항공용 소프트웨어의 설계 · 인증 고려사항

이백준*, 김성겸**

Software Considerations in Airborne System Design and Certification

Baeck-Jun Yi*, Seung-Kyem Kim**

Abstract

It is booming to use computer owing to the information society, and embedded software application have grown in airborne systems and equipment. So this introduces airborne software classification, software life cycle, activities to achieve objectives and software considerations in design and certification.

초 록

고도 정보화 사회라고 일컫는 오늘날 컴퓨터는 거의 모든 분야에서 사용되고 있으며, 항공 산업에서도 항공기 및 관련 장비의 개발 과정에서 내장형 소프트웨어가 차지하는 비중이 급증하고 있다. 이에 항공용 소프트웨어에 대한 개괄적인 사항을 알아보고, 항공용 소프트웨어의 분류, 소프트웨어 라이프사이클, 각 목표를 달성하기 위한 활동과 설계고려사항을 알아본다.

키워드 : 항공 시스템(airborne system), 소프트웨어(software), 고려사항(consideration)

1. 서 론

현대의 시스템은 하드웨어와 소프트웨어가 결합된 형태로 개발되지만 사실상 소프트웨어가 시스템 전체의 품질을 좌우하는 핵심요소, 즉 성능의 결정적인 요소이다. 이러한 기술적 추세에 따라 항공 산업에서도 항공기 및 관련 장비의 개발 과정에서 내장형 소프트웨어가 차지하는 비중이 급증하고 있다.

항공용 소프트웨어란 항공기 및 관련 장비의 내부에 장착되어 항공기 및 관련 장비를 구동하기 위해 사용되는 소프트웨어로 대부분 특정 목적을 위해 사용되며, 수행되는 기능이 고정적이라 범용성이 없다. 또한, 비행과 관련하여 반응시간에 종속적인 실시간 처리가 요구되며, 고온이나 다습한 환경 등 극한 상황에서도 계속 동작하도록 요구되는 특징을 가지고 있으므로, 이의 오동작은 시스템의 성능을 좌우함은 물론 항공기의 신뢰성과 인간의 안전에도 큰 영향을 주고 있어 항공용 소프트웨어의 품질과 인증이 관심사로 대두되고 있다.

* 제품보증그룹/ybj@kari.re.kr
** 항공인증그룹/skykim@kari.re.kr

2. 시스템의 고장과 항공용 소프트웨어의 분류

2.1 시스템의 고장조건

시스템의 고장조건에 대한 분류는 항공기 및 항공기 탑승자에 미치는 고장의 가혹성을 결정함으로서 이루어지는데, 소프트웨어 상의 오류는 고장 상태의 한 원인으로 작용할 수 있는 결함을 유발할 수 있으므로 안전한 운항을 위해 요구되는 소프트웨어의 무결성 수준은 시스템 고장 상태와 관계가 있다. 시스템의 고장은 Catastrophic, Hazardous, Major, Minor, No Effect 등 5가지로 분류한다.

- ㅇ Catastrophic : 계속적인 비행 또는 착륙이 안전하게 이루어지지 못하게 되는 고장
- ㅇ Hazardous : 불리한 운항 상태에 대처하기 위한 승무원의 능력 또는 항공기의 성능을 다음과 같은 정도까지 감소시킬 수 있는 고장
  - 안전여유 또는 기능성의 많은(large) 감소
  - 물리적인 피로 또는 증가된 작업부담으로 인해 비행승무원이 해당 임무를 정확하게 또는 완벽하게 수행하는 것이 어려운 상태
  - 일부 탑승자가 중상 또는 사망에 이를 수 있는 부상을 입는 경우를 포함하여 탑승자에게 유해한 영향이 가해지는 상태
- ㅇ Major : 불리한 운항 상태에 대처하기 위한 승무원의 능력 또는 항공기의 성능을 다음과 같은 정도까지 감소시킬 수 있는 고장
  - 안전여유 또는 기능성의 현저한 (significant) 감소
  - 승무원 작업부담 또는 업무효율을 저해하는 상태의 현저한 증가
  - 부상 가능성을 포함하여 탑승객이 불편을 느낄 수 있는 상태
- ㅇ Minor : 다음과 같이 항공기의 안전성이 현저한 수준으로 까지는 감소하지 않으며 승무원들이 능력 범위에서 조치를 취할 수 있는 고장
  - 안전여유 또는 기능성의 경미한 감소
  - 통상적인 비행계획의 변경 등과 같이 승무원 작업부담의 경미한 감소
  - 탑승객이 어느 정도의 불편함을 느낄 수 있는 상태
- ㅇ No Effect : 항공기의 운항 성능 또는 승무원의 작업부담에 영향을 미치지 않는 고장

