KARI

항공우주산업기술동향 9권1호 (2011) pp. 110~118

산업동향(기술동향)

http://library.kari.re.kr
에서 보실 수 있습니다.

위성시뮬레이터 산업기술동향

이훈희*, 박영웅*, 박근주*, 김대관*, 이선호*, 용기력*

Current Industrial and Technological Trends in Satellite Simulator

Lee, Hoon-Hee*, Park Young-Woong*, Park Keun-Joo*,
Kim Dae-Kwan*, Lee Sun-Ho*, Yong Ki-Lyuk*

ABSTRACT

In this paper, design techniques and technological trends applied on simulator development, which are based on Model Driven Architecture, are described. The methodology, process and modeling support tools for development of simulation software are introduced with both satisfaction of user's requirements and benefit to simulator developers of reducing cost. In addition, contents in ECSS standard for reusability of simulation model and its usages applied on industries are presented and major features of current technological changes with highlights of key-word are discussed.

초 록

본 논문에서는 최근 위성시뮬레이터 개발에 적용되고 있는 모델 주도형 구조의 설계 기술과 동향을 기술한다. 사용자의 요구사항을 만족시키고 개발자에게 비용 상의 이익을 주는 시뮬레이션 소프트웨어의 개발 방법론, 개발 과정, 개발 도구를 살펴본다. 아울러 시뮬레이션 모델의 재사용성을 위한 우주비행체 시뮬레이션 소프트웨어 분야의 유럽 표준의 내용과 산업 적용 사례를 설명하고 핵심 키워드별 최근 기술 변화의 주요 특징을 부각하여 설명한다.

Key Words : Simulation Model(시뮬레이션모델), Satellite Simulator(위성시뮬레이터), Simulation Software(시뮬레이션 소프트웨어), Model Standard(모델 표준)

* 이훈희, 박영웅, 박근주, 김대관, 이선호, 용기력 한국항공우주연구원 위성연구본부 위성기술실 위성제어팀
lhh@kari.re.kr, ywpark@kari.re.kr, kjp@kari.re.kr, dkk@kari.re.kr, shlee71@kari.re.kr, klyong@kari.re.kr

1. 서 론

검증 용 위성시뮬레이터의 진가는 위성체 개발 및 검증 일정상에서 요구하는 적기 조달과 시뮬레이션 요구사항의 충실한 반영에 달려 있다. 본 논문에서는 이러한 위성시뮬레이터를 구성하는 소프트웨어를 모델링 여부에 따라 시뮬레이션 모델과 시뮬레이션 기반소프트웨어로 나누어 양분하여 설명한다.[1]

본 논문에서 기술하는 시뮬레이션 모델(이하 모델)은 물리적 현상 혹은 장치의 기능과 동작을 모사하는 소프트웨어 컴포넌트를 의미하며, 시뮬레이션 환경을 나타내는 시뮬레이션 기반소프트웨어(이하 기반 SW)의 범위는 개발을 위해 필요한 자원으로 보는 넓은 의미의 정의와 달리 사용자에게 시뮬레이션 모델을 실행 제어, 감시 등의 기능을 제공하는 소프트웨어로 한정한다. 예를 들어, SimWARE 기반 SW를 사용하여 개발한 COMS(천리안위성) 시뮬레이터의 이름은 DSSS, MDVE 기반 SW를 이용한 Galileo, SIMSAT 기반 SW를 이용한 Lisa Pathfinder SATSIM, FDVE 기반의 Pleiades BOSS, SimTG 기반의 SatSim 등과 같은 기반 SW에 특정 위성의 구성 모델을 통합한 결과물이 위성 시뮬레이터임을 알 수 있다.

최근 위성시뮬레이터 기술의 변화는 기반 SW의 교유한 기능과 성능 향상보다는 객체 지향 패러다임 기반에서 모델 주도형 구조 설계를 통해 기존 소프트웨어의 재설계에 초점을 맞추고 진행되고 있다. 이러한 연구의 결과물로써 소프트웨어의 유연한 탈착성, 확장성에 중점을 둔 모델 설계, 구현, 통합 도구 등과 시뮬레이션 모델의 표준화된 접속 및 실행 구조를 제공하는 유럽 표준이 제정되었으며 현재 실증과정의 마지막 과정이 진행되고 있다. 국내의 유사한 추세로는 EuroSim 기반 SW를 달탐사선 시뮬레이터에 적용하여 모델의 재사용성을 고려하였으며 유연한 구조의 시뮬레이터 핵심 커널(Kernel) 설계 연구[2]가 진척을 보이고 있다.

2. 모델 설계와 실행

위성시뮬레이터를 비롯한 최근의 시뮬레이션 소프트웨어는 추상적인 상위수준의 구조적 형상과 몸체인 알고리즘을 분리시켜 설계하는 모델 주도형 구조(MDA: Model Driven Architecture) 패러다임을 기반으로 설계되고 있으며 이러한 객체지향 소프트웨어 구현은 UML(Unified Modeling Language) 과 같은 시각적 언어를 사용하는 추세이다.

