한국시뮬레이션학회 논문지
Vol. 19, No. 4, pp. 261-270 (2010. 12)

무기체계 효과도 분석을 위한 통합 모의 엔진의 서비스 구성 방안 연구

김태섭[1†] · 박준호[1] · 김현휘[1] · 박찬종[1] · 이강선[1]

Services of an Integrated Simulation Engine for Weapons Analysis

Taesup Kim · Joonho Park · Hyunhwi Kim · Chanjong Park · Kangsun Lee

ABSTRACT

An integrated simulation engine provides tools, services, and standards to support various activities in the entire M&S from modeling and simulation to analysis of the simulation results. Many countries have developed integrated simulation engines to efficiently assist complex M&S activities. However, we do not have domestic simulation engines especially designed for defense M&S, therefore, developing M&S softwares still remains as a hard task with high cost and tine. OpenSIM(Open Simulation engine for Interoperable Models) is an integrated simulation engine and provides tools, services and standard interfaces for weapons analysis. OpenSIM's services are comprised of classes, member functions and data attributes which are commonly used in modeling, simulating and analyzing weapons systems. In this paper, we introduce OpenSIM's services in C++ APIs and illustrate them through an ASM example(Air to Surface Missile).

Key words : Defense Simulation Engine, MOE(Measure of Effectiveness) of Weapons, Services

요 약

무기체계 효과도 분석을 위한 통합 시뮬레이션 엔진은 모델의 개발부터 시뮬레이션 수행 및 분석에 이르는 모델링 및 시뮬레이션 전 단계에 걸쳐 도구와 서비스 및 인터페이스를 제공해야 한다. 다양한 국방 통합 시뮬레이션 엔진을 제공하고 있는 국외와는 달리, 국내에서는 자체적인 분산 실시간 모의체계 엔진에 대한 연구 및 개발이 미흡하여 무기체계 M&S(Modeling and Simulation) 활동 시 유사 업무가 반복되고 있는 실정이다. 본 논문에서는 국외의 기 개발된 국방 M&S 모의엔진의 서비스를 비교 분석하여 무기체계 효과도 분석을 지원하기 위한 통합 시뮬레이션 엔진인 OpenSIM(Open Simulation Engine for Interoperable Model)을 소개하고, OpenSIM에서 제공하는 서비스의 범위 및 내용을 정의한다. 정의된 서비스는 모델링 · 실행 · 분석에서 공통적으로 많이 사용되는 클래스, 함수 및 데이터 등을 구현하여 프로그래밍의 복잡도를 감소시킬 수 있다.

주요어 : 국방 시뮬레이션 엔진, 무기체계 효과도 분석, 서비스

1. 서 론

무기체계 효과도 분석은 분석 대상 무기의 구조 및 행위, 무기체계가 배치 및 운용 될 교전 자연환경 및 운용 교리를 모두 고려해야 한다. 환경 모델, 무기 모델 및 운용 모델은 네트워크를 통해 이 기종 컴퓨터상에서 분산되어 질 수 있으며, 실 기동 및 가상 시스템 혹은 외부 모델링 시스템 등과의 연동을 통해 시뮬레이션 될 수 있다. 이러한 복잡한 무기체계 효과도 분석 시스템을 제작하는 것은 어려운 작업으로 도메인 전문가, 시스템 분석가, 프로그래밍 전문가 등의 다양한 전문가들의 협력을 통해 이루어질 수 있다. 통합 시뮬레이션 엔진은 분석 대상 시스템의 모델링 · 실행 · 분석을 지원하기 위한 도구와 서비스, 표준을 제공한다. 도구는 사용자와의 상호작용을 통해 모델링 · 실행 · 분석 단계를 결점 없이(Seamless) 진행 가능하도록 지원하는 독립적 실행 프로그램으로 도메인 전문가, 시스템 분석가들의 프로그래밍 작업을 최소화 시키는

* 본 연구는 방위사업청과 국방과학연구소의 지원으로 수행되었습니다.(UD080042AD)
접수일(2010년 9월 30일), 심사일(1차 : 2010년 11월 4일), 게재 확정일(2010년 11월 15일)
[1] 명지대학교 컴퓨터공학과
주 저 자 : 김태섭
교신저자 : 이강선
E-mail; ksl@mju.ac.kr

모델 편집기, 시뮬레이션 제어기 등을 포함한다. 서비스는 모델링 · 실행 · 분석에서 공통적으로 사용되는 클래스, 함수 및 데이터 구조 등을 선 구현해 놓은 것으로 프로그래밍 작업의 복잡도 감소를 유도한다. 모델 및 각종 시뮬레이션 자원에 대한 표준(Standard) 작업은 상호 운용성(interoperability)을 위한 기반을 제공한다.

국외에서는 무기체계 효과도 분석 시 모델의 재사용성과 상호 운용성 증진을 위하여 다양한 통합 시뮬레이션 엔진을 연구한 바 있다. 미 공군에서 개발된 JMASS(Joint Modeling and Simulation System)[1-5]는 객체 단위 모델의 표준화와 연동을 통해 체계 통합 수준으로의 모의 분석을 지원하기 위한 도구 및 서비스를 지원하는 통합 모의 엔진이다. 미 PEOSTRI(시뮬레이션, 훈련, 장비, 획득관리 책임기관)의 주도로 개발된 OneSAF(One Semi-Automated Forces)[6-8]는 첨단 개념 연구, 연구개발 획득, 군사훈련/작전에 활용될 모델을 개발함에 있어 표준 모델을 정의하고 모의 엔진 구조를 표준화하여 모델 상호 연동성과 재사용을 향상시키는 노력을 행하였다. 또한, 미 WarpIV 사에서 개발된 WarpIV[9]는 High Speed communication Service와 다양한 Software Utilities, 병렬 및 분산 시뮬레이션 지원 서비스 등을 지원한다.

