JANT 한국항행학회논문지
Journal of Advanced Navigation Technology

항공 통신 기술

J. Adv. Navig. Technol. 23(5): 345-351, Oct. 2019

항공용 SIL에 적용 가능한 DEVS 형식론 기반의 시뮬레이션 환경 개발

Development of Real Time Simulation Environment Based on DEVS Formalism Applicable to Avionics System Integration Laboratory

서 민 기* · **신 주 철** · **백 경 훈** · **김 성 우**
LIG 넥스원 항공연구소

Min-gi Seo* · **Ju-chul Shin** · **Gyong-hoon Baek** · **Seong-woo Kim**
Avionics R&D Lab, LIG Nex1, Daejeon 34115, Korea

[요 약]

항공용 SIL은 항공전자시스템의 통합 및 검증에 사용되는 통합시험환경이다. 최근에는 항공전자시스템의 요구도 검증을 시스템 통합 측면에서부터 충분히 고려하기 위하여 항공용 SIL 분야의 개발 참여를 비행체 요구분석 단계부터로 앞당기고 있으며, 비행체의 체계종합 일정에 영향을 주지 않도록 항공용 SIL의 개발 비용 및 기간을 최소화하기 위한 노력을 꾸준히 진행하고 있는 추세이다. 본 논문에서는 항전체계 검증에 사용되는 항공용 SIL의 모델링 방법 표준화를 통한 개발기간/비용 단축 및 유지보수성 증대를 위하여 항공용 SIL에 적용 가능한 모델링 형식론 기반의 항공용 시뮬레이션 모델 프레임워크(ASMF)를 제안한다.

[Abstract]

Avionics System Integration Laboratory is an integrated test environment for the integration and the verification of avionics systems. Recently, in order to fully consider the requirements verification of avionics system from the aspect of the entire system integration, the participation in the development of the SIL field is advanced from the requirement analysis of the aircraft. Efforts are being made to minimize the cost and the period of development of a SIL so that it does not affect the overall schedule of the aircraft development. We propose the avionics simulation model framework (ASMF) based on the modeling formalism applicable to SIL in order to reduce development period/cost and increase maintenance by standardizing the modeling methods of SIL.

Key word : ASMF(avionics simulation model framework), SIL(system integration laboratory), DEVS(discrete event system specification).

https://doi.org/10.12673/jant.2019.23.5.345

Received 29 August 2019; **Revised** 4 October 2019
Accepted (Publication) 28 October 2019 (30 October 2019)

***Corresponding Author; Min-gi Seo**

Tel: +82-42-718-3565
E-mail: mingi.seo@lignex1.com

Ⅰ. 서 론

전자기술의 발달로 인해 항공기에서의 비중이 증대되고 있는 항공전자체계(이하 항전체계)에 높은 신뢰성이 요구됨에 따라 검증이 중요해지고 있다[1],[2]. 항전체계의 통합 및 검증환경인 통합시험환경 (SIL; system integration laboratory)은 항전체계의 정상동작뿐만 아니라 장비로는 시험이 어려운 비정상 동작을 모델을 통해 모의할 수 있다[1]. 유/무인 항공기 개발 시 항공기에 장착 전 SIL에서 항전체계를 검증하는 이유는 비용이 가장 크다. 항공기 개발에서 고장탐구에 소요되는 비용은 개발 단계가 진행될 때마다 이전 단계에 비해 약 10배가량 증가되는 것으로 보고되고 있다[2]. 즉, SIL은 항공기 개발 비용을 효과적으로 절감시키기 위한 필수 요소이다.

SIL에 요구되는 개발 특성은 SIL의 조기제작과 수정의 효율성이다. SIL이 없다면 항전체계 검증은 검증 대상인 모든 구성품들의 제작 완료 이후부터 진행할 수 있다. SIL을 조기에 항전체계에서 사용할 수 있다면 구성품 제작 완료 전부터 항전체계 설계에 대한 검증을 진행할 수 있다. 즉, SIL의 개발 기간은 최소화되어야 한다. 다만 항전체계 설계 확정 이후에도 통합시험의 결과 등의 사유로 인터페이스와 운용개념 변경이 빈번하게 발생 될 수 있다. SIL의 개발 범위와 수준이 항전설계와 밀접한 관계를 가지므로 SIL의 수정은 효율적이어야 한다.

