플러그인 방식을 이용한 연동 가능한 시뮬레이션 프레임워크와 환경
Framework and Environment for Interoperable Simulation Using Plug-In Method

배장원 김탁곤
Jangwon Bae Tag Gon Kim
한국과학기술원 전자전산학과 전기 및 전자공학 전공, 대전 305-701
042-350-5454, jwbae@smslab.kaist.ac.kr

ABSTRACT

Interoperability and reusability have been hot issues in the area of modeling and simulation. Reusability means joining an existing simulator to a new simulation with a little or no modification. Interoperability means joining many simulators in a unified simulation and simulating them together with data exchanging and time synchronization.

This paper proposes a framework which can support the reusability of model-level and provide an interoperable environment at the same time. And using Plug-In method, models which have already participated to the simulation can be easily exchanged to other models. The framework provides an interface based on an atomic model in DEVS formalism. Using that interface, model can be easily plugged in to simulation and reusability of model-level can be increased. Interoperability in the framework can be achieved by synchronizing time and exchanging data according to routing information. This paper has an experiment about interoperation between heterogeneous simulators. The result shows us that the correct work in an interoperation simulation

1. 서 론

모델링 및 시뮬레이션(Modeling & Simulation) 영역에서 재사용성(Reusability)과 연동성(Interoperability)은 기존의 시뮬레이터를 사용할 수 있다는 점과 여러 시뮬레이터를 이용해 시뮬레이션을 할 수 있다는 측면에서 많은 주목을 받고 있다[1]. 재사용성이 증가하게 되면, 새로운 모델을 개발하는데 있어서 비용과 시간을 절감하는 효과를 갖게 된다[2]. 특히 모델의 재사용성에 대한 연구는 미국방성에서 활발하게 논의되고 있다[3]. 시뮬레이터간의 연동은 시뮬레이션의 표현 범위를 증가시켜 주는 역할을 한다. 즉, 서로 다른 특성을 지니는 시스템들을 하나의 시뮬레이션의 부분 요소로 사용한다. 이런 방법을 통해 좀 더 자세한 시뮬레이션이 가능해진다.

같은 시뮬레이션을 수행해도 목적에 따라 시뮬레이션에 참여하는 모델의 특성이 바뀐다. 예를 들어, 미사일로 적함을 격추시키는 시뮬레이션을 실행할 때, 미사일의 움직임을 바꾸면서 시뮬레이션을 실행하게 된다. 이런 경우 기존의 방법에서는 모델의 내용이 변경되면 시뮬레이터 내부의 해당 모델의 내용을 수정한 후에, 다시 컴파일하고 링크한 후에 시뮬레이션을 재개하는 작업이 필요하다. 하지만 플러그인(Plug-In) 방식을 사용하면, 위와 같은 작업을 수행하지 않고, 모델을 바로 교환하여 시뮬레이션이 재개하는 것이 가능하다[4]. 이러한 플러그인 방식을 사용하기 위해서는 모델과 시뮬레이터 엔진 사이의 미리 정의된 인터페이스를 사용하는 것이 필요하다.

본 논문에서는 모델의 재사용과 연동을 지원하고, 플러그인 방식의 모델교환이 가능한 프레임워크와 환경을 제안 한다. 제안된 프레임워크에서는 다른 모델 간의 연동을 위해 모델 간의 시간 동기화와 데이터 교환을 지원하고 있다. 그리고 플러그인 방식을 위한 인터페이스는 DEVS(Discrete Event System) 형식론을 사용하여 구현하였다. 사용자는 이 프레임워크를 사용해 DEVS 형식론의 원자 모델에 대한 지식만으로 다른 모델과 연동 시뮬레이션이 가능하다.

본 논문의 구성은 다음과 같다. 2 장에서는 DEVS 형식론에 대해서 알아 본다. 3 장에서는 제안한 프레임워크의 구조에 대해서 알아보고, 4 장에서는 사용자 모델 개발 과정을 살펴본 후에, 5 장에서 실험에 대한 내용과 결과를 언급하고, 6 장에서 결론을 맺는다.

2. DEVS 형식론

제안된 프레임워크의 인터페이스는 DEVS 형식론의 원자 모델을 이용하여 구현되어 있다. DEVS 형식론은 이산 사건 시스템을 객체 단위로 나누어 계층적이고 모듈러한 방법으로 표현할 수 있는 집합론에 근거한 수학적인 틀이다. DEVS 형식론은 시스템 기본적인 구성 요소의 동작 특성을 나타내는 원자 모델과 내부 모델들의 연결관계를 나타내는 결합 모델로 이루어져 있다[5].

2.1.1 원자 모델

원자 모델(Atomic Model)은 DEVS 형식론을 구성하는 가장 기본적인 단위로써 시스템의 행동을 기술하는 모델이다. 원자 모델의 수학적인 표현은 다음과 같다.

