원전의 컴퓨터 소프트웨어 품질확보 방안 고찰 - A Study on the Achieving Software Quality in Nuclear Field -

고 한 준 *
Koh, Han-Joon
오 연 우 **
Oh, Yon-Woo

Abstract

Quality Assurance Principles must be effectively implemented in developments and in use of safety critical software in nuclear industry. Brief definitions related to computer software and quality assurance were defined and several methods for evaluating software quality were proposed herewith. Independent verification and validation was suggested to assure the quality of safety critical software.

1. 서 론

1.1 연구의 배경

원자력 산업에서 컴퓨터 소프트웨어의 이용은 극히 중요한 부분이므로 소프트웨어는 일반적으로 이용되는 비체계적인 방법으로 개발 되거나 사용되어져서는 안된다.

10 CFR 50 Appendix B 에 따르면 원자력발전설비 및 핵연료 재처리에 있어서의 품질보증 프로그램은 "공공의 생명과 안전을 위험하는 예기치 않은 사고를 미연에 방지하거나 완화하는 모든 구조물, 계통, 기기 등에 적용되어져야 한다." 라고 기술되어 있다. 이러한 이유로 원자력분야의 소프트웨어의 이용은 플랜트 설계, 건설, 운전 등의 품질보증 방법과 같이 엔지니어링 관리기법을 따라야 한다. 이것은 원자력분야에서 소프트웨어는 설계에서 건설 또는 운전에 이르기까지의 모든 단계에서 적용되고 있는 실정이다.

비상대응설비, 원전안전정보시스템등은 물론 특히, 최근에는 전세계적으로 원자력발전소의 운전 및 control이 컴퓨터에 의한 자동화의 방법의 도입도 시도되고 있다. 그만큼 원전의 설계에서 운전 및 안전관리체계에 이르기까지 소프트웨어의 비중이 커지고 있다. 이런 연유로, 소프트웨어가 정확하다고 판단할 수 있는 객관적인 수단이나 체계적인 검증절차를 확립하기 위해 소프트웨어 품질보증에 관한 문제가 거론되어 왔고, 따라서 이에대한 품질보증체계가 확립되어 이용되어 왔다. 그럼에도 불구하고, 소프트웨어의 품질을 제3자의 관점에서 평가한다든지, 시험 및 평가의 기법등이 활용되지 못하는 것으로 파악되고 있다.

* 한국원자력연구소 품질기술실
** 한국원자력연구소 품질평가실

본 연구에서는 소프트웨어 품질보증이 무엇이고, 소프트웨어의 품질평가의 수단으로서 활용될 수 있는 기법들 중 품질을 개량화하는 개념과 V/V 를 위하여 자주 이용되는 기술검토, 감사의 방법에 대하여 연구하고자 한다. 왜냐하면 품질은 측정할 수 있을 경우에만 성취가 가능하기 때문이다.

2. 소프트웨어 품질보증의 개념

소프트웨어 품질보증과 관련하여 학자들 또는 Code and Standard에서 정의한 내용을 원문 그대로 소개하면 다음과 같다.

QUALITY

- Any character which makes an object good.
- Conformance to requirements

SOFTWARE

- IEEE software engineering terminology standard ;
  Computer programs, procedures, rules and possibly associated documentation and data pertaining to operation of a computer system.
- Department of Defense standard on defense system software development. (DOD-STD-2167) ;
  A combination of associated computer instructions and computer data definitions required to enable the computer hardware to perform computational or control functions.

SOFTWARE QUALITY

- Conformance to software requirement
- IEEE ;
  The degree to which software possesses a desired combination of attribute.
- DOD-STD-2168 ;
  The degree to which the attributes of the software enable it to perform its specified end item use.
- The fitness for use of the total software product.