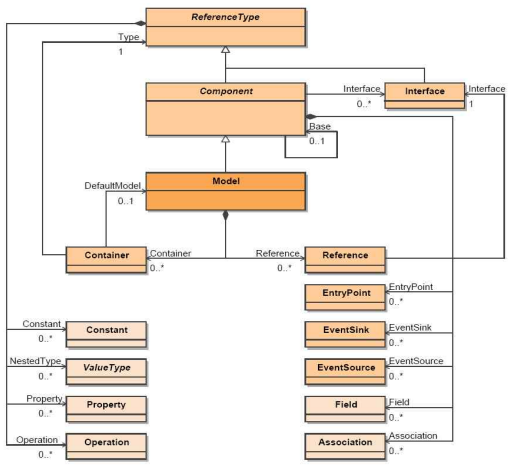

그림 1.Model[3]

유럽의 ECSS(European Cooperation for Space Standardization) 표준에서 정한 시뮬레이션 모델의 표준 구조 형상은 그림 1과 같다. 그림의 중간에 위치하고 있는 모델(Model)은 컴포넌트(Component)를 상속하고 있으므로 컴포넌트의 독립적인 모듈화 특징과 동적 바인딩 특징을 모두 포함하게 된다. 이러한 구조를 통해 아래와 같이 5가지의 다양한 모델 개발 방식을 조합하여 사용할 수 있다고 기술하고 있다.[3]

▹ 클래스 기반 설계

그림 1에서 내부 상태를 정의하기 위한 Field 와 기반 SW 내의 스케줄 조정이나, 다양한 이벤트를 등록할 수 있도록 호출 방식을 표준화한 엔트리포인트(EntryPoint)를 합성화(Composition) 할 수 있도록 하였다.

▹ 인터페이스 기반 설계

임의의 인터페이스(Interface)를 생성/구현할 수 있다.

▹ 컴포넌트 기반 설계

Container를 통해 모델 내부에 다른 모델을 합성화 할 수 있으며 Reference를 통해 다른 컴포넌트를 집단화(Aggregation)할 수 있다. 이러한 컴포넌트는 모델 집단이나 서비스 집단도 가능하다.

▹ 이벤트 기반 설계

모델이 다른 곳으로부터 이벤트를 수신할 수 있는 EventSink나 송신할 수 있는 EventSource를 합성할 수 있다.

▹ 데이터 흐름 기반 설계

일반적으로 모델은 입출력 흐름의 구성이 요구되며 모델의 Field는 개발자가 원하는 입출력 흐름을 정의할 수 있다.

또한 Association 요소를 이용하여 다른 모델이나 Field와 연관관계를 형성 할 수 있다.

그림 2는 모델 개발자로부터 개발된 시뮬레이션 모델과 초기값, 모델 간 연결정보, 스케줄 정보 등을 담고 있는 각종 시뮬레이터 구성 파일이 시뮬레이터 내에 결합되어 동작하는 과정을 나타내고 있다.

시뮬레이터의 요구사항이 확정되면 모델의 소프트웨어 요구사항이 작성되며 설계를 위한 초기 모델링 대상의 기초 자료를 준비한다.

모델 개발자는 그림 3과 같이 모델의 입출력 변수를 정의하고 내부의 하위 모델을 순차적으로 구현한다. update라고 표기된 내부 모델 간 점선은 이벤트 기반 설계의 전형적인 예이다. 기반 SW와 모델 간의 이벤트 설계에도 동일하게 적용할 수 있다.

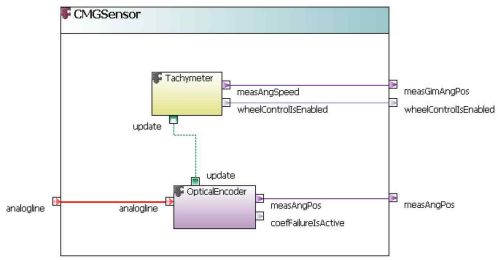

그림 3.모델 다이어그램 에디터[4]

그림 3의 다이어그램 편집기는 현재 영국, 독일, 프랑스 Astrium에서 공동개발한 SimMF[4]이며 기존 개발체제를 대체하고 있다. SimMF는 UML을 기반으로