$M = \langle X, Y, S, \delta_{ext}, \delta_{int}, \lambda, ta \rangle$
X: 이산사건 입력 집합
Y: 이산사건 출력 집합

S: 모델 상태의 집합
δ_{ext} : Q × X → S: 외부 상태 천이 함수
Q={(s,e)|s∈S, 0≤e≤ta(s)}: M 의 모든 상태
δ_{int} : S → S: 내부 상태 천이 함수
λ : S → Y: 출력 함수
ta: S → $R_{0,\infty}^{+}$: 시간 진행 함수

2.1.2 결합 모델

결합 모델(Coupled Model)은 여러 모델을 내부적으로 연결하여 만든 모델이다. 내부 구성요소가 되는 모델은 원자 모델과 결합 모델이 모두 가능한데. 이러한 내부 모델들을 계속 결합하여 더욱 큰 시스템을 표현할 수 있다. 다음은 결합 모델의 수학적 명세이다.

CM = <X, Y, {M_i}, EIC, EOC, IC, SELECT>
X : 이산사건 입력 집합
Y : 이산사건 출력 집합
{M_i} : 컴포넌트 모델들의 집합
EIC : 외부 입력 연결 관계
EOC : 외부 출력 연결 관계
IC : 내부 연결 관계
SELECT : $2^{\{Mi\}}$ - Ø → M_i: 같은 시각에 존재하는 사건을 발생하는 모델들에 대한 선택 함수

3. 제안된 프레임워크

3.1 제안된 프레임워크 개념도

제안된 프레임 워크의 개념도는 아래의 그림 1 과 같다.

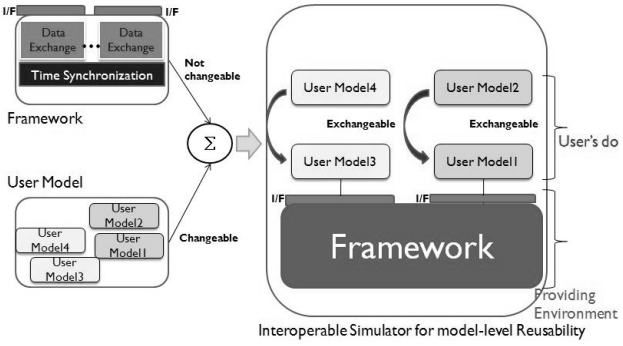

그림 1. 제안된 프레임워크 개념도

제안된 프레임워크는 모델 수준의 재사용성과 함께 연동을 제공하는 시뮬레이터를 사용자에게 제공하는데 목적이 있다. 제안된 프레임워크는 시뮬레이션 중 변경하지 않는 부분을 사용자에게 제공하고, 목적에 따라 변경될 수 있는 부분을 사용자가 직접 구현하여 하나의 완성된 시뮬레이터를 제작하게 된다. 또한 사용자는 프레임워크의 인터페이스에 맞추어 모델을 개발하여, 플러그인 방식을 통해 모델끼리 쉽게 교환할 수 있다. 사용자는 목적에 맞는 모델을 구현한 뒤, 제안된 프레임워크를 이용하여 연동 시뮬레이션을 실행할 수 있다.

3.2 제안된 프레임워크 구조

제안된 프레임워크의 구조는 그림 2 와 같다.

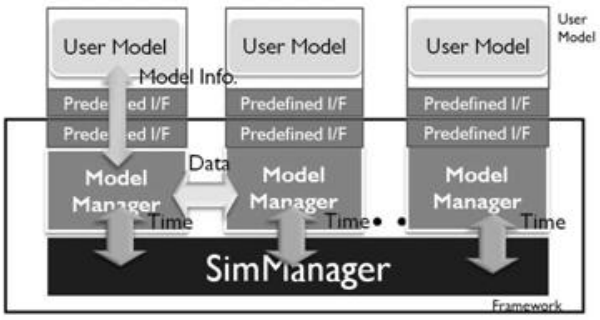

그림 2. 제안된 프레임워크의 구조

제안된 프레임워크의 내부는 하나의 시뮬레이션 관리자(Simulaton Manager)와 다수의 모델 관리자(Model Manager)로 이루어져 있다. 제안된 프레임워크에서는 연동을 위해 필요한 데이터 교환(Data Exchange)과 시간 동기화(Time Synchronization)와 관련된 작업을 한다. 시간 동기화는 전체 시뮬레이션의 시간을 관리하는 작업이므로 시뮬레이션 관리자가 단독으로 관리한다. 데이터 교환은 모델끼리 직접 데이터를 주고 받을 수 있도록 모델 관리자가 각각의 모델이 사용하는 데이터 정보를 통해서 사용한다. 다음 절에서 데이터 교환과 시간 동기화가 어떻게 이루어지는 지에 대해서 설명한다.

3.3 데이터 교환

제안된 프레임워크에서의 데이터 교환은 시뮬레이션 관리자가 모델 관리자에게 시뮬레이션이 시작할 때 보내는 연결 정보(Routing Info)를 바탕으로 이루어진다. 모델 관리자는 이 연결 정보를 바탕으로 데이터를 다른 모델로 직접 보낸다.