SOFTWARE QUALITY ASSURANCE

- NQA-1;
  QA is all those planned and systematic actions necessary to provide adequate confidence that a structure, system, or component will perform satisfactorily in service.
- Juran ;
  QA is the activity of providing to all concerned the evidence needed to establish confidence that quality function is being performed adequately.
- IEEE ;
  QA is a planned and systematic pattern of all actions necessary to provide adequate confidence that the item or product conforms to established technical requirement.
- Dobbins and Buck ;
  SQA is a systematic efforts to improve the delivery condition.
- Fisher and Baker ;

SQA is the functional entity performing software quality assessment and measurement.

- Donald Reifer ;
SQA is the system of methods and procedures used to assure that the software product meets its requirements. The system involves planning, measuring and monitoring developmental activities perofrmed by others. SQA is the systematic activities providing evidence of the fitness for use of the total software product.

위의 여러가지 정의에서 알 수 있는 바와 같이, 소프트웨어 품질보증이란 요구되어지는 소프트웨어 품질(software quality)을 성취하기 위해서 계획되어지고 체계화된 모든 활동을 의미하는 것이며 따라서 품질보증 활동은 materials, 데이타, supplies 또는 서비스 등이 기 확립된 기술적인 요구사항과 일치하는지를 확인케 하여주며 또한 소프트웨어가 만족스럽게 수행되는지를 확인케 한다. 소프트웨어 품질보증의 본질(essence)은 프로그램의 결점(defects)을 사전에 방지하고 그들이 발견되면 제거함으로써 소프트웨어의 사용도(usuability)와 보전도 (maintainability)를 높이는데 있다.

소프트웨어 품질보증은 소프트웨어의 근본적인 결함이나 에러의 잠재력을 제거하고자 하는 것으로써 소프트웨이 품질보증 프로그램은 소프트웨어의 라이프싸이클을 통하여 적절하고 체계적인 품질보증기법을 적용함으로서 소프트웨어 에러의 가능성을 감소시키자는데 그 목적이 있다.

3. 검증 및 확인 (V/V;Verification and Validation)

3.1 V/V 개념

V/V에 대한 여러 정의를 정리하면 다음과 같다.

- Barry Boehm ;
Verification is doing the job right and validation is doing the right job.
- IEEE glossary ;
Verification in the process of determining whether or not the products of a given phase of the software development cycle fulfill the requirements established during the previous phase. Validation is the process of evaluating software at the and of the software development process to ensure compliance with software requirements.
  - DOD-STD-2168 ;
Verification is the process of determining the products of a given phase of the software development cycle to ensure correctness and consistency with respect to the products and standards provided as input to that phase. Validation is the process of evaluating software to ensure compliance with specified requirements.

위의 정의에서와 같이 V/V는 신뢰도 (reliability)를 향상하기 위한 체계적인 접근방법으로서 프로그램 개발과 병행하여 수행될 때 효과적이며 소프트웨어 요구사항, 설계 및 코딩상의 에러들을 조기에 발견할 수 있게 한다. 이러한 V/V는 프로그램 개발 조직과 독립인 프로그램 개발자에 의해 수행되어야 한다.

Verification은 computer instructions이 design을 정확히 나타내고 design은 수학적인 모델을 정확히 나타내었다 하는 점이 증명되도록 고려되어져야 한다.

Verification은 전단계에서 정립된 요구사항 (requirements)들에의 일관성 (consistency)을 확립하기 위하여 각 개발단계 products의 세부적인 검토로 이루어진다. 반면에 validation은 검증된 컴퓨터 프로그램이 자연현상 (natural phenomena)을 적절히 표현했다는 것을 입증해야 한다.

Validation은 컴퓨터 프로그램을 테스트하고 그 테스트 결과를 평가하는 것으로 이루어지며 validation

의 타당성을 확인하기 위하여 개발자에 의해 수행된 testing은 V/V의 책임이 있는 독립조직에 의해 평가되고 (evaluated), 보완되어야 하며(supplemented), 성능이 수행(performed) 되어야 한다. 프로그램은 테스트결과들이 요구사항시방서의 내용을 만족시킨다고 할때 확인 (validation) 된다.