2.2 항공용 소프트웨어의 분류

1980년대에 접어들어 항공기와 엔진에 사용되는 항공용 시스템 및 장비에서 소프트웨어를 사용하는 경우가 급격히 증가함에 따라 감항요구조건을 충족시킬 수 있도록 산업계가 수용할 수 있는 지침을 마련하기 위해 RTCA(Radio Technical Commission for Aeronautics, Inc.)에서 RTCA DO-178을 작성하였는데, 이 규격에서 항공용 소프트웨어를 다음과 같이 분류하였다.

표 1. 소프트웨어의 분류

구 분	A급 (Level A)	B급 (Level B)	C급 (Level C)	D급 (Level D)	E급 (Level E)
비정상 작동에 따른 고장조건	치명 고장 (Catastrophic)	위협 고장 (Hazardous)	중 고장 (Major)	경 고장 (Minor)	영향 없음 (No effect)

2.2.1 소프트웨어 레벨의 결정

소프트웨어의 레벨은 시스템 안전성평가 프로세스를 통해 결정한 고장상태에 대한 소프트웨어의 기여정도에 근거하여 결정하는데, 소프트웨어의 레벨은 인증요건에 대한 적합성을 입증하기 위해 요구되는 노력의 수준이 고장조건의 등급에 따라 달라질 수 있다.
소프트웨어 컴포넌트의 비정상적인 작동으로 인하여 1개 이상의 고장상태가 초래되는 경우에는

해당 컴포넌트의 가장 심각한 고장 조건 등급을 통해 해당 컴포넌트에 대한 소프트웨어 레벨을 결정한다. 그리고 소프트웨어 컴포넌트 중 어느 하나만 비정상적으로 작동하더라도 고장상태에 이르게 되는 직렬형(serial implementation)의 경우 모든 컴포넌트는 해당 시스템 기능의 가장 심각한 고장 상태 카테고리에 상응하는 소프트웨어 레벨을 갖게 된다.
소프트웨어 레벨에 맞는 소프트웨어 개발은 해당 소프트웨어의 고장률을 지정하는 것은 아니며 따라서 시스템 안전성 평가 프로세스에서 소프트웨어 레벨 또는 소프트웨어 레벨에 근거한 소프트웨어 신뢰성 등급을 사용할 수는 없다.

3. 항공용 소프트웨어의 개발

3.1 시스템과 소프트웨어간의 교류

소프트웨어 개발과정에서 시스템 안전성 평가 프로세스와 시스템설계 프로세스는 상호의존적(interdependence)이므로 시스템 라이프사이클 프로세스와 소프트웨어 라이프사이클 프로세스 간 정보 흐름은 반복적으로 이루어진다. 그림 1은 안전성과 관련하여 시스템 라이프사이클 프로세스와 소프트웨어 라이프사이클 프로세스 간에 이루어지는 정보 흐름을 나타낸 것으로, 시스템 안전성 평가 프로세스에서는 시스템의 고장조건을 결정하고 이를 분류하게 되며 시스템 안전성 평가 프로세스 내에서 시스템 설계에 대한 분석을 통해 고장조건에 대해 요구되는 내성(immunity), 고장조건에 대한 반응을 규정한 안전성 요구조건을 규정한다.
이러한 요구조건은 하드웨어 및 소프트웨어에 대하여 결함에 의한 영향을 미리 배제하거나 제한하도록 정의되며 결함 감지 및 결함 내포정도를 제시할 수 있고, 하드웨어 설계 프로세스와 소프트웨어 개발 프로세스 중에 결정이 이루어지기 때문에 시스템 안전성 평가 프로세스를 통해 시스템 설계를 분석함으로서 안전성 관련 요건을 만족하는지 여부를 검증하게 된다.