국내는 모델의 개발부터 시뮬레이션의 수행까지 전단계의 도구와 서비스, 인터페이스를 제공하는 분산 실시간 모의체계 엔진에 대한 연구 및 개발이 미흡한 상태이다. 따라서 기 개발된 유사 무기체계를 재사용하여 신규 무기체계에 대한 모델 개발에 활용한 사례가 매우 드물며 유사 업무의 M&S(Modeling and Simulation) 활동을 반복하고 있다[10-14]. 본 논문에서는 무기체계 효과도 분석을 지원하기 위한 통합 시뮬레이션 엔진인 OpenSIM(Open Simulation Engine for Interoperable Model)을 소개한다. OpenSIM은 모델 개발과 시뮬레이션 수행, 분석을 비롯하여 외부 연동기를 통한 타 공학도구, Web, C4I, LV, 다해상도 모델과의 연동을 지원하도록 설계되었다[15]. 이에 본 논문에서는 현재 개발 중인 OpenSIM의 서비스 범위 및 내용을 소개한다. 또한, 이를 위해 기존 국외 국방 M&S 모의엔진의 서비스를 살펴보고, 비교 분석한 결과를 바탕으로 OpenSIM의 서비스 범위 및 내용을 C++ API 형태로 정의하며, 간단한 예제를 통해 구성된 서비스의 유용성을 입증하도록 한다. 본 논문의 구성은 다음과 같다. 2장에서는 국외 통합 시뮬레이션 엔진에서 제공하는 서비스를 살펴보고, 3장에서는 OpenSIM의 개발 목적과 지원하는 서비스를 식별하여 설명한다. 또한 식별된 서비스의 유용성 입증을 위하여 간단한 공대지 미사일 분석 예제를 보인다. 4장에서 결론과 함께 향후 연구 방향을 제시한다.

2. 기존 연구

본 장에서는 국방 M&S 모의엔진의 대표적 실례인 JMASS와 OneSAF, WarpIV의 특성을 파악하고, 각 모의엔진에서 지원하는 모델링 · 시뮬레이션 · 분석 서비스를 알아보도록 한다.

2.1. JMASS

JMASS는 재사용을 위한 표준 소프트웨어 구조 모델 및 인터페이스, 모델링 및 시뮬레이션을 위한 서비스, 모델링 · 실행 · 사후분석을 위한 시각적 도구들의 집합이다. JMASS는 모델 개발자가 표준 인터페이스와 구조에 따라 모델을 개발 할 수 있도록 하는 소프트웨어 구조 모델을 정의하며, 모델링 · 실행 · 분석 단계를 지원하기 위한 8가지 서비스를 제공한다. 각 서비스를 살펴보면 다음과 같다.

- Atmosphere : 대기 서비스는 특정 위치 및 고도의 대기와 관련된 데이터(온도, 습도, 풍향 등)를 제공하여 항공기 · 미사일의 공기역학모델, 센서모델 등에서 사용될 수 있다.
- Terrain : 지형 서비스는 특정 좌표의 고도, 토양, 초목화 정보를 제공한다.
- Spatial : 공간 서비스는 공동 참조점으로 부터의 위치, 방향, 속도(가속도) 등의 공간 인식 기능을 제공한다.
- Scheduling : 스케줄링 서비스는 시뮬레이션에서 발생하는 Event 들을 시간에 따라 처리하는 기능을 제공한다.
- Journaling : 저널링 서비스는 시뮬레이션 진행 중에 플레이어(컴포넌트)의 특정 속성 값에 대한 출력을 지원한다.
- Message Logging : 메시지 로깅 서비스는 시뮬레이션 실행 중 메시지 출력을 지원한다.
- Message Transmission : 메시지 전달 서비스는 플레이어와 플레이어 사이에 Port를 통한 메시지 전달을 지원한다.
- Identification : 식별 서비스는 특정 플레이어와 컴포넌트를 식별하기 위해 고유한 ID를 제공한다.

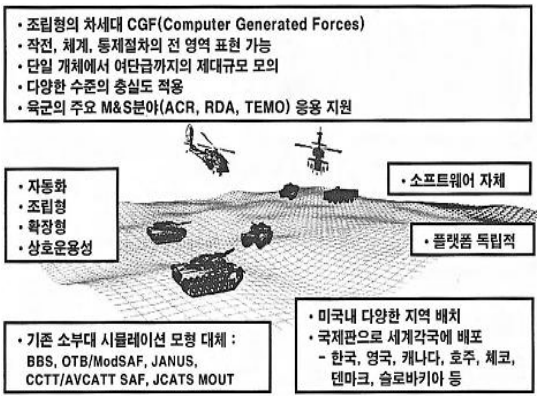

그림 1. OneSAF 특성[6]

2.2. OneSAF

OneSAF는 첨단개념연구(ACR), 훈련/연습 및 군사작전(TEMO), 연구개발 및 획득(RDA) 등의 다양한 M&S 영역에 대한 통합 시뮬레이션 서비스를 제공하기 위해 단일 체계가 아닌 시뮬레이션 수행을 위한 기반체계를 모두 포함하는 통합 시뮬레이션 체계로서 그 특성을 종합적으로 표현하면 그림 1과 같다.