3.2 원전 S/W 개발시 V/V 방안

소프트웨어 시스템의 개발에 있어서 V/V는 조직적으로 독립인 기관에서 수행되어야 하는 것이 일반적이며 시스템의 규모가 방대하고 복잡할수록 이점은 더욱 강조되어야 한다. 전체
개발에 대한 V/V의 비중은 project management, 프로젝트의 성격, 규모에 따라 다르며 경험적인 통계에 의하면 사무처리용 소프트웨어의 경우 10-15%, 과학기술, 공정제어, 시스템 접속 및 통신 소프트웨어는 20-30%정도이고, 원전에 직접 이용되는 경우는 그보다 훨씬 높게 평가되어야 할 것이다.

원자력 및 항공우주분야 등에 쓰이는 safety critical software 프로젝트에서는 독립적인 검증 및 확인 (IV&V; independent verification and validation)이 수행되어 high quality 및 reliability를 갖는 소프트웨어가 개발되어야 하며 이러한 IV&V에 소요되는 비용은 전체 개발비용의 약 25%인 것으로 보고되고 있다.

V/V를 개발조직내의 어떤 조직이 수행할 수 도 있으나, 본래 V/V의 취지는 개발조직과는 조직적으로 독립인 조직이 수행하는 것이 좋다는 의견이다. 다만, 개발조직내에서 V/V를 수행할 경우, 프로그램개발자가 아닌 'Personally Independent'의 관계에 있는 사람이 수행해야 한다.

3.3 기술검토와 감사

소프트웨어는 개발, 운영, 유지보수 단계에서 소프트웨어 품질보증 요건에 적합한가를 결정하기 위해 주기적으로 검토되고 감사를 받아야 한다. 엔지니어링 업무의 품질과 상태를 평가하고, 요구되는 기술문서의 작성과 품질기준에의 적합성을 보증하기 위해서 기술적인 검토와 감
사가 주기적으로 시행되어야 한다. 개발중의 소프트웨어의 검토(review)는 소프트웨어 V/V 그룹에 의해 사용되어지는 주된 방법으로 다음과 같이 분류된다.

. Software requirements review (SRR)
. Preliminary design review (PDR)
. Critical design review (CDR)
. Software verification review (SVR)
. Functional configuration audit (FCA)
. Physical configuration audit (PCA)
. In-process audit

3.3.1 기술검토

기술검토는 소프트웨어 품질을 확립하기 위한 도움을 주는것 이상의 여러 목적이 있다. 기술검토는 소프트웨어 개발자의 경험을 여러 사람과 함께 나눌 수 있는 기회를 제공한다.
소프트웨어 검토는 개발 프로젝트와 관련된 팀은 물론 각 개인의 기술적 능력을 향상시킬 수있는 장점이 있다.

검토 및 감사시에 의해 사용되는 절차서에는 참여자, 책임사항, 검토되어야할 문서의 종류 등이 기술되어야 하고 후속조치 사항과 그 책임사항이 나타나야 한다. 검토를 단계별로 분류하면 다음과 같다.

Software Requirements Review(SRR) 는 SRS(software reqirement specification)가 완성된 단계에서 요구사항 시방서의 적절성(adequacy), 기술적 타당성, 요구사항의 완전성 등을 보증하기 위한 것이다. 즉 SRR은 SRS가 소프트웨어의 운전 및 유지보수 단계에서 완벽하고, 검증가능하며 (verifiable), 일관성있고, 유지보수 및 수정이 가능하며 추적이 용이하고 사용하기 쉽게 작성될 것인지를 평가 한다. SRR에 참여하는 사람은 소프트웨어 설계요원, 검증요원, 소프트웨어 품질보증 요원, 시스템 엔지니어

링 요원, 사용자 및 운영자 등이다. SRR의 결과로서 식별된 결함사항과 그에 대한 시정조치의 계획과 일정 등이 문서에 포함되어 유지되어야 한다.

Preliminary Design Review(PDR) 는 기능시방서 단계에서 예비설계의 기술적 타당성을 평가하는 것으로서 첫째 사용된 설계기법에 대한 기술적 적절성을 평가하고, 둘째 SRS의 기능과 성능의 요구사항이 조화되어 설계되었는가를 체크하고, 셋째 소프트웨어와 하드웨어 및 사용자간의 인터페이스의 조화를 입증해야 한다.