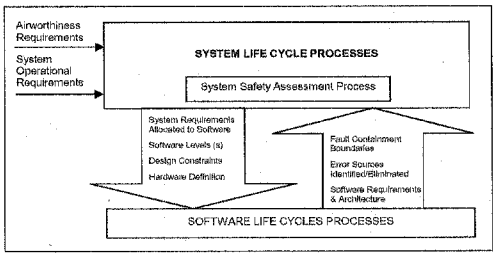

그림 1. 시스템과 소프트웨어사이의 정보흐름

또한, 시스템 안전성평가 프로세스에서는 소프트웨어 라이프사이클 프로세스를 통해 제공되는 정보를 사용하여 소프트웨어 설계와 그 실행(implementation)이 시스템 안전성에 미치는 영향을 결정함하고, 제공 정보에는 고장 내포 범위(fault containment boundary), 소프트웨어 요구조건, 소프트웨어 아키텍처 및 소프트웨어 아키텍처 또는 소프트웨어 설계 프로세스에서 사용되는 도구 또는 기타의 방법을 통해 감지되거나 제거되었을 수 있는 에러 유발원 등이 포함되며 시스템 요구조건과 소프트웨어 설계 데이터 사이의 추적성은 시스템 안전성 평가 프로세스에 중요한 요소이다. 그리고 소프트웨어를 변경(modification)하는 경우 시스템 안전성에 영향을 미칠 수 있으므로 시스템 안전성평가 프로세스를 통한 평가가 필요하다.

3.2 소프트웨어 라이프사이클

소프트웨어 개발에 관련된 라이프사이클 프로세스는 다음 세 가지로 요약할 수 있다.

- Planning Process
  프로젝트에 대하여 소프트웨어 개발 업무와 무결성(integral) 프로세스를 정의하고 조정하는 소프트웨어 계획프로세스
- Development Process
  소프트웨어 제품을 생산하는 소프트웨어 개발 프로세스임. 이러한 프로세스로는 소프트웨어 요구조건 프로세스, 소프트웨어 설계 프로세스, 소프트웨어 코딩 프로세스 및 통합프로세스가 있음

○ Integral Process

소프트웨어 라이프 사이클 프로세스와 그 결과물의 정확성, 관리(control) 및 신뢰성을 보증하기 위한 무결성 프로세스임. 무결성 프로세스로는 소프트웨어 검증 프로세스, 소프트웨어 형상관리 프로세스, 소프트웨어 품질 보증 프로세스 및 인증 협의(liaison) 프로세스가 있음

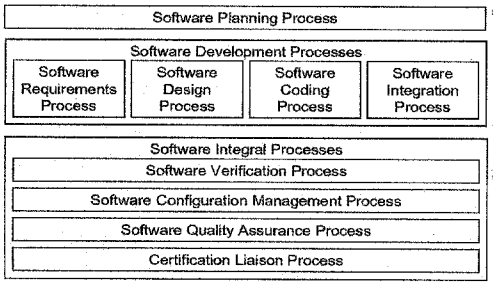

그림 2. 소프트웨어 라이프사이클 프로세스의 구성

3.3 항공용 소프트웨어 시스템 요구조건

안전성 요구조건은 소프트웨어 라이프사이클 프로세스에 대한 입력이 되는 시스템 요구조건의 일부를 구성하므로, 안전성 요구조건이 소프트웨어 라이프사이클 전 과정에 걸쳐 충실히 이행되도록 하기 위해서는 시스템 요구조건은 아래를 포함하거나 참조하여야 한다.

- 시스템 명세서(description)와 하드웨어 정의(definition)
- 인증요구조건(FAR, JAR, Advisory Circular 등)
- 소프트웨어에 할당된 시스템 요구조건 (기능 요구조건, 성능 요구조건, 안전성 관련 요구조건 등)
- 소프트웨어 레벨 및 레벨결정에 대한 근거자료, 실패조건 및 그에 따른 분류, 소프트웨어에 할당된 관련 기능
- 안전성 확보전략 및 설계 제한사항 (partitioning, dissimilarity, redundancy 또는 안전성 모니터링 등)

시스템 검증활동(verification)을 용이하게 하기 위해 시스템 라이프사이클 프로세스에서 소프트웨어 라이프사이클 프로세스에 대한 요구조건을 규정할 수도 있다.