OneSAF의 모델은 시뮬레이션을 위한 구성요소를 표현한 것으로 단일의 전투 플랫폼을 의미하는 개체 모델과 군대 조직을 의미하는 유닛 모델, 실 장비와 무기의 기본 특성 및 물리적 기능을 표현하는 물리 모델, 지휘통제 및 전술적 기능을 수행하는 개체 및 유닛의 활동을 의미하는 행위모델, 시뮬레이션과 관련된 지형 및 기상과 같은 환경 자료를 표현한 환경 모델이 있다. OneSAF는 다양한 시뮬레이션 목적에 필요한 도구 및 서비스를 계층구조의 PLAF(Product Line Architecture Framework) 아키텍처 형태로 정의하고 있다[16-17]. OneSAF의 지원 서비스는 다음과 같다.

- Composition : 조립 서비스는 컴포넌트들이 조립되어 특정 제품으로 구성되도록 지원한다.
- Environment Runtime : 환경 런타임 서비스는 OneSAF의 분산 환경 운용 시 공동 환경을 제공하기 위한 서비스로 주로 훈련과 같은 다중 사용자 환경에서의 OneSAF 운용을 지원한다.
- Environment Reasoning : 환경 추론 서비스는 지형 및 기상과 같은 환경 데이터와의 관계를 지원한다.
- GUI : GUI 서비스는 OneSAF 윈도우 양식과 디스플레이를 위한 기본 GUI 스타일을 지원한다.
- Plan View Display : 계획 뷰 디스플레이는 시뮬레이션 계획, 시뮬레이션 간 객체 및 유닛들의 형태와 상태, 지형 등을 시각적으로 표현한다.
- Data Collection : 데이터 수집 서비스는 사후 분석을 수행하기 위한 관련 데이터 수집을 지원한다.
- Simulation : 시뮬레이션 서비스는 시뮬레이션의 시간 진행과 사건관리, 난수 생성 등의 기본 시뮬레이션 기능에 대한 수행을 지원한다.
- Simulation Object Runtime Database : 시뮬레이션 객체 런타임 데이터베이스는 OneSAF의 분산 환경을 지원하기 위한 데이터베이스 및 데이터를 관리한다.
- Modeling : 모델링 서비스는 시뮬레이션 서비스에 대한 접근 관리 및 모델링 수행을 지원한다.
- System Repository : 시스템 저장소 서비스는 OneSAF 컴포넌트가 OneSAF 저장소의 데이터 접근이 가능하도록 지원한다.

2.3. WarpIV

미 WarpIV에서 개발된 WarpIV 시뮬레이션 엔진은 고성능 컴퓨팅과 상호 운용성 솔루션을 제공하며, 모델의 재사용 및 통합은 오픈소스 컴포넌트 기반의 Plug and Play 모델링 패러다임을 추구한다. WarpIV는 모델링 및 시뮬레이션을 분리하여, 모델링을 지원하기 위한 표준 서비스를 연구하는 OSAMS(Open System Architecture for Modeling and Simulation)와 시뮬레이션을 지원하기 위한 표준 서비스를 연구하는 OpenMSA(Open Modeling and Simulation Architecture)로 구분한다. 상기 설명한 WarpIV의 OSAMS와 OpenMSA를 지원하기 위해 제공되는 서비스는 다음과 같다.

- Standard Modeling Framework : WarpIV의 객체와 이벤트를 생성하고 정의한다.
- Process Model Behaviors : 모델 객체 및 이벤트에 대해 예약된 시간에 이산 시점에서의 처리 기능을 제공한다.
- Components and Composites : WarpIV의 기본이 되는 시뮬레이션 객체 및 컴포넌트를 구성하고 컴포넌트 간의 통신이 가능하도록 지원한다.
- Roll Back Framework : WarpIV는 병렬 및 분산된 시뮬레이션 이벤트에 대해 이벤트 처리 이전에 수정된 상태 변수, 메모리 할당과 해제에 대한 메모리 손상, 동적 생성된 Element에 대한 컨테이너 조작 등에 대해 roll back(undo 및 redo)을 제공한다.
- Run Time Class : 객체지향 프로그램의 데이터를

동적으로 표현한다. Run Time Class는 자주 사용되는 시나리오의 정의, 구성, 구조에 대해 XML로 읽기 쉬운 형식으로 시뮬레이션 결과를 조직하고 네트워크를 통해 구조화된 메시지 통신이 가능하게 한다.

- Distributed Simulation Management : 이 기종 시스템 및 상호운용 표준에 대한 Publish-Subscribe 서비스를 제공한다.
- Persistence : WarpIV에서 사용되는 객체를 네트워크 데이터베이스 서버에서 관리하기 위한 서비스를 제공한다.
- External Interface : 시뮬레이션 인터페이스 외부에서 시뮬레이션을 Kill, Pause, Resume 할 수 있는 시뮬레이션 컨트롤을 지원한다.
- High Speed Communication : 시뮬레이션 수행의 병렬처리를 제공한다.
- ORB(Object Request Broker) : UNIX 기반의 Client/Server 형식의 네트워크 통신을 지원한다.
- Configuration and Operation : 순차 및 병렬, 분산 실행에 대한 컨트롤을 파라미터를 통해 지원하며, 시뮬레이션의 통계 정보를 제공한다.