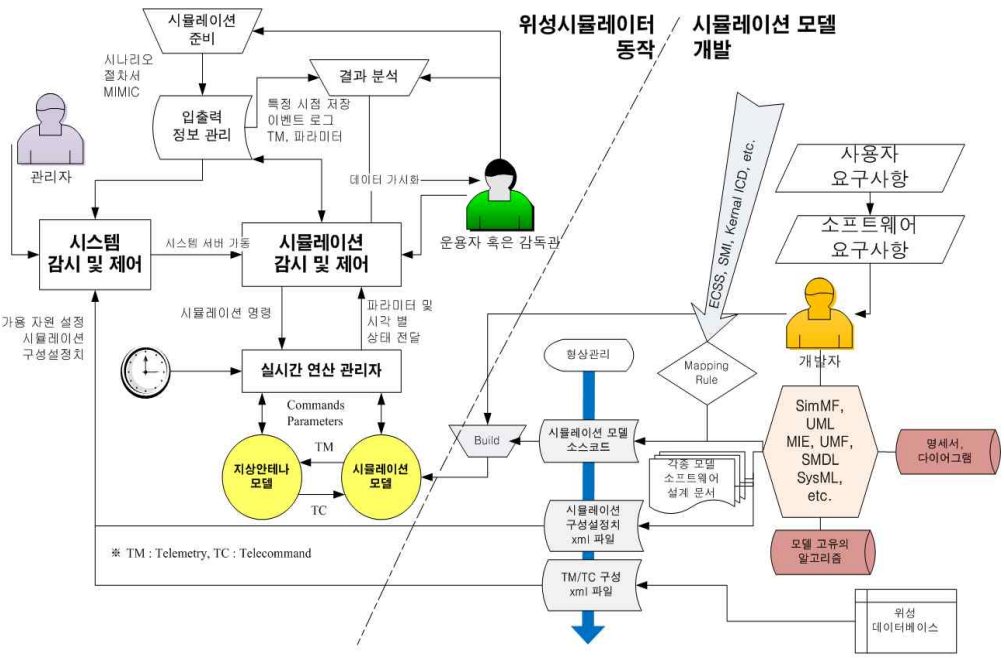

그림 2. 시뮬레이션 모델 개발 과정과 위성시뮬레이터 동작 구조

하는 위성 시뮬레이션모델 개발을 위한 다양한 기능을 포함하고 있다. 이 외에도 기업 컨소시엄이 개발한 SysML[5]와 VEGA Space에서 개발한 UMF[6] 등의 다양한 그래픽 환경의 모델 개발 도구가 있다.

모델 개발 도구의 특징은 구현상의 다소 차이가 있으나 기본적으로 모델의 입출력 데이터 정의, 이벤트 정의, 모델의 구성관계 등을 구현하면 다이어그램이 생성되며 마지막으로 고유의 모델 알고리즘(예, 모터의 동작)이 융합되면 모델의 제작은 완료되며 단위시험으로 이어진다. 또한 개발자가 원하는 기반 SW 및 운영체제의 호환성(Mapping Rule)에 부합하는 코드 생성과 표준화된 시뮬레이션 구성 설정치 파일 생성, 자동 문서화 기능을 포함하고 있다. 특히 생성되는 모델의 표준화된 소스코드는 컴파일 후 기반 SW와 결합하여 시뮬레이터에서 시뮬레이션 구성 설정치를 이용하여 동적으로 구성하고 실행시킬 수 있도록 지원한다.

최종 개발된 바이너리 형태의 모델과 시뮬레이션 구성 정보는 기반 SW가 위치하는 해당 플랫폼의 특정 폴더에 이동되며 이후의 사용자별 시뮬레이션의 준비, 시작, 결과 분석의 운용은 그림 2의 점선 좌상단의 기능 블록 다이어그램에서 나타내고 있다. 자세한 동작 과정은 천리안 위성 시뮬레이터 기반소프트웨어의 특징을 참조한다.[1]

3. 시뮬레이터 기술 현황

2 장에서 언급된 모델의 개발, 기반 SW와의 결합, 시뮬레이터 상의 유기적 동작 과정의 개괄적인 배경에서 현재 시뮬레이터 개발에 적용되고 있는 기술의 특징과 변화를 살펴보자.

3.1 런타임(Runtime) 위성체 구조 생성 방식

통합개발 과정에서 개발자에 의해 고정화된 소프트웨어 구조의 기존 방식과는 달리 시뮬레이터 사용 시작 시점(Booting)에서 가상 위성체의 소프트웨어 구조가 생성되고 하위 모델을 순차적으로 초기화하는 메커니즘을 지향하고 있다. 천리안위성을 구성하는 자이로(Gyro) 센서는 3축 상에 각각 2중화 구조로 총 6개가 장착되어 있다. 자이로 센서는 모두 동일한 종류이므로 모델 개발 과정에서 자이로 모델은 하나만 제작된다. 이 모델은 위성체의 구조에 따라 상이하게 결합되므로 결합 구조 정보는 읽어 들이기 쉬운 파일에 저장하여 시뮬레이터 초기화 시 동적으로 로드(load)하는 방식을 채택하고 있다. 기존 시뮬레이터의 구조의 경우 위성체 구조나 모델의 실행 주기, 순서, 초기값의 변경이 필요한 경우 빈번한 컴파일과 통합검증 과정이 수반된 것과는 대조적이다.

