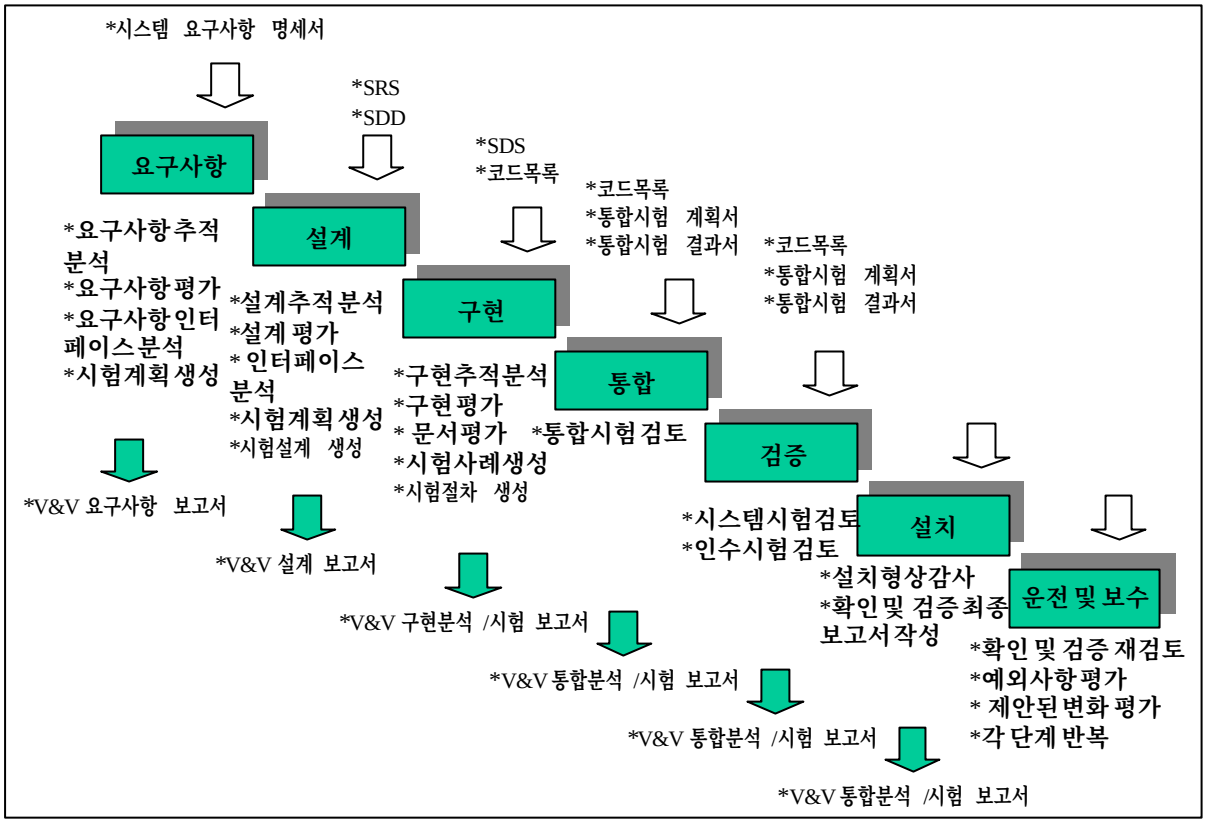

그림 1 소프트웨어 생명주기 확인 및 검증

모호하게 서술된 요구사항(예를 들면 정확하지 않고, 완벽하지 않으며 모호하고 시험할 수 없는)은 소프트웨어의 비용을 늘이고 신뢰성에 문제점을가져다준다. 소프트웨어 요구사항이 충분히 전달되었다 하더라도 유지보수를 함에 있어서 문제점을 가져다줄 수 있다. 그것은 보편적인 요구사항(유지보수성, 품질, 재 사용성)이 개발초기에 고려되지 않았기 때문이다.