표 1은 상기 설명한 JMASS와 OneSAF, WarpIV에서 제공하는 서비스를 정리한 것이다. 분산 실시간 모의 엔진이 높은 수준의 재사용과 상호 운용성을 지원하기 위해서는 원칙적으로 표 1에서 도출된 바와 같은 모델 개발 · 실행 · 분석의 전체 과정을 지원하는 서비스를 제공해야 한다.

JMASS, OneSAF, WarpIV는 본 논문의 목적인 무기체계 효과도 분석을 지원하기 위한 모델링 및 시뮬레이션 엔진으로 활용될 때 다음과 같은 문제점을 지니고 있다.

- 무기체계 모델은 대개의 경우 인터페이스를 제외한 동작과 관련된 데이터 및 알고리즘을 보안상의 이유로 비공개 하는 경우가 일반적이다. JMASS 및 WarpIV는 무기체계의 인터페이스 이외의 행위에 대한 정보를 C++ 객체로 캡슐화 한 클래스의 재사용을 지원한다. 이는 소스코드를 통해 해당 무기체계 모델의 데이터 및 알고리즘이 공개되어 있음을 의미하며, 구현언어에 종속적이므로 모델러가 재사용 여부를 판단하기 위해서 클래스 구조 및 행위에 대한 소스코드 분석과 이해 과정을 필요로 한다.
- OneSAF는 JAR 형식의 컴포넌트를 통해 유사 무기체계 모델에 대한 실행 수준의 재사용을 지원하여 중요 데이터 및 알고리즘에 대한 보안을 유지 할 수 있다. 그러나 무기체계의 인터페이스를 제외한 행위에 대한 부가적인 정보를 제공하지 않고 있으므로 사용자가 재사용 여부를 판단하는데 어려움이 따른다.
- 무기체계 분석은 개별 무기체계의 성능지수를 제공하는 공학급 모델과 무기체계 모델이 활용 · 운용되는 교전급 모델의 연동이 필수적이다. 그러나 JMASS, OneSAF, WarpIV는 공학급/교전급 혼합 시뮬레이션(Hybrid Simulation)에 특화된 서비스를 지원하고 있지 않다.

표 1. JMASS, OneSAF, WarpIV

	JMASS	OneSAF	WarpIV
모델링	· Atmosphere · Terrain · Spatial · Journaling · Identification	· Composition · Environment Runtime · GUI · Modeling	· Standard Modeling Framework · Process Model Behaviors · Components and Composites · Run Time Class
시뮬레이션	· Scheduling · Message Transmission	· Plan View Display · Simulation · Simulation Object Runtime Database	· Distributed Simulation Management · External Interface · High Speed Communication · ORB · Roll Back Framework · Persistence
분석	· Message Logging	· Data Collection · System Repository	· Configurations and Operation

3. OpenSIM Service

OpenSIM은 무기체계 효과도 분석을 지원하기 위한 목적으로 설계된 통합 시뮬레이션 엔진이다. OpenSIM에서는 무기체계 및 운용과 관련되어 JMASS, OneSAF, WarpIV와 달리 공학급/교전급 모델을 작성하고 각 모델간의 혼합 시뮬레이션을 지원하기 위한 서비스 및 도구를 지원하는데 초점을 두고 있다. 또한, 유사 무기체계에 대한 재사용을 극대화하기 위해 모델의 물리 구조와 행위

구조를 하나의 단위로 묶어 캡슐화 하여 실행단위의 컴포넌트를 제작한 후 이를 재사용 단위로 사용한다. 또한, 상태 변화에 따른 모델의 행위를 정보화하여 사용자가 재사용 시 조회 할 수 있도록 지원한다. 본 장에서는 무기체계 분석을 위한 모델 개발 · 실행 · 분석의 전체 과정을 지원하는 OpenSIM의 서비스를 정의하고 정의된 서비스에 대해 C++로 명세한 API(Application Programming Interface)를 살펴본다.

3.1. OpenSIM Service Architecture

OpenSIM은 복합 무기체계 효과도 분석을 지원하기 위한 분산 실시간 체계 모의 엔진으로서 모델 개발 · 실행 · 분석의 전체 과정을 지원하기 위한 서비스를 제공한다. OpenSIM은 HLA(High Level Architecture) 기반 페더레이트, LVC 페더레이션과의 연동을 제공하기 위해 그림 2와 같은 계층화된 서비스의 집합을 제공한다. 제공되는 서비스는 3.2.장에 명세된 APIs를 통해 지원받을 수 있다.