표 1. 구성파일(XML) 정보

분류	항목
1	시각 정보(적분간격, 임무시각 등), 시뮬레이션 제어 정보
2	기반 SW/ 모델 계층구조 및 변수
3	소프트웨어 간 연결 정보
4	경계값 계산을 위한 격자 구성 정보
5	로그 파일 출력 정보

표 1은 OpenSimKit[7] 에 사용되고 있는 구성파일의 예이며 초기화 시 필요한 다양한 정보가 포함된 것을 알 수 있다. 이러한 파일은 이종의 시뮬레이터로부터 저장 혹은 로딩될 수 있도록 XML(Extensible Markup Language) 포맷을 선호한다.

표 2는 유럽 ECSS-E-TM-40-07 표준에 따른 시뮬레이터의 초기화 과정을 나타내고 있다. 모델의 동적 구성과 초기화를 위해 정의된 Assembly

(3.5 절 참조)나 Configuration(3.11 절 참조) 역시 XML 파일에 저장된다. 모델을 실행하기 위한 연산 스케줄링의 설계는 그림 4와 같은 설정 창을 통해 Assembly 내에 구성된 모델을 대상으로 Schedule 에 태스크(task)와 이벤트를 저장한다.

3.2 모델 개발과 기반 SW 개발의 비종속화

위성 개발 사업에서 개발된 모델을 재사용하기 위한 기존 전략은 일반적으로 기반 SW와 개별 모델 간의 접속 규약 추가 혹은 수정을 최소화하고 이에 종속된 모델의 개발과 모델 간 접속을 진행하는 방식이었다.

표 2. 시뮬레이터 초기화 과정

순서	모드	설명
1	구성	기반 SW의 초기화 후(그림 2의 관리자 역할), Assembly로부터 정보를 분석하여 모델의 계층화 구조 생성하고 Configuration으로부터 Field 값을 로딩한다. 보통 Publish 과정이라고 한다. 이때 위성체의 계층화 구조가 동적으로 형성된다.
2	결합	기반 SW는 모델의 계층화 구조를 통해 모든 모델의 접속을 연결한다.
3	초기화	기반 SW는 스케줄러(Schedule)가 실행할 엔트리 포인트(Entry Point)를 등록한다.
4	준비	시뮬레이션 시각은 0초이며 정지상태이다.

최근의 모델 개발 방식은 기반 SW의 기능이 부족하거나 최악의 경우 미선정 상태에서 모델 개발이 착수되는 경우, 기반 SW의 종속적인 부분을 무시하고 초기 모델 개발에 집중할 수 있도록 기반 SW와는 독립적이고 평행한 개발 환경을 실현하고 있다.

모델 개발 도구의 코드 생성 기능은 기반 SW 의 핵심인 커널이 요구하는 접속 규약에 따라 모델 코드를 생성하므로 모델 개발을 기반 SW와 분리하여 추진할 수 있다. 예를 들어, 모델 개발 도구인 SimMF가 기반 SW인 SimTG 커널에 부합하는 코드를 생성할 수 있다. 따라서 위성시뮬레이터 개발자는 기반 SW의 요구사항 검토 및 선정 과정의 시간을 단축하고 위성개발 사업 일정상의 모델 개발 착수 시점을 앞당기는 장점을 얻는다.

반면에 개발자가 원하는 기반 SW가 모델링 도구에서 지원하지 않을 가능성이 있다는 단점이 있다. 이러한 문제는 수많은 기반 SW의 홍수 속에 모델링 도구 개발자인 우주기업에게는 반복되는 부담이며 이를 극복하기 위하여 SimMF, SIMSAT[8], EuroSim[9] 등의 기반 SW는 기반 SW와 모델 간 유럽표준 항목이 정의된 ECSS-E-TM-40-07[4]을 반영하여 모델 개발과 기반 SW 개발의 연성을 제거하고 있으며 바이너리 형태의 모델 자원의 공유가 현실화 되었다.

그림 4의 독일 VEGA가 개발한 모델 개발도구의 스케줄 편집기는 그래픽 환경에서 연산 주기와 순서를 정의할 수 있도록 하여 개별 모델 개발과, 모델과 기반 SW의 통합 과정 사이의 프로세스 구분을 불확실하게 변모시키고 있다.