Critical Design Review(CDR) 는 코딩 작업하기전에 상세설계의 기술적 타당성, 완벽성 및 정확성을 평가한다. CDR의 목적은 상세설계가 SRS의 요건을 만족하는가 하는 내용 즉, 하드웨어와 소프트웨어의 조화성(Compatibility)를 평가하고 제품설계의 기술, 코스트, 일정 등을 구하기 위해 SDD(Software Design Description)상에 나타난 상세설계의 옳고 그름을 평가하는 것이다. CDR은 소프트웨어 품질보증조직, 개발자 및 사용자가 참여하며 검토중 발견되는 결함사항과 이의 해결을 위한 계획 및 일정이 검토의 결과와 함께 문서로 유지되어야 하며, 수정된 SDD는 다음 단계인 시스템 구현 및 코딩단계의 지침으로 사용된다.

Software Verification Review(SVR) 은 완성된 SVVP를 평가하기 위해 행한다. SVVP는 요구시방서, 개념설계 및 상세설계가 진전됨에 따라 개발되는 것이므로, SVVP에 대한 중복적인
검토가 필요하다. SVR은 SVVP에 되어 있는 방법들이 적절하고, 소프트웨어에 대한 수긍할 만한 검증을 제공할 수 있는가를 보증하기 위한 것이다. SVR의 결과에 대한 기록에는 검토시 도출된 결점과 이의 해결에 대한 계획과 일정이 포함되어야 한다.

3.3.2 감 사

Functional Configuration Audit (기능감사)는 SRS의 모든 요구사항들이 적용되었는지를 입증하기 위하여 소프트웨어 양도 이전에 실시한다.
즉, 기능감사는 코드와 SRS에서 서술된 요구사항을 비교하여 코드가 요구사항을 만족했는지를 결정하기 위한 것이다. 기능 감사의 결과인 결함사항과 이에 대한 해결책과 일정이 문서로서 기록 유지되어야 하며, 결함사항이 제거되면 소프트웨어는 사용자에게 전달 될 수 있다.
Physical Configuration Audit (물리감사)는 컴퓨터 프로그램과 관련된 모든 기술문서들이 완전하고 일관성이 있으며 사용자가 사용하기에 적합한가를 결정하기 위해 행한다. 이의 대상으로서는 SRS, SDD 및 사용자를 위해 작성된 기타 공식문서 등 컴퓨터 프로그램과 관련된 기술문서가 이에 해당한다.

In-Process Audit (개발중 감사)는 Walk-throughs와 검사 등을 통하여 소프트웨어 제품의 일관성을 검증하고자 하는 것이다.

소프트웨어 품질보증 감사는 소프트웨어 품질보증 계획(SQA program)에서 규정된 절차서, 기준(standards), 방법들의 효과와 이의 이행상태를 평가하는 것으로서, 내부절차, 사업 SQA plans, configuration management 등이 소프트웨어 라이프사이클의 전단계에 걸쳐서 감사 되어져야 한다. 감사자는 다음 단계에 혼동을 줄 수 있는 소지가 있는 생략사항, 분명한 모순이나 항목은 물론 규정된 내용에의 적합성에 대한 각 다큐멘트의 형태를 점검해야 한다. 또한 필요한 모든 문서가 존재하는가를 검토하고 각 각의 품질이 어느 정도인가도 검토해야 한다. 소프트웨어 품질보증 감사 보고서는 사업 책임자에게 제출되어야 하며 이러한 감사가 설계, 코딩, 문서 작성시에 함께 수행될 경우 재작업이나 코스트의 초과를 초래하는 사태를 미연에 방지할 수 있는 것이다.