- Internal High Speed Communication : Multi-core 기반의 HSC(High Speed Communication)을 제공한다.
- External Distributed Communications : TCP/IP, HTTPS 등의 네트워크 중심의 분산 컴퓨팅 환경을 조성하여 원하는 결과를 얻을 수 있도록 지원한다.
- Event Management Services : 무기체계 효과도 분석을 위한 교전급 모델 구현 시 이벤트 기반의 시뮬레이션 Scheduling을 지원하기 위한 서비스이다.
- Time Management Service : 혼합 시뮬레이션의 끊김 없는 진행을 위한 시간 진행 서비스를 지원한다.
- Standard Modeling Framework : OpenSIM에서는 상호 연동 및 모델간 재사용 증진을 위해 그림 3과 같은 표준 소프트웨어 모델 구조를 정의하였다. OpenSIM의 표준 소프트웨어 구조 모델은 시뮬레이션을 위한 최상위 레벨의 구성요소인 모델과 모델을 구성하는 하위 레벨의 구성요소로써 일반적으로 시뮬레이션 상에서 동작하는 독립적 객체를 표현하거나 물리적 구성요소를 나타내는 컴포넌트, 컴포넌트를 구성하는 서브 컴포넌트, 모델 또는 컴포넌트를 구성하는 하부 구성요소들의 연결 정보를 제공하는 채널로 이루어져 있다. 모든 컴포넌트는 상태 및 상태 전이에 따르는 행위 변화를 통해 묘사되며, 해당 정보는 재사용을 위해 정형화 된다.
- Distributed Simulation Management Services : OpenSIM에 HLA/RTI[18-19] 환경을 기반으로 분산 시뮬레이션 서비스를 제공한다.
- SOM/FOM Translation Services : HLA/RTI를 지원하기 위한 SOM, FOM[20]을 생성하고 관리한다.
- Resolution Services : 외부 시뮬레이터로부터 생성된 모델과의 해상도(resolution)가 상이할 경우, 이를 고려한 연동을 지원한다.
- Gateway Interface Services : 전송방식이 다른 HLA/RTI, SOA[21] 간의 데이터 전송을 위한 서비스를 제공한다.

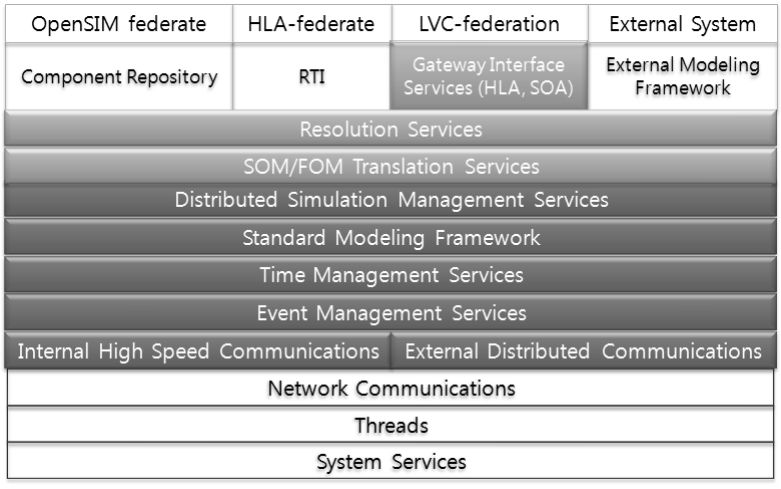

그림 2. OpenSIM Service Architecture

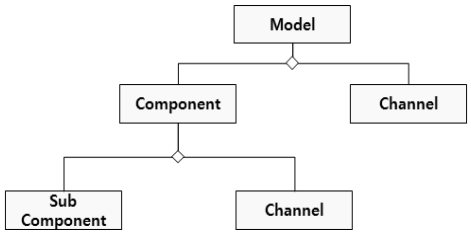

그림 3. OpenSIM 표준 소프트웨어 구조 모델

3.2. OpenSIM APIs

OpenSIM에서는 3.1.장에서 정의된 서비스에 대해 C++ 언어를 사용하여 API 형태로 명세하였다. 현재까지 정의된 API의 범위는 모델링, 시뮬레이션, 분석, 분산 및 병렬 시뮬레이션 지원 서비스와 웹 연동, 기타 유틸리티 서비스이다. 본 장에서는 간단한 공대지 미사일 분석 시뮬레이션을 통해 OpenSIM의 API 활용 예를 보인다. 3.2.1 - 3.2.6은 공대지 미사일의 효과도를 분석하기 위해 모델링된 Radar와 Missile로 구성된 전투기, 전투기가 운용되는 대기 환경, 전투기가 공격할 Target을 OpenSIM의 API를 통해 구현한 내용을 소개한다.

3.2.1. Atmosphere

Atmosphere는 주어진 고도와 위치에 따른 대기 환경 데이터를 모델이 참조할 수 있도록 지원한다. Atmosphere는 대기 환경의 상태를 갖는 Op_AtmosphereState 클래스와 특정 입력에 따른 대기 환경 데이터를 가지고 오는 Op_AtmosphereManager 클래스로 구성되어 있다. 사용자가 Op_AtmosphereQuery에 함수를 사용하여 경도와 위도, 고도, updateTime을 입력하면 입력한 장소의 대기 환경 데이터를 DB의 대기 환경 데이터 테이블로부터 얻어와 Op_AtmosphereState로 가져온다. 공대지 미사일의 효과도에 영향을 미치는 대기 환경을 구성하기 위해서는 표 2와 같은 Atmosphere 서비스를 활용할 수 있다.

3.2.2. Logging

Logging은 시뮬레이션 중에 메시지를 기록하여 편리하게 디버깅할 수 있도록 도움을 준다. 디버깅은 4가지 (Information, Warning, Error, Fatal) 형태로 볼 수 있고 로그 메시지를 GUI와 File의 2가지 매체를 통해 확인 가능하다. Log를 하기 위해서는 Log 할 모델의 함수 내에서 Op_Logger 클래스의 LogMessage() 함수를 사용하면 된다. Op_Logger 클래스에는 로그를 활성화 하는 멤버와 4가지 Log Level 멤버가 있어 GUI 상에서 체크하여 원하는 것을 볼 수 있다. 표 3은 Information과 Error 형태에 대한 Logging 사용 예제이다.