3.3 모델 개발과 데이터베이스 정보와의 비종속화

모델 설계 정보와 위성 데이터베이스의 방대한 정보와의 긴밀한 관계로 인하여 모델 개발 시 정보 간의 연결 관계를 유지하는 작업이 요구된다. 가령 휠(Wheel) 모델의 출력 중 모터 전류 값을 갖는 시뮬레이션 출력 변수가 데이터베이스 내의 12,700 여개의 TM(Telemetry) 가운데 AIRDRCA 라는 TM 코드와 맵

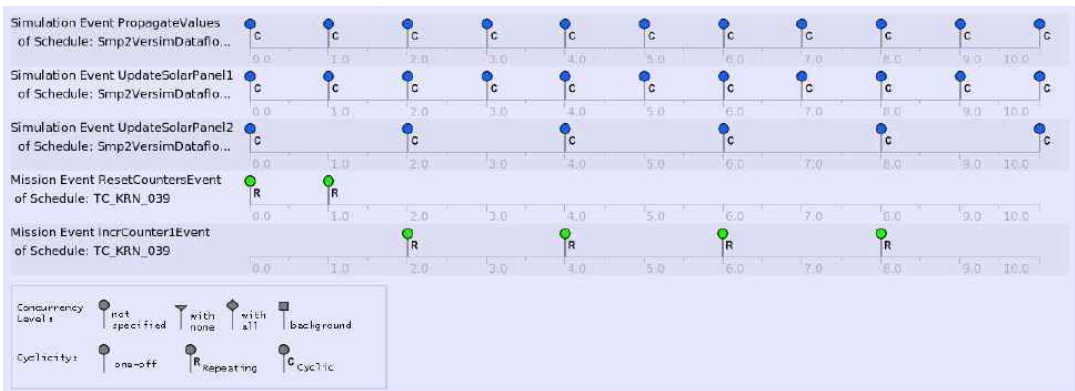

그림 4.모델의 이벤트 스케줄링[8]

핑(Mapping) 관계를 형성하고 유지하는 것은 부담이 될 수 있다. 이러한 부담을 감소시키기 위해서 모델 개발 시 출력 변수가 동적으로 데이터베이스 상의 TM 코드와 연결이 되어 현재 상태를 개발자에게 제공하도록 한다. 데이터베이스에서 정의된 TM 코드의 이름이 변경되거나 삭제된 경우 연결 오류가 발생하여, 통합 검증단계 이전인 모델 코드 구현 단계에서 연결 상태를 보장할 수 있다. 요컨대 모델 개발 도구를 통해 최신의 데이터베이스를 동적으로 연결하여 종속성으로 발생하는 오류를 사전에 방지하도록 설계하고 있다.

3.4 위성 설계 정보 기반 실시간 오류 추적

성능 혹은 제약 조건 등의 위성시스템 수준의 요구사항 파라미터 값이나, 위성체 구성 정보, 정상 운영 상태의 TM 값, 고장 판별 기준, 각종 규칙 등으로 구성된 위성 데이터베이스를 시뮬레이터와 연동하여 규칙 위반, 원인, 복구 방법을 로그 메시지를 화면에 출력한다. SIMSAT의 UMF도구는 허용 범위를 나타내는 LIMIT 표준화 속성을 설정하여 런타임 상에서 오류를 검출하는 데 적용한 바 있다. 아울러 FDIR(Failure Detection, Isolation and Recovery)의 동작 과정이나 각종 이상상태 분석을 지원할 수 있다.

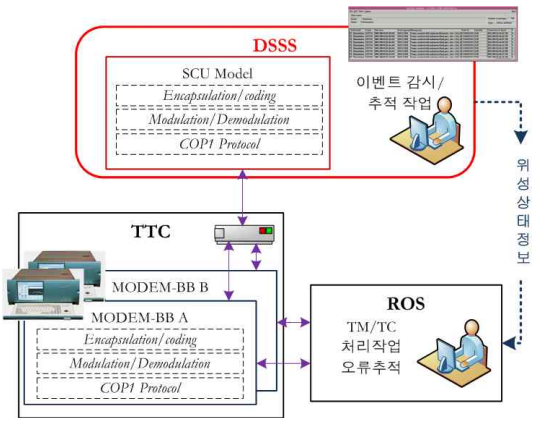

그림 5.LOCKOUT 오류 검출 과정[10]

천리안위성에 사용된 폐루프 명령 전송 프로토콜인 COP-1(Command Operation Procedure-1)[11]시험의 예를 들면, 명령 프레임(frame) 생성을 담당하는 지상의 FOP-1(Frame Operation Procedure-1)과 이를 문제없이 수신했는지 확인하는 위성의 FARM-1(Frame Acceptance and Reporting Mechanism-1) 사이의 오류를 방지하기 위한 각각의 슬라이딩 윈도우(Sliding Window) 상태를 시뮬레이터 화면에 출력한다. 슬라이딩 윈도우를 벗어나 "Lockout" 상태로 전환되어 원격 명령이 송신되지 않는 원인과 해결 방법을 사용자에게 알려줌으로써 시험자의 절차 상의 오류는 물론 TTC나 ROS의 오류 검출도 발견할 수 있다.