해외 선진 업체인 Northrop Grumman사는 항공용 SIL의 재사용을 위해 SIL의 일반화 개념을 적용한 GSIL (generic SIL)을 개발하여 시뮬레이션 프레임워크 기반의 모델을 개발하였다. GSIL 이후엔 가상화 개념이 도입된 VSIL(virtual SIL)을 개발하여 SIL의 개발 생명 주기(development life cycle)를 더 감소시키기 위해 노력하고 있다. 해외 선진사들은 오랜 노하우가 포함된 이 기술의 기술이전을 제한하고 있다[3],[4].

국내 사례를 살펴보면, 항공기 개발 사업 별로 SIL의 하드웨어/소프트웨어 환경 및 요구도가 결정되었다. 모델은 SIL HW/SW 환경 하 In-house 방식으로 개발되었고, 짧은 개발 일정 내 진행하다보니 타 개발 사업에 적용할 수 있는 이식성을 고려하기 어려웠었다. 점차 국내의 유무인기 사업이 진행되어 SIL 개발 노하우가 축적되면서 생산성/품질 향상 필요성이 대두되었다.

본 연구를 통해 SIL의 개발 생산성 향상을 위한 개선 방안을 제안하고자 한다. 모델 표준화 및 이식성에 초점을 둔 항공용 SIL의 비행체 탑재장비 모델 프레임워크 (ASMF; avionics simulation model framework)의 정의, ASMF의 주요 구성 요소, SIL 개발자 관점에서의 운용 개념 순으로 내용을 구성하였다.

Ⅱ. ASMF 정의

ASMF는 항공용 SIL에 적용 가능한 비행체 탑재장비 모델의 공용 프레임워크로, ASMF를 기반으로 SIL을 개발하는 방식과 기존 SIL 개발방식을 비교하면 다음과 같다.

표 1. 전통적인 방법과 ASMF 기반의 SIL 개발 방법 비교
Table 1. Comparison of conventional methods and ASMF-based SIL development methods.

Category	Conventional SIL	ASMF-based SIL development method
Engineer skill level	Understanding of avionics + Intermediate level programming skills	Understanding of avionics + Beginner level programming skills
Engineering activity	- Design(UML, etc.) - Coding	- Modeling
Productivity	Lower than ASMF method - Lack of continuity between design and coding - Reusability is programmer dependent	Higher than traditional method - Strong continuity between design and coding - Reuseability is part of the modeling approach
Quality	Large deviations between outputs - Depends on programmer - Tested after coding - Difficult for the client to understand outputs	Small deviations between outputs - Uniform quality regardless of programmer - Test from the beginning of the design - Easy for the client to understand outputs
Maintainability	Lower than ASMF method - Depends on programmer	Higher than traditional method - Uniform quality regardless of programmer

기존 SIL과 ASMF는 개발자에 요구되는 역량이 다르다. 기존 SIL 모델 개발자는 프로그래밍 지식을 겸비하고 도메인 영역에 해당하는 모델링 대상 항전장비 기능/성능에 대한 이해를 습득해야 개발할 수 있었다. 반면 ASMF 기반 개발에서는 항전장비에 대한 이해가 높은 도메인 전문가를 위주로 SIL 모델을 개발할 수 있다. 이러한 차이는 ASMF가 코드 개발 중심에서 탈피하여 요구사항을 모델로 생성하는 과정에 중점을 둔 개발방법론인 모델 기반 개발(MDD; model driven development)의 성향을 갖기 때문이다. 모델 기반 개발에서는 소프트웨어 프로그래밍 요소를 최소화하고 도메인 기술에 집중하여, 급변하는 사용자의 요구사항 변화를 소프트웨어에 정확히 반영하는 것을 중요하게 여긴다[5].

ASMF와 기존 SIL 간 개발 프로세스 차이가 발생되는 부분은 설계와 구현이다. 기존 SIL에서의 설계는 도구를 사용하여 UML(unified modeling language)로 작성하였고, 구현은 개발 언어의 제약으로 코드 자동 생성 기능 없이 개발자가 UML을 참조하여 모델을 구동할 플랫폼 환경 하에 수행하였다. 빈번한 요구사항 변경 등의 환경에서 설계에 해당하는 UML과 소스 코드 간의 관계 무결성을 개발자가 보장하는 것은 쉽지 않다. 일반적으로 설계-구현 간 많은 불일치가 발생되어 개발기간이 지연되

는 등의 문제가 있으며, 이를 해결하기 위해 불일치 요소 검출 도구 등의 연구가 진행된 바 있다.[8] 반면 ASMF는 설계와 구현 부분이 일원화 되어 있다. 개발자가 IDE를 이용하여 블록도와 상태도를 작성하는 활동이 설계이며, 이 설계 자료상에서 필요한 조건과 값 등을 기입하면 소스 코드 자동 생성 기능으로 구현이 완료된다. 이 설계/구현 일원화 특성은 개발 생산성 및 유지보수성과 관련이 깊다.