표 2. Atmosphere 사용 예제

```
double altitude = 5000.0;
double latitude = 21.0;
double longitude = 43.0;
TimeType updateTime = 1.0;

Op_AtmosphereState atmosState;
atmosState = Op_AtmosphereManager::Query(
altitude, latitude, longitude, updataTime);
```

표 3. Logging 사용 예제

```
Op_Logger::LogMessage("debug", "ASM",
                      Op_Logger::Information, 0.0,
                      "This is the output String Take");

Op_Logger::LogMessage("debug", "ASM",
                    Op_Logger::Error, 0.0, "%s %s %d");

// Log를 콘솔 화면에 출력
Op_Logger::SetOccurrence(true);
```

표 4. AirCraft Journaling 사용 예제

```
// AirCraft는 Op_ModelC을 상속 받아 생성되었다 가정
AirCraft *airCraft = new Aircraft();
Op_ASCIIJournaler *aJournaler = new Op_ASCIIJournaler();

aJournaler -> SettimeStep(0.1);
aJournaler -> AddAttribute("airCraft", "locationX", "X 좌표");
aJournaler -> AddAttribute("airCraft", "locationY", "Y 좌표");
aJournaler -> AddAttribute("airCraft", "locationZ", "Z 좌표");
aJournaler -> PrintJournal("비행좌표", ",");
```

3.2.3. Journaling

Journaling은 시뮬레이션 중에 선택 되어진 속성 값을 추출하여 특정 형식의 파일에 저장한다. 저장되는 파일은 ASCII 형식의 .ajd 파일과 Binary 형식의 .bjd 파일, MATLAB 형식의 .mat 파일이 있다. Journaling은 모델 개발자가 모델이 정확하게 구현되었는지 검증하기 위해서 사용하며 분석가들은 특별한 모델의 행동에 대해 잘 알기위해 사용한다. OpenSIM의 Journaling은 기본구조를 정의한 Op_Journaler 클래스와 이를 상속받는 Op_ASCIIJournaler, Op_BinaryJournaler, Op_MATLABJournaler 클래스를 사용하여 3가지 형식의 파일로 출력한다. 표 4는 전투기의 X, Y, Z 좌표에 대하여 ASCII 형식의 Jour-

naling Service를 사용한 예제이다.

3.2.4. Basic Modeling

OpenSIM의 표준 모델링 프레임워크는 DEVS[22] 형식론을 통해 실현할 수 있다. 그림 3에서 설명한 OpenSIM 모델프레임 워크에서 Sub-Component는 DEVS의 원자 모델로, Component는 DEVS의 결합모델로 대응되며, Channel은 Port로 실현될 수 있다. 또한 OpenSIM의 모든 Sub-Component의 행위는 DEVS의 상태와 상태 천이를 통해 모델링 되도록 한다. OpenSIM에서는 DEVS 형식론에 따르는 모델을 생성하고 이에 대한 시뮬레이션을 지원하는 DEVSim++ 엔진[23] 을 바탕으로 Basic Modeling 서비스를 구현하였다. Op_Model 클래스는 OpenSIM 모델을 구현하는데 필요한 기본 구조로서 Op_ModelA와 Op_ModelC에서 공통으로 사용하는 입/출력 이벤트 집합 및 시뮬레이션에 필요한 시간 정보를 관리하는 변수/함수 등의 원자 모델과 결합 모델 모두에 해당하는 특성들이 정의되어 있는 원자 모델과 결합 모델의 부모 클래스이다. Op_ModelA는 원자 모델의 기본 알고리즘을 구현해 놓은 것으로 입/출력 이벤트 집합은 Op_Model을 상속받아 구현하고, 외부 상태 천이 함수와 내부 상태 천이 함수, 출력 함수, 시간 전진 함수를 구현하면 된다. Op_ModelC는 결합 모델의 기본 알고리즘을 구현한 것으로 입/출력 이벤트 집합과 모델 우선순위 결정 함수는 Op_Model을 상속받아 구현하고, 외부 입력 연결 관계와 외부 출력 연결 관계, 내부 연결 관계를 구현하면 된다. 표 5는 OpenSIM의 모델링 프레임워크에 따라 AirCraft을 생성하는 예제이다.

표 5. AirCraft 모델 생성 예제

```
// Radar ModelA 정의 (Radar.h)
// 사용자는 외부 상태 천이, 내부 상태 천이, 출력
// 시간 전진 함수를 .cpp에 구현하여야 한다.
class Radar : public Op_ModelA {
public:
  Radar();
  Radar(std::string name);
  ~Radar();

  // 외부 상태 천이 함수
  virtual bool ExTransFn(const Message &);
  // 내부 상태 천이 함수
  virtual bool IntTransFn();
  // 출력 함수
  virtual bool OutputFn(Message &);
  // 시간 전진 함수
  virtual TimeType TimeAdvanceFn();

  enum Status {WAIT, SEND, DETECTION};

protected:
  int m_Status;
};

// Missile ModelA 정의 (Missile.h)
class Missile : public Op_ModelA {
public:
  Missile();
  Missile(std::string name);
  ~Missile();

  virtual bool ExTransFn(const Message &);
  virtual bool IntTransFn();
  virtual bool OutputFn(Message &);
  virtual TimeType TimeAdvanceFn();

  enum Status {WAIT, MOVE, SEND};

protected:
  int m_Status;
};

// Radar와 Missile ModelA를 결합하여
// AirCraft ModelC 구현 (AirCraft.cpp)
AirCraft::AirCraft() {

  SetName("AirCraft");
  Model *radar, *missile;
  radar = new Radar("Radar");
  missile = new Missile("Missile");

  // 하위 컴포넌트 등록
  AddComponent(2, radar, missile);

  // AirCraft Ports 등록
  AddInPorts(1, "from_monitor");
  AddOutPorts(2, "to_no_detect", "to_detect");

  // AirCraft와 Radar, Missile Coupling
  AddCoupling(this, "from_monitor", radar, "from_aircraft");
  AddCoupling(radar, "to_aircraft", this, "to_no_detect");
  AddCoupling(radar, "to_missile", missile, "from_radar");
  AddCoupling(missile, "to_aircraft", this, "to_detect");
```