3.5 시뮬레이터 소프트웨어 재사용성 의미 확장

앞서 유럽 표준과 관련된 바이너리 형태의 모델 재사용성에 대해서 간단히 언급한 바 있다. 기존 모델 재사용 방식에서 하나의 위성 개발 사업 내에서 검증 목적으로 요구되는 다양한 시뮬레이터가 하나의 일관된 모델을 공유하거나, 연속적인 혹은 이질적인 위성 개발 사업에서 기 개발된 모델을 재사용하는 경우에 기반 SW 혹은 기반 HW에 종속된 접속을 위한 추가적인 코드(Wrapper Code) 작성이 요구되었다. 최근에 우주비행체 시뮬레이션 모델 재사용의 의미는 표준에 입각하여 컴파일과 재검증 작업을 최소화하여 런타임 소프트웨어를 재구성하여 실행하는 것으로 확장되고 있다.

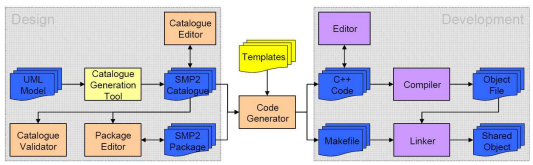

그림 6.시뮬레이션 모델의
Metamodel 구현 및 빌드[12]

유럽의 ECSS-E-TM-40-07의 Volume 2는 시뮬레이션 모델 정의 언어(SMDL: Simulation Model Definition Language)를 통해 Metamodel을 구현하는 방법을 상세히 기술하고 있다. 그림 6은 ESOC의 Swarm Constellation Simulator[12]에 실제 적용된 표준화된 모델 개발 과정을 나타낸다. 그림 7는 UMF 도구를 이용하여 설계한 카탈로그(Catalog)와 어셈블리

(Assembly)의 예이다. 카탈로그는 모델의 상위 수준의 모양을 나타내며 구체적으로 모델, 데이터 타입, 인터페이스의 정의를 말한다. 어셈블리는 모델 인스턴스 통합 구성을 나타낸다. 그림 7의 왼쪽 카탈로그 구조에서 Model Heater는 TurnOn 과 TurnOff의 점등 명령이 정의되어 있는 것을 알 수 있으며 오른쪽 어셈블리 구조에서 NiCdBattery 종류의 배터리가 총 Batt1, Batt2, Batt3로 인스턴스화된 것을 확인할 수 있다.

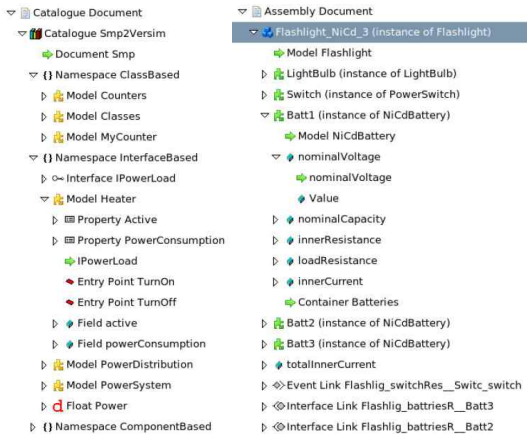

그림 7.카탈로그와 어셈블리 설계[8]

3.6 위성체 구조와 유사한 계층화 모델 설계 지향

모델 설계 단계에서 시스템에서 하위 시스템, 컴포넌트로 분화하여 개발하는 방식은 코드의 재사용성과 가독성을 높이고 분산 연산 성능과 일정상의 이익이 있다. 세분화된 모델은 다중 프로세서 상의 용이한 분산처리를 실현하며, 위성을 구성하는 장치의 사양이나 구체적인 설계 파라미터가 확보되지 않은 상태에서 상위 수준의 추상적인 설계 및 코드 구현을 진행할 수 있다는 장점이 있다.

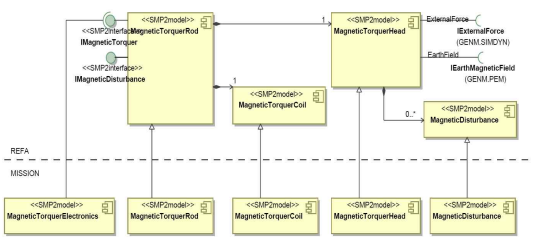

그림 8. 자기토커 상위 수준 설계[12]

그림 8은 Swarm Simulator를 구성하는 자기토커의 초기 상위 수준의 설계도를 나타내고 있다. REFA(Reference Architecture)와 ECSS 표준을 적용하여 Swarm 임무에 맞게 구체화시키고 있다. 다만 위성체 구조에 집중하여 수직 계층의 세분화 정도가 높아지면 모델 혹은 컴포넌트 간에 인터페이스(Interface) 및 인스턴스(Instance)가 불필요하게 증가하여 시뮬레이터 초기화 최대 시간의 요구사항을 위협할 수 있다. 또한 실제 위성체의 계층화 구조를 가독성 혹은 시각적 설계를 위해 소프트웨어 설계에 최대한 유사하게 적용하는 과정에서 비효율적인 설계로 이어질 수 있다. 예를 들어, 다양한 위성 내부 장치와 연관되는 전력, 온도 등의 물리적 수치 연산을 위성체 구조와 유사하게 구현되기 위해서는 전력을 사용하거나 열손실이 발생하는 모든 전자장치 혹은 부품이 제각각 전력계와 열제어계 모델의 특성을 다중상속(Multiple Inheritance)하여 모델의 복잡도를 증가시킨다. 따라서 이러한 물리적 거동 및 현상을 계산하는 모델은 예외적으로 위성체 구조를 지향하지 않으며 독립된 수치적분기와 같이 각 모델의 요구한 요청에 응답만 하는 전통적 구조를 여전히 유지하고 있다.