재활용성은 개발 생산성과 밀접한 관계를 갖는 특성이다. 기존 SIL 개발에서는 개발자가 의도적으로 재활용성을 고려하여 개발한 후, 이를 사용해야 개발 생산성이 향상되었다. 즉, 재활용성 확보가 개발자에 의존적이었다. ASMF는 개발 방식이 재활용성을 내포하고 있다. 작성해야 하는 블록도 및 상태도가 모듈 단위로 재사용/재활용될 수 있기 때문이며, ASMF에서는 장비들의 공통 기능을 구현한 기본 모델을 제공함으로써 SIL 개발자에게 더 높은 개발 생산성을 보장할 계획이다.

기존 SIL에서는 개발자 역량과 품질을 분리하기 어려웠다. 기능 시험도 구현 후에 진행이 가능하고, 사용자의 요구사항 확인(Verify)에서도 설계와 구현 이원화로 인해 논쟁이 생길 소지가 있었다. ASMF는 코드를 자동 생성해주므로 기존 SIL에 비해서 개발자 기술 수준에 독립적이라고 할 수 있다. 설계와 구현이 일원화되어 있기 때문에 설계 직후 생성된 코드로 테스트를 해볼 수 있으니 설계 초기부터 테스트를 적용해봄으로써 품질 향상뿐만 아니라 향후 발생될 고장 탐구에 투입될 비용을 절감하는 효과도 볼 수 있다. 또한 설계/구현 일원화 특성으로 사용자의 요구사항 확인에 더 원활한 협의가 가능하다[5].

ASMF는 국방 분야에서 시스템 분석 및 훈련 도구로 널리 사용되고 있는 DEVS (discrete event system specification) 형식론을 적용하였다. DEVS 형식론은 상태 변수가 시간 축의 비연속적인 일련의 시점에서만 변화하는 시스템에 적합하다. SIL 관점에서의 항전장비 모델링은 ICD (interface control documents) 기반의 통신 시점 중심으로 시스템이 변화되므로 앞서 서술한 특성과 일치한다. 또한 DEVS는 각 개별 하부 시스템을 컴포넌트화하고 이를 조립하여 상위체계로 표현하기 때문에 모듈화 특성을 내포하고 있으며, 절차 관점으로 시스템의 행동을 구현했던 기존 방식을 객체지향 관점의 모델링 방식으로 전환하는데 도움을 받을 수 있다[6].

ASMF의 주요 구성 요소는 3가지이다. SIL 모델 개발자가 프레임워크 기반의 모델을 DEVS 형식론으로 개발할 수 있는 도구인 통합개발환경 (IDE; integrated development environment), 항전체계를 구성하는 항전장비들을 모의하는 항공용 시뮬레이션 모델 (ASM; avionics simulation model), 모델이 구동될 실시간 시뮬레이션 환경 (RTSE; real-time simulation environment)의 하드웨어와 운영체제에 종속되지 않고 독립적으로 ASM을 운용하고, 통신 인터페이스와의 연동을 관리하는 항공용 시뮬레이션 프레임워크 (ASF; avionics simulation framework)이다. 부수적으로는 ASMF를 검증하거나, ASMF에 비행모의 또는 표적 등의 센서 데이터를 제공하기 위한 외부 환경 (OTW&E; out the window & environment)을 추가할 수 있다.

Ⅲ. ASMF 구성

3-1 통합개발환경(IDE)

IDE의 주요 기능은 ASM 개발 기능과 ASM의 데이터를 제어 또는 모니터링 할 수 있는 패널(ASMP, avionics simulation model panel) 개발 기능이다. 부가적으로 RTSE와 연동하여 시뮬레이션을 운용할 수 있는 기능과 모델을 라이브러리로 변환하여 관리하는 기능 등이 있다.

IDE는 DEVS 형식론 기반으로 블록도/상태도 중심으로 표현하여 ASM을 개발한다. IDE의 상태도 및 블록도의 표현 대상은 DEVS 형식론의 원자 모델과 결합 모델이다. ASM의 통신 인터페이스의 입출력을 위해 DEVS 형식론에는 없는 연동 블록 개념을 도입하여 개발자가 정의할 수 있도록 하였다.