	Objective		Applicability by SW level				Output		Control category by SW level			
	Description	Ref.	A	B	C	D	Description	Ref.	A	B	C	D
1	Software development and integral processes activities are defined.	4.1a 4.3	○	○	○	○	Plan for Software Aspects of Certification	11.1	①	①	②	②
							Software Development Plan	11.2	①	①	②	②
							Software Verification Plan	11.3	①	①	②	②
							SCM Plan	11.4	①	①	②	②
							SQA Plan	11.5	①	①	②	②
2	Transition criteria, inter-relationships and sequencing among processes are defined.	4.1b 4.3	○	○	○							
3	Software life cycle environment is defined.	4.1c	○	○	○							
4	Additional considerations are addressed.	4.1d	○	○	○	○						
5	Software development standards are defined.	4.1e	○	○	○		SW Requirement Standards	11.8	①	①	②	
							SW Design Standards	11.7	①	①	②	
							SW Code Standards	11.8	①	①	②	
6	Software plans comply with this document.	4.1f 4.6	○	○	○		SQA Records	11.15	②	②	②	
							Software Verification Results	11.14	②	②	②	
7	Software plans are coordinated.	4.1g 4.6	○	○	○		SQA Records	11.15	②	②	②	
							Software Verification Results	11.14	②	②	②	

LEGEND
- ● The objective should be satisfied with independence.
- ○ The objective should be satisfied.
- Blank Satisfaction of objective is at applicant's discretion.
- ① Data satisfies the objectives of Control Category 1 (CC1).
- ② Data satisfies the objectives of Control Category 2 (CC2).

그림 3. 소프트웨어 레벨에 따른 목표

3.4 소프트웨어 설계 고려사항

3.4.1 소프트웨어 개발 환경

소프트웨어 개발 환경은 고품질의 소프트웨어 제작을 위해 매우 중요한 요소로, 소프트웨어 개발 환경은 또한 항공용 소프트웨어 제작에 몇 가지 방식으로 악영향을 미칠 수 있다. 예를 들어, 컴파일러는 오염된 출력을 생성하여 에러를 유발할 수 있고, 링커의 경우 메모리 할당 오류의 발생을 감지하지 못할 수도 있다.

또한, 컴파일러의 적합성에 대한 고려가 이루어져야 하는데 이것이 유효성을 지니기 위해서는 소프트웨어 검증 프로세스 활동에서 프로그래밍 언어와 컴파일러의 특성을 고려할 필요가 있다. 소프트웨어 계획수립 프로세스에서는 프로그래밍 언어를 선정하고 검증 계획을 수립할 때에 이러한 특성을 고려한다.

3.4.2 시스템 아키텍처

시스템 안전성평가 프로세스에서는 아키텍처 측면에서의 설계 결정사항을 고려하여 이러한 결정사항이 소프트웨어의 레벨 또는 기능성에 영향을 미치는지 여부를 결정한다. 오류의 영향을 제한하거나 오류를 감지하여 시스템이 적절한 대응을 통해 그 오류를 봉쇄하도록 하는 아키텍처에는 다음과 같은 기법을 사용한다.

- 분할(Partitioning)
  기능적으로 독립적인 소프트웨어 컴포넌트들을 분리하여 결함을 억제하거나 고립시키는 기술로서 잠재적으로는 소프트웨어 검증 프로세스에 소요되는 노력을 경감시킬 수 있음
- 다중버전 소프트웨어(Multiple-Version Dissimilar Software)
  Multiple-version dissimilar 소프트웨어는 컴포넌트들에 발생할 수 있는 공통의 오류 원인을 회피할 수 있도록 동일한 기능을 제공하는 두개 이상의 소프트웨어 컴포넌트를 개발하는 시스템 설계기법의 일종
- 안전성 모니터링(Safety Monitoring)
  안전성 모니터링은 고장 상태를 야기할 수 있는 고장에 대하여 직접적으로 기능을 모니터링하여 특정 고장 상태를 방지는 방법으로 모니터링 기능은 하드웨어, 소프트웨어 또는 하드웨어/소프트웨어 조합에서 수행될 수 있음

3.4.3 소프트웨어 시험 환경

소프트웨어 시험환경 계획 수립의 목적은 통합 프로세스의 출력을 시험하는데 사용할 방법, 도구, 절차 및 하드웨어를 규정하는데 있다. 시험은 목표 컴퓨터, 목표 컴퓨터 에뮬레이터 또는 호스트 컴퓨터 시뮬레이터를 사용하여 수행될 수 있으므로, 이러한 특성을 고려한다.

3.4.4 소프트웨어 검증시스템

시스템 요구조건은 시스템 운용 요구조건과 시스템 안전성 평가 프로세스를 통해 도출된 안전 관련 요구조건을 기반으로 개발되는데, 항공용 소프트웨어에 대한 시스템 요구조건으로부터 다음과 같은 두 가지의 소프트웨어 특성이 수립된다.