3.7 하드웨어 특성과 동작 기능의 모델링 분리

국내외 많은 위성 개발 기관 혹은 업체가 제작하는 위성체는 구성하는 장치의 구성과 기능이 유사하며 표준화 추세에 따라 더욱 유사한 구조를 지향할 것이다. 독일 VEGA Space 업체는 이러한 동작 기능의 유사성을 파악하여 모델의 재사용성을 높이고 있다. 일례로 위성의 데이터처리계에 TM/TC 처리 장치는 다양한 업체로부터 개발되어 하드웨어 특성은 차이가 있으나 주요 동작 기능은 대동소이하다. 따라서 모델 설계 시 HW 부분과 기능 부분의 두 계층으로 나누어 재사용 시 기능 부분은 수정 없이 사용하고 HW 구성 부분은 공급업체로부터 전달된 HW 특성치를 고려하는 전략을 제시하고 있다.

3.8 실시간 모델 편집기의 일관성 확인 기능

UML 모델링에서 모델 개발 산출물은 시각적으로

다이어그램, 명세서 등 다양한 형태가 될 수 있다. 영국, 독일, 프랑스 Astrium의 SimMF의 경우, 모델 개발 시 Eclipse EMF, GMF, Xtext[4] 플러그인을 통해 각각 테이블, 다이어그램, 텍스트 편집 방식을 혼합하여 사용하기 때문에 동적으로 연결된 데이터를 실시간으로 확인하여 한쪽에서의 수정이 다른 두 곳으로 파급되도록 하였다. 이러한 방식을 통해 동일한 모델의 3가지 형태를 보이고 반복되는 저장을 피할 수 있다.

3.9 반환값 없는 스크립트 명령

시뮬레이터 개발자 혹은 사용자는 목적에 따라 일련의 시험 로직을 구현한 스크립트를 실행시켜 자동으로 장시간의 단위 혹은 통합검증 시험을 수행한다. CCS의 UCL, OpenCenter의 ELIZA, SCOS-2000의 TCL, Pluto 언어는 이러한 시험을 위한 전용 스크립트 언어의 예이다. 기존 시뮬레이터는 이러한 스크립트가 실행될 때 반환값을 통해 사용자에게 이상 유무를 알려 주었으나 최근에는 반환값이 없는 Void 타입의 명령을 사용하여 예외(Exception) 처리를 통한 로그북 메시지 출력 방법으로 대체되고 있다.

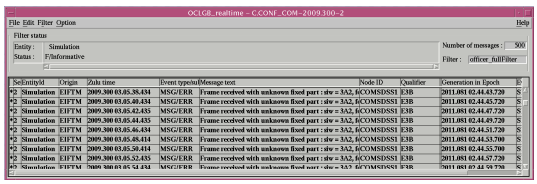

그림 9.기반 SW의 구성

예를 들어, 고장 주입 명령을 전송하여 시뮬레이터가 해당 메소드를 실행 한 후 성공한 경우 TRUE 값을 실패한 경우 FALSE 값을 보낸다고 하자. 반환값 전달 방법에서 실패한 경우에 대한 오류 원인 추적 시, 발생된 FALSE 값을 이용하여 소프트웨어에 의한 것인지 다른 논리적 오류에 의한 것인지 파악하는 것이 쉽지 않다. 반환값이 없는 예외 처리를 통해 로그북에 나타난 순차적인 오류 정보를 통해 복합적인 디버깅 시험을 지원한다. [13]

3.10 크로스 플랫폼 빌드 스크립트

사용자는 기반 SW의 클라이언트가 윈도우, 리눅스 등 다양한 플랫폼에서 동작하기를 원한다. 이에 따라 다수의 모델을 컴파일하고 링크하기 위해서 모델 개발 도구는 리눅스와 윈도우에서 동작하는 빌드 스크립트를 자동으로 생성하는 기능을 추가하고 있다.[14]