ASM의 데이터를 제어 또는 모니터링 할 수 있는 ASMP는 ASM 운용 측면에서의 활용 빈도가 높은데, 시뮬레이션 실행 중 ICD 데이터 및 ASM 내부 데이터를 시현 및 설정할 수 있기 때문이다. 시현은 에디트 박스, 라벨 등의 GUI 컨트롤 요소와 시현하고자 하는 데이터를 연결하는 것으로 가능하다. ASM의 고장 모의 및 성능 제어를 위한 ASM 데이터 설정은 버튼, 체크박스 등의 GUI 컨트롤 요소와 연결하여 설정할 수 있도록 하였다. ASMP는 OFP 및 항전체계 검증 차원에서 필요한 데이터를 모니터링 및 제어하고자 하는 필요성을 즉각 수용하여 제공할 수 있는 즉시성과 SIL 운용자가 쉽게 사용할 수 있는 편의성을 갖추는 것이 가장 중요하다. 본 연구에서는 ASMP와 시뮬레이션 실행을 각기 분리하여 즉시성을 확보하였고, 요구사항 수집, 관련 개발자의 개발 경험 등을 활용하여 편의성을 확보하고 있다.

3-2 항공용 시뮬레이션 모델(ASM)

ASM은 기본 모델, 확장 모델, ASMP로 구성된다. 기본 모델은 ASM에 필요한 장비의 필수 기능 및 임무컴퓨터(MC; mission computer) 연동에 필요한 탑재연동 기능을 SIL 개발자가 재사용 가능하도록 반제품(半製品) 형태로 제공하는 모듈 집합이다. 확장 모델은 모델에 필요한 장비의 부가 기능 및 입출력 인터페이스 설정/제어 기능 집합이다.

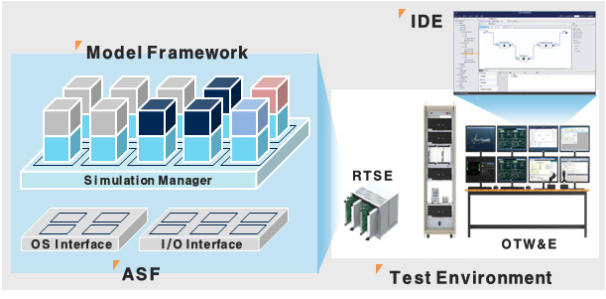

그림 1. ASMF 개발 범위
Fig. 1. Scope of ASMF.

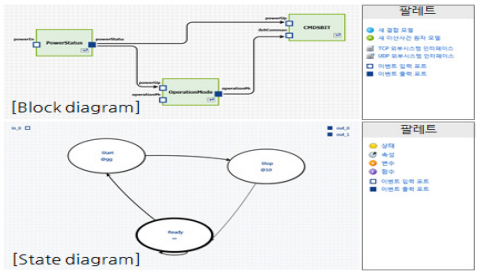

그림 2. 블록도 및 상태도 예시
Fig. 2. Example of block/state diagram.

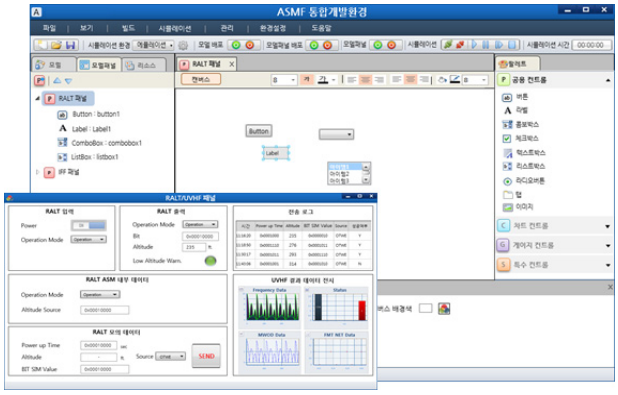

그림 3. ASMP(GUI) 예시
Fig. 3. Example of ASMP(GUI).

표 2. ASM 기본/확장모델 분류
Table 2. ASM base/extension model classification.

Level		Description	Modeling procedure
I	II		
Basis	Primary	Essential functions for performing missions	STEP 1 - Reuse the base model - Modify different features from modeling equipment
	Secondary	Basic functions of avionics equipment	
Plug-in	Accessory	Additional functions for performing missions	STEP 2 - Make by recycling extension models - Create an extension model for the modeling equipment using IDE
	Interface block	Function to set and control external I/O interface of equipment	

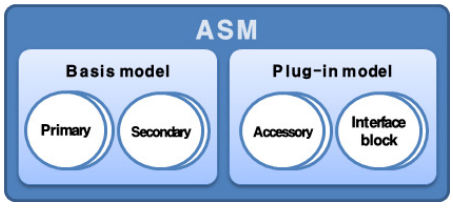

그림 4. ASM 기본/확장모델 기능 간 관계
Fig. 4. Relationship between ASM basis/plug-in model functions.