4. 소프트웨어 품질보증 프로그램의 수립 요건

앞 절에서 개발단계의 소프트웨어의 품질을 측정,평가하기 위한 방안에 대하여 간략히 고찰하였다. 소프트웨어의 품질을 시스템적으로 확보하기 위한 최적의 방안은 소프트웨어의 개발 초기부터 소프트웨어 품질보증프로그램을 수립하여, 이를 이행토록 하는 것이다.
본 장에서는 그러한 소프트웨어 품질보증 프로그램의 수립을 위한 제 요건을 제시하고자 한다.

첫째, 프로젝트 관리 (project management)의 강화.
프로젝트 관리는 SQA의 중요한 요소들이며 개발 및 이용단계를 포함한 소프트웨어의 전 라이프사이클에 수행되어져야 한다. 프로젝트관리는 SQA 기능의 관리, configuration management, standards의 확립, V/V, SQAP(Software Quality Assurance Plan)의 작성등을 포함한다. 프로젝트관리는 소프트웨어 개발의 예산을 결정하기 때문에 결국 소프트웨어의 품질을 좌우하게 된다. 품질이 좋지 않은 소프트웨어는 사용하기도 어렵고 비용이 많이 들기 때문에 소프트웨어 신뢰성과 같은 기타 사항들이 소프트웨어의 개발시부터 고려되어야 한다.

둘째, 소프트웨어 품질보증정책의 수립
소프트웨어 품질보증(SQA)은 소프트웨어 제품이 미리 설정된 요구사항과 일치하는가를 확인하기 위하여 전 라이프사이클에 대한 절차를 기술하고 기술된 방법을 적용 등으로 구성된다.
SQA의 기능의 목적은 소프트웨어 개발과정에 대한 통제, 교육계획 및 보고를 의미한다.
소프트웨어 품질보증의 정책은 SQA 계획의 내용을 이행하고, 적용하기 위해 필요하다. 경영자는 QA 요구사항들이 모든 엔지니어링 과정에 적용될 수 있는 것과 같이 소프트웨어 품질보증 역시 개발, 구현, 사용 및 소프트웨어의 관리등에도 적용될 수 있다는 것을 인식해야 한다. SQA의 기능은 프로젝트 관리의 의사결정을 하는 것이 아니라 기 확립된 standards와 절차서에 대한 적합 및 부적합 사항을 결정하는 것이며 프로젝트의 방향을 움직일 수 있는 힘이 있어야 한다. 실제로 적합한가, 아니면 부적합 한가를 결정하는 것이 SQA의 목적이라고 할 수 있다. SQA 조직은 SQA의 정책이 배타적으로 감시를 하기 위한 것이 아니라 조력하기 위한 것이라면 프로젝트 조직으로 부터 기꺼이 채택될 것이다.

셋째, SQA 관리 조직의 적절한 설정
소프트웨어 품질보증의 인력은 소프트웨어 시방, 소프트웨어 디자인 및 검증과 같은 소프트웨어 개발시의 기술적인 경험이 있어야 한다. SQA의 상급 기술진은 프로젝트의 행정관리 능력면에서 뛰어나야 한다. SQA의 인력은 구조화 프로그래밍, top-down 설계 및 구현, 검증기술과 같은 최신의 프로그래밍 기법에 능통해야 하며 소프트웨어 개발자들이 주장하는 개념이나 개발기법에 대하여 의사전달을 할 수 있을 뿐만 아니라 QA practices, regulations, standards에도 능통해야 한다.
SQA 조직은 서프트웨어의 개발조직과 독립된 위치에 있어야 하다. 또한 품질상의 문제를 도출하고, 그에 대한 해답을 제공하고 추천할 수 있도록 충분한 책임과 권한이 부여되어야 하며 조직상의 자유가 있어야 한다. 그러나 프로젝트의 규모가 작을 경우에는 조직의 비독립성으로 인한 위험을 무릅쓰고 SQA의 기능이 개발조직에 통합되어 운영될 수도 있다.