또한 소프트웨어 개발과정에 요구사항을 변경한다는 것은(예를 들면 새로운 기법, 새로운 임무, 새로운 지식, 새로운 시스템 인터페이스 등을 통합) 상당한 에러를 발생하는 기회를 가져다준다. 안전필수 소프트웨어인 경우 이러한 영향은 재산, 인명피해와 같은 심각한 영향을 가져다 줄 수 있으므로 요구사항의 변경은 더욱 신중하게 고려하여야 한다. 소프트웨어 요구사항 확인 및 검증은 이러한 문제점의 발생을 방지하기 위한 노력이다.

소프트웨어 요구사항에 대한 확인 및 검증은 시스템 개발 초기(개발 가능성에 대한 연구 등)에 작성된 문서에 대한 점검도 포함한다. 소프트웨어의 가정(assumption), 알고리즘(algorithm) 및 물리적 규칙(physical rule) 등을 포함한 요구사항이 프로젝트에 적합함을 미리 확인한 다음 소프트웨어 확인 및 검증은 비로소 이러한 사항들을 점검한다. 안전필수 소프트웨어에서는 소프트웨어, 하드웨어 및 인간, 즉 인간-기계의 요소도 확인되어야 한다.

소프트웨어 요구사항 확인 및 검증 활동의 입력물 문서는 자연언어(natural languages) 혹은 정형된 수학언어(formal mathematical languages)로 서술될 수 있으며, 그래픽이거나 차트로도 표시할 수 있다.

소프트웨어 요구사항 확인 및 검증과 동시에 소프트웨어 시스템 및 인수시험 계획이 이루어진다. 소프트웨어 확인 및 검증은 시스템에 대한 시험을 점검함으로서 포괄적인 시험과 적합한 자

원의 준비를 보증한다.

4.2 소프트웨어 설계 확인 및 검증(Software Design V&V)

소프트웨어 설계 확인 및 검증 활동은 소프트웨어 설계가 요구사항을 만족시키는지를 확인해야 할 뿐만 아니라 소프트웨어 설계가 전반 시스템의 요구사항도 만족시키는지를 검증해야 한다. 전반 시스템에 대해 확인을 진행하기 전에 소프트웨어 요구사항 및 소프트웨어 설계 확인(verification)에 대한 설명이 있어야 한다.

설계 에러는 기능적 요구사항이 잘못 전달되었거나, 시간, 데이터 구조, 메모리 공간 및 정확성의 부정확한 수행에 의해 유도된다. 소프트웨어 설계 확인 및 검증은 소프트웨어 요구사항이 제대로 전달되고, 충분하게 수행되었음을 확인한다. 소프트웨어 설계 확인 및 검증 활동은 소프트웨어 요구사항에 대한 확인 및 검증 프로세스, 소프트웨어 설계 및 소프트웨어 설계 보완이 완성된 후 수행된다. 소프트웨어 요구사항의 추적성, 평가성 및 인터페이스 분석 등 타스크는 소프트웨어의 요구사항이 빠지지 않았고, 완벽하고 정확하게 서술되었음을 확인한다.

안전필수 소프트웨어 설계의 확인 및 검증은 일반적인 사항들을 고려해야 할 뿐만아니라 정확도, 안전성, 보안, 타이밍 등 요건들도 고려되어야 한다.

소프트웨어 설계 확인 및 검증 활동에서는 또한 소프트웨어 통합시험 계획이 생성되며, 이러한 시험을 위한 설계를 수행하여야 한다.

4.3 소프트웨어 구현 확인 및 검증(Software Implementation V&V)

소프트웨어 구현 확인 및 검증은 코드에 대한 확인이며, 코드 확인 활동은 소프트웨어의 설계가 코드로 정확하게 구현되었는지를 점검하는 것이다. 일반적으로 이 활동은 코드의 세부까지 상세히 검사해야함으로 지루하게 느껴지기도 한다. 점검의 자동조작은 코드의 오류를 찾을 때 사람으로 인한 에러를 방지할 수 있으며, 또한 프로세스의 속도도 증가할 수 있다.