표 3와 그림 4는 기본 모델과 확장 모델의 분류 기준과 관계를 설명한 내용이다. 분류의 핵심은 재사용성이다. ASM의 재사용에 가장 연관되는 요소는 항전체계 설계이며, 항전체계 독립성이 큰 영역이 ASM의 기본 모델로 분류되고, 항전체계 설계에 종속적인 영역이 확장 모델에 해당한다. 기본 모델은 향후 SIL 신규 개발 시 재사용할 수 있도록 비행체 탑재장비들의 공통 기능과 대상 장비의 공통 기능을 식별하여 모델링하였다. 사용자는 기본 모델을 바탕으로 확장 모델에 해당하는 부가적인 기능과 외부 인터페이스를 추가하면 시뮬레이션 가능한 ASM을 완성할 수 있다.

기본 모델의 탑재 연동 기능은 초기화 기능, 자체점검(BIT; built in test) 기능 등을 포함하였다. 대부분 장비들은 탑재 연동 기능을 그대로 사용하지만, 모델링 대상 장비가 요청에 의한 자체점검(IBIT; initiated built in test) 기능이 없는 경우와 같이 특수한 경우에 한하여 기본 모델의 블록도를 수정하여 적용한다. 필수 기능은 모델링 대상 장비 별로 식별된다. CMDS(counter-measure dispenser system)를 예로 들면, 인벤토리 내 기만체를 투하하는 조건을 결정하는 기능, 투하를 제어하는 기능, 인벤토리를 관리하는 기능 등이 CMDS의 필수 기능이다.

개발한 ASM의 단위 또는 기능 수준의 검증을 위해서는 ASM을 제어할 수 있어야 하며 ASM의 응답을 확인할 수 있어야 한다. 본 연구에서는 이를 위해 각 ASM 별 검증용 MC ASM을 정의하였다. 검증용 MC ASM은 SIL 검증 대상인 MC의 해당 ASM 관련 기능만 추출한 것으로, ASM 기능 확인을 위한 명령인가와 결과 확인을 수행한다. 단위 테스트 용어로 보면 명령인가는 테스트 드라이버, 결과 확인은 테스트 스텁(stub)에 가깝다.

3-3 항공용 시뮬레이션 프레임워크(ASF)

ASF는 시뮬레이션 관리 (SM; simulation management) 기능과 RTSE 인터페이스 관리 (RIM; RTSE interface management) 기능으로 구성된다. SM은 DEVS 형식론 기반의 ASM을 구동하는 엔진을 비롯하여 초기화, 메시지 교환 기능 등의 응용 프로그램 영역에 가까운 기반 기능을 제공한다. RIM은 운영체제 및 펌웨어 영역에 가까운 기반 기능을 제공한다. 외부 IO를 제어하는 인터페이스 운용, HW 인터페이스 및 운영체제로부터의 독립성을 제공하기 위한 추상화 계층이 RIM에 해당한다.

ASF의 구성요소는 주로 독립성 제공 관점으로 분류하였다. SM은 구동될 RTSE 환경에 구애받지 않는데, 이는 RIM이 SM에 하드웨어와 운영체제를 기능 수준으로 추상화하여 제공하기 때문이다. SM과 하드웨어 및 운영체제 사이에서 RIM이 연계하기 때문에 하드웨어/운영체제 변경에 따른 SM 변경 영향성을 최소화했다.

SM의 구성요소 중 하나인 시뮬레이션 엔진은 시뮬레이션

실행을 위한 시뮬레이터와 시뮬레이션 대상 모델로 구분할 수 있다. DEVS 형식론에서는 코디네이터(coordinator)와 시뮬레이터를 구분하였으나, 본 연구에서는 시뮬레이터로 통합하였다. 시뮬레이터는 외부로부터 시간 전진(time advance)을 위한 시간 변화 이벤트와 외부 천이(external transition)의 발동 조건인 포트 입력 이벤트 및 데이터를 수신하고, 이후엔 내부 송신이 진행된다. 외부로부터 수신한 데이터들을 바탕으로 시간 전진과 포트 데이터 입력을 트리 구조로 구성된 모델의 루트인 최상위 결합 모델에 전달하여 트리 구조를 따라 전파된다. 전파된 정보는 해당하는 원자 모델에서 내부 천이(internal transition) 또는 외부 천이를 발동한다. 원자 모델 수준에서의 실행 결과는 내부 상태 변화로 그치거나 모델 출력으로 이뤄지며, 연결 관계가 있다면 이 흐름은 연결된 모델로의 외부 천이로 전파된다. 이러한 전파는 최종적으로 최상위 모델까지 전달되어 시뮬레이터 외부로 송신된다.