3.11 기반 SW 구조의 유연한 확장성

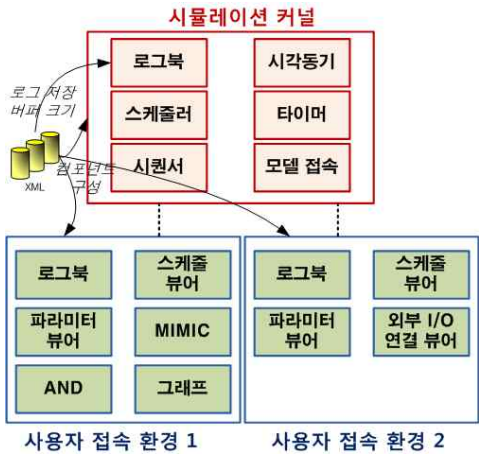

그림 10.기반 SW의 구성

앞서 언급한 모델의 구조뿐만 아니라 기반 SW의 구조도 유연하게 재구성할 수 있다. 기반 SW의 기능성 컴포넌트의 하나인 로그북(Logbook 혹은 Logger)의 경고, 오류 메시지의 기록 최대 용량 혹은 로그 파일 저장 위치 등은 기반 SW를 다시 빌드하지 않아도 쉽게 변경할 수 있다. 만약 그래프 기능을 추가하고자 한다면 서비스 목록에 그래프 컴포넌트를 추가 등록하면 된다. 이러한 정보 역시 XML 형식의 파일에 저장되고 재구성마다 갱신되고 저장된다.

3.12 고장 주입 기능의 표준화

모델의 Field 데이터를 강제적으로 변경시키기 위해서 ECSS-E-TM-40-07 표준에 그림 11과 같이 Fallible 모델을 정의하였다. 해당 데이터를 기반 SW에서 접근, 변경할 수 있으며 변경 후 다른 데이터 라인으

로부터 갱신되지 않도록 차단해준다.

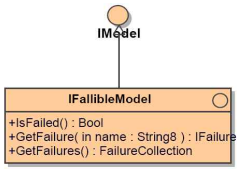

그림 11.FallibleModel[3]

4. 결 론

UML 기반의 시뮬레이션 소프트웨어 모델의 설계, 구현 기술 변화는 기반 SW와 모델 간, 시뮬레이터와 위성데이터 간의 완전한 독립을 예견하고 있다. 기반 SW는 시뮬레이터 개발 착수 시점 이전에 준비되어야 하고 모델링을 위한 하드웨어 정보가 계획된 시점에서 전달되어야 하는 기존의 시뮬레이터 개발의 제약사항은 시뮬레이션 소프트웨어의 탈착성과 개발 방법론이 정돈된 유럽 ECSS 표준 제정과 실증을 통해 극복되고 있다. 본 논문에서 기술된 시뮬레이터의 소프트웨어 구조, 방법론, 개발 프로세스에 관한 기술적인 변화에 대해 위성시뮬레이터와 관련하여 기술하였으나 시뮬레이션 산업 전 분야에서 유용하게 참조될 수 있을 것으로 기대된다.

참고문헌

1. 이훈희, 구철회, 박영웅, 박근주, 김대관, 용기력, “천리안위성 시뮬레이터 기반소프트웨어 구조와 기능”, 항공우주학회 춘계학술대회, 2011

2. Cheol-Hea Koo, Hoon-Hee Lee, Yee-Jin Cheon, "SMI Compatible Simulation Scheduler Design for Reuse of Model Complying with SMP Statndard", Journal of Astronomy and Space Sciences JASS, 2010

3. ESA-ESTEC, ECSS-E-TM-40-07 Simulation modelling platform - Volume 2: Metamodel, ECSS, 2011

4. Olivier Zanon, Ambros Morscher, “The SimTG Simulation Modeling Framework”, 11th International Workshop on SESP, 2010

5. http://www.sysml.org

6. Fritzen P., Segneri D., Pignede M., "SWARMSIM-The first fully SMP2 based Simulator for ESOC", 11th International Workshop on SESP, 2010

7. http://www.opensimkit.org

8.http://www.esoc.esa.int/portal/egos-web/products/simulators/SIMSAT

9. http://www.eurosim.nl

10. 이훈희, 김방엽, 박봉규, 양군호, 백명진, 천용식, “천리안위성 운용 준비를 위한 위성시뮬레이터 활용효과 분석”, 항공우주기술지, 제 9권, 제 1호, 2010

11. STAB, PSS-04-107 Issue 2 Packet telecommand standard, ESA, 1992, pp. 6

12. Max Pignede, Jose Morales, Peter Frizen, John, Levis, "Swarm Constellation Simulator", Spaceops, 2010

13. Jens Eickhoff, Simulating Spacecraft Systems, Springer, 2009

14. Ant technology, http://ant.apache.org

15. Jens Eickhoff, Michael Fritz, Rouven Witt, Ivan Kossev, Mario Kobald, Alexander Brandt, “OPENSIMKIT-AN OPEN-SOURCE SYSTEM SIMULATION ENVIRONMENT APPLIED TO SPACE PROJECTS”, 4th International Conference on Astrodynamics Tools and Techniques, 2010