3.2.4. Utilities

OpenSIM은 모델링 및 시뮬레이션 수행의 편의를 돕고자 표 6과 같이 Data 구조와 수학 알고리즘, 난수 및 분포 등의 Utilities를 제공한다.

표 6. Utilities 분류

Data 구조	Queue, List
수학 알고리즘	Matrix, Runge-Kutta
난수 및 분포	Normal, Gaussian, Uniform, Exponential
기타	Lanchester Equations Kalman Filter

표 7. Utilities Lanchester Equations 사용 예제

```
Op_Lanchest lanchest;

lanchest.Lanchest(0.0, airCraft_Strength airCraft_Attrition,
                  target_Strength, target_Attrition);

// AirCraft와 Target의 상태를 Update
lanchester.Update(timeStep, airCraft_Strength,
                  target_Strength);

lanchester.GetTime();
cout << "AirCraft = " << airCraft_Strength;
cout << "Target = " << target_Strength;
```

공대지 미사일 효과도 분석 시 Target의 폭파정도 및 아군 AirCraft의 상태를 모델링하기 위해서는 표 7과 같이 OpenSIM의 Lanchester Equations 서비스를 활용 할 수 있다.

3.2.5. Analysis

OpenSIM에서는 시뮬레이션의 수행에 대한 통계를 분석하는 Op_Statistics 클래스와 System 사용에 대해 분석하는 Op_Report 클래스를 지원한다. Op_Statistics는 시뮬레이션의 평균값과 표준편차, 신뢰도를 제공하며, Op_Report는 시뮬레이션 평균 수행 시간, CPU 사용량, 실 메모리 사용 현황, 네트워크 이용률을 확인 할 수 있도록 지원한다. 표 8은 공대지 미사일 효과도 분석 시 평균 시뮬레이션 시간, CPU 및 메모리 사용 현황, 평균, 분산 등의 통계치를 얻기 위해 OpenSIM의 Analysis 서비스를 활용한 예를 보인다.

표 8. Analysis 사용 예제

```
// 시뮬레이션 통계 분석
Op_Statistics ASM_Statistics;

double confidenceInterval = 0.95;
double mean, variance, standardDeviation;

// 시뮬레이션에서 얻고자 하는 평균
mean = ASM_Statistics.Mean(&data, 1000);

// 시뮬레이션에서 얻고자 하는 분산
variance = ASM_Statistics.Variance(&data, mean);

// 시뮬레이션에서 얻고자 하는 표준편차
standardDeviation =
ASM_Statistics.StandardDeviation(&data, variance);

// 시뮬레이션 수행 신뢰도
ASM_Statistics.Confidence(&data,
confidenceInterval, standardDeviation, mean);

// BatchMean 기법 사용
ASM_Statistics.BatchMean(&data, mean, 0.1, 10);

// 시뮬레이션 시스템 분석
Op_Report ASM_report;

// 1000번 시뮬레이션 수행에 대한 평균 시간
ASM_report.GetAverageSimulationTime(1000);

// 시뮬레이션 수행 시 CPU 사용 현황
ASM_report.GetCPUUtil();

// 시뮬레이션 수행 시 메모리 사용 현황
ASM_report.GetMemoryUsage();
```

4. 결 론

통합 시뮬레이션 엔진은 사용자와의 상호작용과 프로그래밍 작업의 복잡도 해결을 위하여 분석 대상 시스템의 모델링 · 실행 · 분석을 지원하기 위한 도구와 서비스, 인터페이스를 제공해야 한다. 그러나 국내는 모델의 개발부터 시뮬레이션의 수행까지 전단계의 도구와 서비스, 인터페이스를 제공하는 분산 실시간 모의체계 엔진에 대한 연구 및 개발이 미흡한 실정이다. 본 논문은 무기체계 효과도 분석을 지원하기 위한 통합 시뮬레이션 엔진인 OpenSIM을 소개하고 OpenSIM의 서비스 범위 및 내용을 정의하였다. 또한, OpenSIM에서 제공하는 서비스를 이용한 공대지 미사일의 모델링 과정을 통해, 구성된 API가 효과적으로 프로그래밍 작업을 지원할 수 있음을 보였다. 향후에는 다양한 무기체계 모의실험에 개발된 서비스를 적용하여 OpenSIM의 서비스를 검증하고 추가적으로 필요한 서비스를 식별할 예정이다. 또한, 서비스를 자동적으로 생성하고 수행할 수 있는 OpenSIM의 도구들을 개발할 예정이다.