RIM의 구성요소 중 하나인 인터페이스 운용은 IDE에서 작성된 연동블록에 맞춰 통신 인터페이스로 데이터를 송수신하는 것이 주요 기능이다. 추상화 계층 중 하드웨어 추상화 계층은 인터페이스 종류 수준과 인터페이스 종류 별 제조사 구분 없이 인터페이스 실행을 기능 수준으로 추상화하여 제공한다. 운영체제 추상화 계층은 운영체제와 연관되는 기능을 추상화하여 제공한다. 인터페이스 운용은 추상화 계층 위에서 구동되며, 추상화된 방법을 통해 하드웨어 및 운영체제로부터 독립적인 실행을 보장 받는다.

Ⅳ. 운용 개념

SIL 개발자가 ASMF로 SIL을 개발할 때는 그림 4의 순서로 진행한다. 체계 요구도가 구체화 되어 SIL의 요구도가 수립되면 설계 단계에서는 기본 모델의 개발 여부를 결정한다. 기본 모델의 개발은 SIL에서 식별된 장비가 기본 모델에 없는 경우 수행된다. IDE는 기본 모델을 신규 생성하여 추가할 수 있는 것 외에도 배포 버전의 기본 모델을 수정/저장 및 버전을 개정할 수 있기 때문에 모든 경우의 대응이 가능하다.

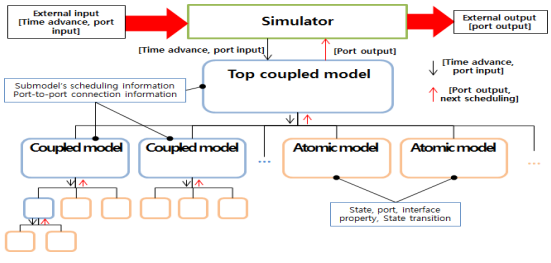

그림 5. 시뮬레이션 엔진의 구성과 실행개념
Fig. 5. Configuration and execution concepts of the simulation engine.

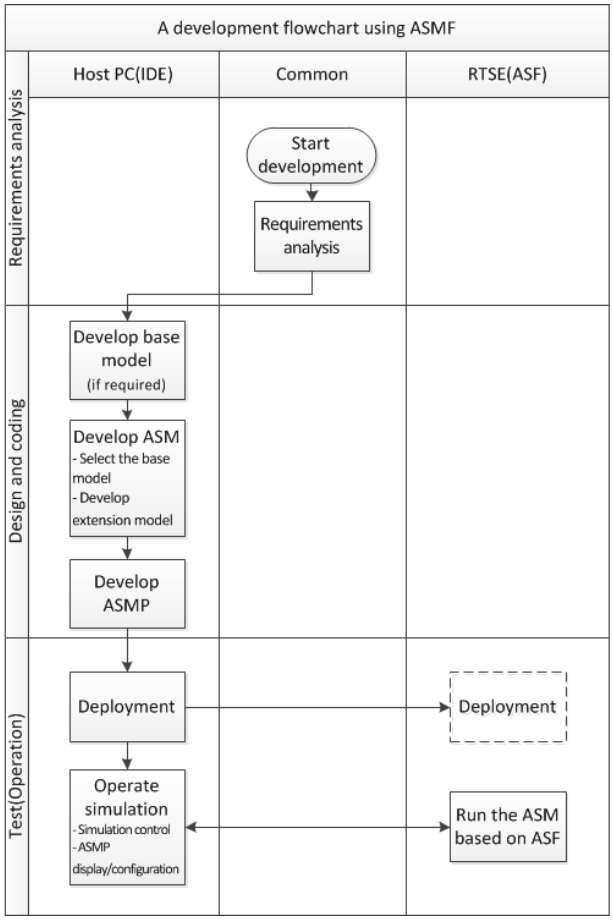

그림 6. ASMF 기반의 SIL 개발 흐름도
Fig. 6. A development flowchart using ASMF.

ASM 개발은 기본 모델 선택 후 확장 모델을 개발한다. 선정된 개발 대상 장비가 제조사만의 특수 기능이어서 기본 모델에 포함되지 않은 경우는 확장 모델의 부가 기능을 이용한다. ICD를 참조하여 통신 인터페이스 및 ASM 데이터와의 연결 작업을 마치면 확장 모델 개발까지 끝난 것으로, ASM 개발이 완료된다.