넷째, 소프트웨어 품질보증 마인드의 확산
소프트웨어 품질보증 계획의 실시에 앞서 SQA를 담당하고 있는 사람은 무슨 문제가 생기고, 왜 그것이 진행되었으며, 어떻게 그것이 조직에 영향을 끼치고, 기대되는 것이 정확히 무엇인지를 이해하고 있어야 한다.
소프트웨어의 개발, 유지 및 보수에 관련된 사람은 체계적인 엔지니어링 기법과 품질보증 계획이 소프트웨어의 개발 과정을 방해 한다기 보다는 도움을 줄 것이라는 확신을 갖고 있어야 한다.
소프트웨어 품질보증 업무는
- . 소프트웨어 품질보증 프로그램의 작성
- . 소프트웨어 품질보증의 정책, 절차서 및 기준의 개발
- . 소프트웨어의 인증과 검증

. SQA 업무를 수행하는 인원의 교육 및 훈련
. 다음 사항들에 대한 소프트웨어 품질보증 감사
  - 설계 (design)
  - Configuration management(CM)
  - 검증(testing)
  - Verification and Validation

등으로 이들 업무에 대한 조직의 책임사항이 명시되어야 한다. 여기서 CM(configuration management)은 전 라이프 사이클을 통하여 소프트웨어 제품을 식별하고 개정 수행상태를 기록, 보존하는데 사용되는 방법을 전자화하여 관리하는 것이다.

5. 결 론

원전의 설계에서 운전/운영에 이르기까지 이용되는 컴퓨터 소프트웨어의 품질확보를 위해서는 소프트웨어의 라이프사이클을 통하여 각 단계별 품질을 측정,계량화 할 수 있는 방법이 검토되고 개발되어져야 한다. 현재, 소프트웨어의 품질을 계량화하는 방안이 일부 소개되기도 하였으나, 아직 보편적으로 널리 쓰이지는 못하는 실정이며, 이에대한 연구개발이 지속적으로 이루어져야 한다.

소프트웨어의 품질확보를 위한 좋은 방안을 제시하면 다음과 같다. 4 장에서 설명한 바 와 같이 적절한 품질보증프로그램을 수립하여 소프트웨어의 전 개발단계에 QA principles을 적용
해야 하며, 이러한 SQA policy하에서 verification 및 validation을 수행하는 것이다. 이러한 관점에서 이제까지 논의한 내용을 결론으로 요약하면 다음과 같다.

첫째, 소프트웨어의 개발과정에 대한 품질평가를 위한 V/V는 제3자가 수행하도록 한다.

둘째, 소프트웨어 품질보증 감사는 소프트웨어 라이프사이클의 모든 단계에 걸쳐서 감사되어져야 한다.

셋째, 소프트웨어 품질보증의 정책은 SQA의 정책이 개발자를 배타적으로 감시를 하기 위한 것이 아니라 조력하기 위한 것이라는 인식이 되도록 수립되어야 한다.

네째, SQA의 인력은 구조화 프로그래밍, top-down 설계 및 구현, 검증기술과 같은 최신의 프로그래밍 기법에 능통해야 하며 소프트웨어 개발자들이 주장하는 개념이나 개발기법에 대하여 의사소통을 할 수 있을 뿐만 아니라 QA practices, regulations, standards에도 능통해야 한다.

다섯째, Hardware의 경우도 그렇듯이 S/W에 대한 review 및 audit에도 개발초기부터 충분한 노력을 기울여야 한다.

여섯째, 소프트웨어 품질보증 조직의 독립성이 인정되어야 하고 각 조직의 업무 및 책임사항이 소프트웨어 라이프 사이클과 연결지어 정의되어야 한다.

참고문헌

1] 과기처 고시 " 프로그램 품질보증 기준" ('91.4.9 , 제 91-3 호)
2] Software Quality Assurance Related to the Safety of Nuclear Power Plants. Volume I,II Nuclear Training Center/ KAERI, 1994
3] Handbook of Software Quality Assurance Techniques Applicable to the Nuclear Industry, NUREG/CR-4640
4] Quality Assurance Requirements of Computer Software for Nuclear Facility Applications, 1992 ASME NQA-2 part 2.7
5] IAEA Technical Report Series No. 282 - Manual on Quality Assurance for Computer Software Related to the Safety of Nuclear Power Plants.