문법적 에러는 구조적 프로그래밍, 코드의 재사용, 프로그램 표준 및 스타일의 적용을 통해서 줄일 수 있다. 또한 강력한 컴퓨터 언어, 더 좋은 컴파일러의 적용 등으로 하여 에러를 줄이는데 더 쉬워졌다. 하지만 설계를 코드로 변화시키는 과정에서 종종 문제가 발생할 수 있다. 때문에 코드 확인 및 검증은 소프트웨어 확인 및 검증 활동에서 중요한 위치를 차지한다.

코드 확인은 에러를 발견하고 제거하는 마지막 기회이다. 이러한 에러는 불필요한 비용의 원인으로 되며, 시험의 정상적인 진행을 지연시킬 수 있다.

구현 확인 및 검증에서 단위시험 계획 및 시험절차를 생성하며, 또한 실지 단위시험을 통하여 코드의 오류를 제거할 수 있다.

4.4 소프트웨어 통합 확인 및 검증(Software Integration V&V)

소프트웨어 통합은 여러 모듈들 간의 인터페이스와 관련된 에러를 찾기 위한 활동으로서 각 모듈들이 통합된 후 신규 시스템이 통합시험 계획서에 표현된 통합 시험 요구사항을 만족하는지 검증한다.

소프트웨어 통합 확인 및 검증은 각 모듈을 통합했을 때의 소프트웨어가 제대로 사용자의 요구사항을 만족시키는지를 검증하는 것이다. 이를 달성하기 위해 상세설계의 통합시험 요구사항에 근거하여 이를 정확하고 철저하게 시험하기 위한 시험 케이스 및 데이터를 개발하여 프로그램이

시스템으로 통합된 후 신규 시스템이 통합 시험계획서에 표현된 통합 시험 요구사항을 만족하는지 검증한다.

통합 시험의 성공여부의 판단기준과 작업 일정 및 책임, 소요자원에 대한 계획을 세우고 통합 시험단계를 수행하는데 있어 지켜야 할 표준 및 절차를 수립한다.

통합시험의 필요성은 구현단계에서 이미 단위시험 및 코드 안전성 분석 및 확인을 했더라도 각 모듈을 통합했을 때의 오류가 발생할 가능성이 존재하기 때문에 모듈들간의 인터페이스의 오류를 발견하고 제거함으로써 전반 소프트웨어의 안전성 및 신뢰성을 보증할 수 있다.

통합시험을 수행한 후 예산 및 작업계획에 대한 실적을 분석하여 주요 문제점 및 개선해야 할 사항 등을 찾아내고, 산출물에 대한 내부검토를 한다.

4.5 소프트웨어 검증 확인 및 검증(Software Validation V&V)

소프트웨어 검증은 통합시험에서 소프트웨어 설계에서 발생된 오류를 발견하며, 이러한 오류를 수정한 후 개발된 소프트웨어가 시스템의 요구사항을 만족하는지 조사하며 고객 요구사항의 만족 여부를 검증하는 활동이다.

소프트웨어 검증 확인 및 검증은 소프트웨어 하드웨어가 결합된 시스템의 기능들이 요구사항과 설계 내용대로 이행되는 지와 요구된 성능을 발휘하는지를 검증하는 것이다. 이러한 목적을 달성하기 위해 시스템 시험과 사용자 인수시험 수행한다.

소프트웨어는 컴퓨터의 한 구성요소이기 때문에 소프트웨어 자체가 안전하다고 해서 그 소프트웨어가 위치해야 하는 하드웨어와의 인터페이스 관련부분에 대한 안전성을 확신할 수가 없다. 그러므로 소프트웨어의 각 모듈을 통합 한 후 소프트웨어를 컴퓨터 시스템이 요구하는 하드웨어 및 다른 소프트웨어 시스템들과 통합하여 시험하여야 한다.

또한 사용자 인수시험을 통하여 개발된 소프트웨어가 고객의 요구를 만족하는지를 시험한다. 인수시험을 수행함에 있어서 일반적으로 알파 및 베타 두 가지 시험기법을 택한다. 사용자 인수시험 단계 동안의 시험과정과 결과에 대해 산출된 시험 문서들을 점검함으로서 과정이 계획대로 수행되었는지, 그 결과 시스템의 기능 및 품질 등 기대했던 결과를 만족시켰는지 검증한다.