- 소프트웨어는 시스템 요구조건에 의해 정의된 바에 따라 규정된 기능을 수행함
- 소프트웨어는 시스템 안전성 평가 프로세스에서 규정한 비정상작동이 발생하지 않고, 이 현상을 제거하기 위한 추가적인 시스템 요구조건을 구성하여야 함.

이러한 시스템 요구조건은 소프트웨어 상위레벨 요구조건으로 개발되어야 하며 소프트웨어 검증 프로세스 활동을 통해 검증되어야 하는데, 검증 프로세스는 그림 4와 같이 수행된다.

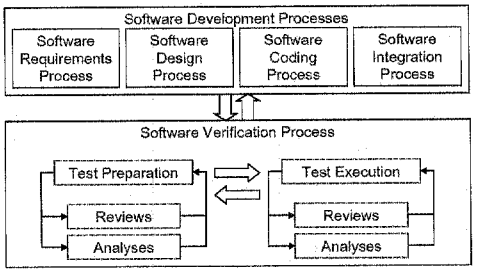

그림 4. 소프트웨어 개발 및 검증 절차

시스템 검증을 통해 중요한 코드 구조 커버리지를 얻을 수 있는데, 소프트웨어 검증에 따라 기술된 다양한 테스트 활동의 커버리지 목표(coverage objectives)를 달성하기 위해 시스템 검증 테스트 범위 분석을 이용할 수 있다.
그리고, 소프트웨어 개발 도구에 대한 검증 기준은 다음과 같다.

첫째, 소프트웨어 개발 도구를 검증해야 하는 경우, 도구의 소프트웨어 개발 프로세스는 항공용 소프트웨어의 개발 프로세스의 목표와 동일한 목표를 충족해야 한다.

둘째, 도구(tool)에 배정된 소프트웨어 레벨은 도구를 통해 제작되는 항공용 소프트웨어의 레벨과 동일해야 한다. 단, 신청인이 도구의 소프트웨어 레벨을 낮추는 것이 정당함을 인증당국에 입증한 경우는 예외로 한다.

3.5 소프트웨어 인증

인증 당국은 소프트웨어 관점의 인증 계획이 인증기준을 충족할 수 있도록 합의된 입증 방안과 무결성이 완전한지에 대해 평가하는데, 인증당국은 신청인이 제안한 소프트웨어 레벨이 시스템 안전성 평가 프로세스의 출력과 여타의 시스템 라이프 사이클 데이터와 일치함을 확인한다. 적합성을 결정하기 위해 소프트웨어의 경우, 이는 소프트웨어 완성 요약서와 적합성 입증 자료를 검토함으로서 이루어진다. 인증에 필요한 자료는 다음과 같다.

- 각종 계획서
  (인증, 개발, 검증, 형상관리, 품질보증 등)
- 각종 표준
  (요구조건 표준, 설계 표준, 코드 표준 등)
- 소프트웨어 요구조건 데이터 및 설계 설명서
- 소스 코드, 실행가능한 오브젝트 코드
- 소프트웨어 검증 사례와 절차 및 결과
- 각종 색인(소프트웨어 라이프사이클 환경 형상, 소프트웨어 형상 색인)
- 문제 보고서 및 형상관리 기록, 품질보증 기록
- 소프트웨어 완성 요약서

4. 결 론

항공기 시스템 및 장비에 사용되는 소프트웨어가 안전성과 관련하여 감항기술기준에 의해 요구되는 신뢰 수준으로 설계·개발될 수 있도록 항공용 소프트웨어에 대한 개괄적인 사항을 알아보고, 항공용 소프트웨어의 분류, 소프트웨어 라이프사이클, 각 목표를 달성하기 위한 활동과 설계고려사항과 인증 프로세스에 대한 이해를 돕기 위해 시스템 라이프사이클과 소프트웨어 라이프사이클간의 관계를 살펴보았다. 이들은 항공용 소프트웨어의 설계단계에서의 설계 요구조건, 개발·검증단계에서의 소프트웨어 합치성 검토, 인증단계에서의 적합성 결정 등에 활용될 수 있을 것으로 기대된다.

참 고 문 헌

1. DO-178B, "Software Consideration in Airborne Systems and Equipment Certification", RTCA, 1992.
2. AC20-115B, "Radio Technical Commission for Aeronautic, Inc. Document RTCA/DO-178B", FAA, 1993.
3. AC25.1309-1A, "System Design and Analysis", FAA, 1988.
4. 소프트웨어공학, 홍릉과학출판사, 1989.