참 고 문 헌

1. 장상철, 정상윤 (2002), "전투실험 활성화를 위한 모의체계 발전방안", 국방정책연구, pp. 137-179.
2. 손미애, 유승근, 박태유 (2002), "국방 M&S 활성화 연구: 차세대 획득체계(JMASS) 도입·활용방안 연구", 한국국방연구원.
3. Robert J. Meyer (2001), "Joint Modeling And Simulation System, What it does,…and What it doesn't," Simulation Interoperability Standards Organization (SISO), 01S-SIW-117.
4. Jim Russell (2003), "An Overview of Modeling Digital Communications Networks with JMASS and DIS", Simulation Interoperability Standards Organization (SISO), 03S-SIW-024.
5. Bob Mayer (2003), "Joint Modeling and Simulation System(JMASS), Transition from Development to Sustainment", Simulation Interoperability Standards Organization (SISO), 03S-SIW-210.
6. 문형곤, 유승근, 김태홍 (2008), "OneSAF 모형 도입, 실용화(I)", 한국국방연구원.
7. 방위사업청 (2007), "OneSAF 개념 소개", 사업관리본부 자료실.
8. OneSAF Public Site, "OneSAF Papers and Presentations", http://www.onesaf.net/
9. WarpIV Public Site, "WarpIV Papers and Presentations", http://www.warpIV.com/
10. 윤상윤 (2004), "국방 모델링/시뮬레이션 현 실태 진단 및 발전 방안", 국방정책 연구, pp. 139-143.
11. 임명식, 김기형 (2003), "컴퓨터 네트워크 가상 실습을 위한 컴포넌트 기반의 시뮬레이터 설계 및 구현", 한국시뮬레이션학회논문지, 제12권, 제1호, pp. 1-10.
12. 김주영, 김탁곤 (2006), "DEVS를 이용한 워게임 시뮬레이터 자동 합성 방법론", 한국시뮬레이션학회 학술대회 논문집, 춘계학술대회, pp. 67-72.
13. 국방 M&S 기술특화 센터 (2007), "분산 실시간 체계모의 엔진 연구(MS-41)-RFP", 국방 M&S 기술특화 센터 RFP.
14. 윤상윤, 한경섭 (2005), "우리 군의 M&S 비전과 과제", 국방정책연구, pp. 2-36.
15. 김태섭, 장희정, 이재민, 이강선 (2010), "무기체계 분석을 위한 모의엔진 아키텍처 연구", 한국시뮬레이션학회, 한국시뮬레이션학회논문지, 제19권 제2호, pp. 51-62.
16. 김동윤, 백두권, 박수용, 김상수, 조은애 (2007), "소프트웨어 기술 개발 전략 연구 : 소프트웨어공학 분야", 한국소프트웨어진흥원, pp. 1-120.
17. 장수호, 라현정, 김수동 (2005), "프로덕트라인 아키텍쳐의 실용적 설계 기법", 한국정보과학회, 정보과학회논문지 : 소프트웨어 및 응용, 제32권 제 3호 2005.3, pp. 163-172.
18. 홍정희, 안정현, 김탁곤 (2008), "IEEE 1516 HLA/RTI 표준을 만족하는 시간 관리 서비스 모듈의 설계 및 구현", 한국시뮬레이션학회, 한국시뮬레이션학회논문지, 제17권 제1호 2008.3, pp. 43~52.
19. 현세웅, 윤석준 (2009), "HLA-RTI에 기반 한 비행시뮬레이션에 관한 연구", 한국항공우주학회, 한국항공우주학회지, 제 37권 제6호 2009.6, pp. 602-608.
20. IEEE (2010), "IEEE Standard for Modeling and Simulation(M&S) High Level Architecture(HLA)--Object Model Template (OMT) Specification", IEEE, pp. 1-112. http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5557731
21. 한국정보보호진흥원 (2007), "소프트웨어 아키텍처 기반 설계 모델 및 명세기법 개발", KISA-WP-2007-0045, 한국정보보호진흥원, pp. 1-132.
22. 황도성, 박성봉, 안명수, 김탁곤 (2008), "개선된 DEVS 모델을 이용한 전략 시뮬레이터 개발", 한국시뮬레이션학회 학술대회논문지, 추계학술대회, pp. 22-27.
23. 안명수, 박성봉, 김탁곤 (1996), "DEVS에 기반한 분산 시뮬레이션 환경 D-DEVSim++의 설계 및 구현", 한국시뮬레이션학회, 한국시뮬레이션학회논문지, 제5권 제2호 1996. 12, pp. 41~58.

김 태 섭 (karma1209@mju.ac.kr)

2009 명지대학교 컴퓨터공학과 학사
2009 명지대학교 컴퓨터공학과 석사과정

관심분야 : 컴퓨터시뮬레이션, 국방 M&S, 무기체계 효과도 분석

박 준 호 (allin917@mju.ac.kr)

2010 명지대학교 컴퓨터공학과 학사
2010 명지대학교 컴퓨터공학과 석사과정

관심분야 : 컴퓨터시뮬레이션, 국방 M&S

김 현 휘 (unghwi@mju.ac.kr)

2010 명지대학교 컴퓨터공학과 학사
2010 명지대학교 컴퓨터공학과 석사과정

관심분야 : 컴퓨터시뮬레이션, 국방 M&S

박 찬 종 (pcj0824@mju.ac.kr)

2010 명지대학교 컴퓨터공학과 학사과정

관심분야 : 컴퓨터시뮬레이션, 국방 M&S

이 강 선 (ksl@mju.ac.kr)

1992 이화여자대학교 전자계산학과 학사
1994 이화여자대학교 전자계산학과 석사
1998 미) University of Florida, 박사
현재 명지대학교 컴퓨터공학과 주임교수

관심분야 : 컴퓨터시뮬레이션, 국방 M&S, 웹서비스