그림 6은 IFF 모델의 초기화 기능을 ASMF에서 ASM으로 구현한 결과이다. ASMF에서는 상태도를 작성하고, 상태별 속성과 상태천이 조건, 상태도에서의 입출력을 결정하여 원자 모델을 작성한다. Legacy는 이러한 논리를 직접 개발하여야 한다. 가독성, 유지보수 측면에서 ASM은 DEVS 형식론 기반으로 제작되므로 구현에 규약이 없는 Legacy보다 유용하다.

ASM 개발 후 ASMP를 제작한다. ASMP는 대상 ASM에 대한 데이터들을 대상으로 매뉴얼 혹은 주석을 참조하여 ASMP 컨트롤과 연결한다. ASMP는 SIL 운용 중에도 수정할 수 있다.

배포는 개발 완료된 ASM들을 RTSE에 적재하는 작업이다. 사전에 RTSE에 탑재되어 있는 ASF가 실행될 수 있도록 ASM 실행 파일과 관련된 설정 파일들을 함께 적재한다.

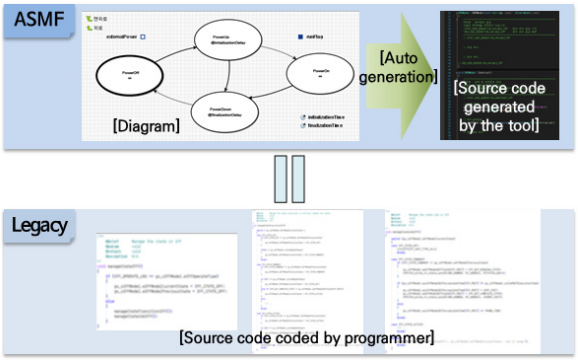

그림 7. ASMF와 legacy에서 IFF 모델의 초기화 기능을 구현한 결과
Fig. 7. Outputs of implementing initialization feature of IFF model in ASMF and legacy.

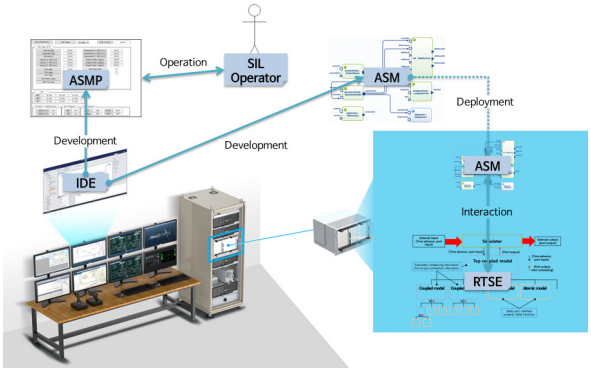

그림 8. ASMF 운용 개념
Fig. 8. ASMF operational concept.

시뮬레이션 운용은 시뮬레이션을 실행하여 항전체계를 검증하는 과정이다. ASMP의 ASM 모니터링 및 제어를 활용하여 항전체계 내 데이터들을 대상으로 검증을 수행한다.

위 설명을 간략한 운용 개념도로 표현하면 아래 그림과 같다.

Ⅴ. 결 론

ASMF는 모델 개발의 표준화로 항공용 SIL의 개발 기간 단축, 신뢰성 확보가 가능하다. 개발 기간 단축은 SIL 개발자들이 대상 CPU 및 통신 인터페이스 하드웨어와 운영체제를 고려하지 않고 온전히 모델 개발에 집중할 수 있는 여건을 제공하며, 개발된 모델은 프레임워크 아래 향후 재사용 가능한 형태로 관리가 가능하다. 간접적으로는 모델이 가시적으로 표현되어 고객과의 의사소통하기에 용이하고, 설계와 구현 일원화로 요구사항의 확인 과정 및 결과가 더 투명해질 수 있다. 신뢰성은 ASMF 연구 과정에서 확보되지만, 다른 사업에서 ASMF로 개발 및 검증한 모델을 신규 사업에 재사용함으로써 얻게 되는 이점으로도 볼 수 있다. 결론적으로, 모델 개발의 표준화로 비용 감소 및 신뢰성 확보가 될 수 있는 이유는 재사용성이 가장 큰 요인이다.

MDD의 유용성이 논의되고 점차 확장되고 있다. 소프트웨어 프로그래밍 요소를 최소화하고 도메인 기술에 집중하여 기술의 축적이 유리하고, 생산성/품질 향상과 설계 산출물-소스코드 일원화로 유지보수성의 향상을 함께 획득할 수 있는 장점이 있기 때문이다. 방위산업 해외 선진 업체인 Lockheed martin은 이미 90년도에 MDD를 도입하여 여러 플랫폼에서 사용한 가능한 SW를 확보하였고, 어플리케이션 개발 시간을 20% 감소시킨 바 있다[5].