4.6 소프트웨어 설치 확인 및 검증(Softwre Installation V&V)

소프트웨어 설치 확인 및 검증은 소프트웨어가 고객에게 배달한 후에 진행하는 마지막 인수 시험으로서 소프트웨어가 정확하게 고객에게 배달되었는지와 소프트웨어 및 하드웨어의 인터페이스 사이의 정확한 연계를 확인한다.

안전필수 소프트웨어 설치 확인 및 검증은 소프트웨어와 하드웨어의 인터페이스를 중요시 하여야 한다. 또한 고객(운전자)에 대한 훈련도 매우 중요하다.

소프트웨어 설치 확인 및 검증 활동을 통하여 최종 확인 및 검증 보고서가 생성되며, 소프트웨어 생명주기 각 단계에서의 확인 및 검증 결과가 정리된다.

4.7 소프트웨어 운전 유지보수 확인 및 검증(Software Operation and Maintenance V&V)

운전 및 유지보수 확인 및 검증은 소프트웨어가 다 만들어진 후 모든 소프트웨어 확인 및 검증 활동이 가능해지며 중복되어 빠뜨린 것이 없다는 것을 확인한다. 소프트웨어 운전 유지보수 확인 및 검증 활동은 시스템의 변화가 미치는 영향도 점검한다.

소프트웨어 운전 확인 및 검증은 통합된 시스템에 대한 주기적인 점검을 진행하며, 시스템의 작동에 영향을 주는 모든 변화를 기록한다. 소프트웨어 유지보수 확인 및 검증은 유지보수 범위에서 소프트웨어 확인 및 검증을 계획한다.

만약 소프트웨어 개발과정에서 소프트웨어 확인 및 검증이 진행되지 않았으면 작동과 유지보수 단계에서 소프트웨어 개발 단계에 연관되는 확인 및 검증을 고려해야 한다. 이러한 확인 및 검증 활동에는 소프트웨어 요구사항의 생성 및 소스 코드로부터의 소프트웨어 설계 정보도 포함된다.

5. 결론

소프트웨어 확인 및 검증은 소프트웨어 안전분석 및 형상관리와 더불어 소프트웨어의 품질을 보증하기 위해 필요하다. 본 논문은 안전필수 소프트웨어를 개발함에 있어서 확인 및 검증의 계획과 소프트웨어 개발 생명주기에 걸쳐 수행되는 확인 및 검증 타스크들에 대해 기술하였다.

안전필수 소프트웨어의 확인 및 검증은 일반 소프트웨어 확인 및 검증에서의 요건들을 반드시 지켜야 할 뿐만아니라, 그 자체의 특성으로 하여 소프트웨어의 정확성, 안전성, 보안성, 타이밍, 시스템 기타 요소와의 인터페이스 등 소프트웨어의 안전성, 신뢰성에 영향을 가져다 주는 요건들도 충분히 고려되어야 한다.

안전필수 소프트웨어의 확인 및 검증에 있어서 소프트웨어의 품질을 보증할 수 있는 표준 혹은 틀을 확립하는 것이 매우 중요하다. 체계적인 확인 및 검증을 수행하는 것은 일반적인 소프트웨어 뿐만아니라 안전성이 중요시되는 시스템의 일부인 소프트웨어를 개발하는데 있어 크게 이바지 할 수 있을 것이다.

참고문헌

[1] 시스템공학연구소, "한국형 정보시스템 개발방법론"

[2] 윤청, "성공적인 소프트웨어 개발 방법론", 생능 출판사, 1996

[3] 과학기술처, "디지털 계측제어시스템 안전성 평가기술 개발", 1997

[4] J.D.Lawrence "Software Reliability and Safety in Nuclear Reactor Protection Systems", 1993

[5] Roger S. Pressman, 유해영 역, "Software Engineering", 홍重堂, 1995