모델 표준화를 목적으로 하는 ASMF는 MDD의 성격을 갖는다고 볼 수 있다. ASMF는 도메인 기술에 해당하는 항전체계 기술을 ASM 기본 모델 및 확장 모델로 축적시키고자 하며, 개발자로부터 DEVS 형식론 관련 소프트웨어 프로그래밍을 분리시켰고, 재사용/재활용성 향상과 설계와 소스코드 간의 일치성을 제공하기 때문이다. 이러한 강점을 바탕으로 ASMF의 프로그래밍 요소를 더욱 최소화하고, 개발자에게 제공할 편의 기능들을 추가 식별하여 보완해간다면 SIL 분야에 적지 않은 파급효과가 있을 것으로 보인다.

Acknowledgments

본 연구는 방위사업청과 방위산업기술지원센터의 지원(사업명 : 항공용 SIL의 비행체 탑재장비 모델 프레임워크 기술 개발, 계약번호 : UC170001D)하에 수행되었습니다.

References

[1] W. H. Chang, J. S. Park, Y. W. Jo and J. K. Byun, "Verification of hierarchically structured avionics system utilizing multi-mode system integration laboratory," *Journal of The Korean Society for Aeronautical and Space Sciences*, Vol. 45, No. 11, pp. 998-1005, Nov. 2017.

[2] Y. K. Kim, M. C. Kim, W. W. Choi and W. S. Oh, "Development of the MEP integration test environment for surion," *Journal of The Korean Society for Aeronautical and Space Sciences*, Vol. 39, No. 7, pp. 666-673, Jul. 2011.

[3] Y. W. Jo, B. G. Kim, J. S. Park and J. U. Lee, "Development of system integration laboratory for the verification of UAV avionics system requirements," *Journal of The Korean Society for Aeronautical and Space Sciences*, Vol. 40, No. 5, pp. 446-453, May. 2012.

[4] J. H. Kim, S. C. Lee and K. S. Ryu, "Development of avionics hot bench for avionics system integration test," *Journal of The Korean Society for Aeronautical and Space*

Sciences, Vol. 36, No. 5, pp. 507-513, May. 2008.

[5] H. S. Chin, T. H. Kim, MDD usability discussion and case analysis, Software Policy & Research Institute, 22, Daewangpangyo-ro 712beon-gil, Bundang-gu, Seongnam-si, Gyeonggi-do, Republic of Korea, SPRi Issue Report 2015-012, pp. 1-27, Dec. 2015.

[6] K. C. Hwang, M. G. Lee, S. J. Han, J. M. Yoon, Y. J. You, S. B. Kim, Y. I. Nah, J. H. Kim, D. H. Lee, "The DEVS integrated development environment for simulation-based battle experimentation," *Journal of the Korea Society for Simulation*, Vol. 22, No. 4, pp. 39-47, Dec. 2013.

[7] S. W. Kim, J. C. Shin, M. G. Seo, "Simulation model framework for avionics SIL," in *Avionics Systems Symposium Korea*, Yeosu, Korea, p.188, Jul. 2018.

[8] Y. J. Kwang, H. C. Shin and A. H. Pan, "Design and Implementation of an automated tool for inconsistency detection among application system development products," *The KIPS Transactions : Part D*, Vol. 11, No. 5, pp. 1087-1094, Oct. 2004.

서 민 기 (Min-gi Seo)
2012년 2월 : 한국항공대학교 컴퓨터공학과 (공학사)
2011년 10월 ~ 현재 : LIG넥스원 항공연구소 선임연구원
※ 관심분야 : 항공전자, 내장형 소프트웨어

신 주 철 (Ju-chul Shin)
2008년 2월 : 포항공과대학교 컴퓨터공학과 (공학사)
2009년 1월 ~ 현재 : LIG넥스원 항공연구소 선임연구원
※ 관심분야 : 항공전자, 내장형 소프트웨어

백 경 훈 (Gyong-hoon Baek)
2002년 2월 : 한국과학기술원 기계공학과 (공학사)
2002년 1월 ~ 2016년 5월 : ㈜도담시스템스 수석연구원
2016년 9월 ~ 현재 : LIG넥스원 항공연구소 수석연구원
※ 관심분야 : 항공전자, 실시간 시뮬레이션

김 성 우 (Seong-woo Kim)
2002년 8월 : 부산대학교 정보통신공학과 (공학석사)
2002년 10월 ~ 현재 : LIG넥스원 항공연구소 수석연구원
※ 관심분야 : 실시간 시뮬레이션 기법 및 시험환경 응용 개발