3.3.1 연결 정보

연결 정보란 시뮬레이션 관리자에서 관리하는 특정 데이터의 어느 모델에서 나와서 어느 모델로 들어가는 정보를 나타낸다. 사용자는 시뮬레이션을 실행하기 전에 하나의 모델에서 사용하는 데이터를 기술하는 LDS(Local Data Set)와 전체 시뮬레이션 에서 사용하는 데이터를 기술하는 GDS(Global Data Set)을 기술 해줘야 한다. LDS 는 프레임워크에서 제공하는 형식에 맞추어 작성 하고, GDS 는 LDS 의 정보를 합친 내용으로 작성해준다.

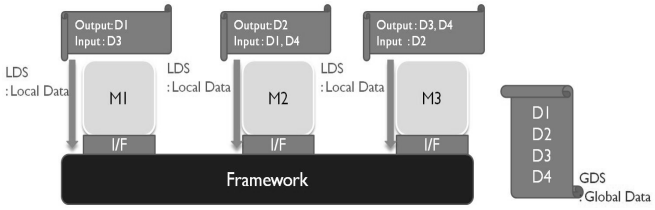

그림 3. LDS 와 GDS

시뮬레이션 매니저는 시뮬레이션을 시작하면 LDS 와 GDS 파일을 읽어, 연결 정보를 생성하고, 모델 매니저에게 연결 정보를 전달해준다. 모델 매니저는 연결 정보를 바탕으로 모델의 입출력 데이터를 전달한다.

3.4 시간 동기화

제안된 프레임워크의 시간 동기화는 모델 관리자가 연결된 모델의 다음 스케줄을 받아, 그 정보를 시뮬레이션 관리자에게 시간 진행 요청(TimeAdvanceRequest, TAR)의 형태로 요청하면, 시뮬레이션 관리자가 모든 모델 관리자의 시간 진행 요청 값을 참조하여 전체 시뮬레이션 시간 진행하면서 동기를 맞추고 있다. 제안된 프레임워크의 시간 동기화 알고리즘은 그림 4 와 같다.

```
SimManager
  Inputs :
    TAR(tn)  message.
  Outputs :
    TAG(*, tn) message
  variables :
    tn   // Time of Next event
    m   // ModelManger
Intiallization :
     Update  all TAR message(Tn) of ModelManagers
When all TAR message (tn) received
    tn = min { Tn1, Tn2, Tn3, .. ,Tnk}
    if (tn  equal to 'infinity')
        End simulation
    else
    send  TAG(*,tn) message to ModelManager m  of tn
```

그림 4. 시간 동기화 알고리즘

4. 사용자 모델 개발

사용자가 프레임워크를 이용해서 시뮬레이션 하기 위해서는 공통된 인터페이스를 사용하여 모델을 개발해야 한다. 세부 절에서는 사용자 인터페이스에 사용자 모델의 개발 절차에 대해서 알아본다.

4.1 사용자 모델 인터페이스

제안된 프레임워크의 인터페이스는 어떤 모델에도 적용가능 하도록 만들 수 있으나, 현재 프레임워크에서는 크게 DEVS 모델 인터페이스와 MATLAB 모델 인터페이스가 구현되어 있다.

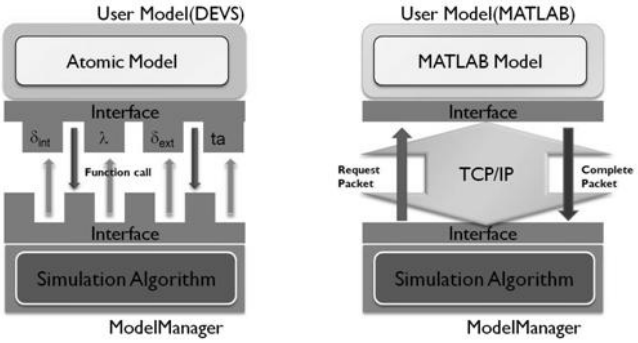

그림 5. 사용자 모델 인터페이스

모델과 일대일로 연결되는 모델 관리자는 모델과 그림 5 와 같은 인터페이스를 통해 데이터를 주고 받는다. DEVS 모델은 모델 매니저가 모델 내부의 DEVS 원자 모델의 함수를 직접 호출한 후, 리턴 값을 받고, MATLAB 모델은 모델 매니저와 TCP/IP 통신을 맺은 후, 패킷을 주고 받으며 통신한다.

4.2 사용자 모델 개발 절차

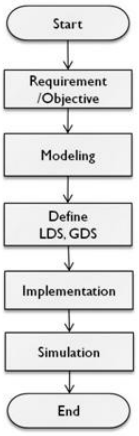

그림 6. 사용자 모델 개발 절차

사용자 모델을 개발하는 절차는 그림 6 과 같다. 우선 모델링 할 시스템에 대해서 분석을 한다. 분석한 결과를 이용해 DEVS 형식론을 이용하여 모델링을 수행한 후, 각 모델이 사용할 데이터와 전체 시뮬레이션에서 사용할 데이터를 정의하고 주어진 형식에 맞추어 LDS 와 GDS 를 작성한다. 모델링 결과를 이용하여 DEVS 모델은 DLL(Dynamuc Link Library)형식으로 구현하고, MATLAB 모델은 Simulink 를 이용하여 구현한다. 구현한 결과물을 프레임워크에서 제공하는 컨트롤러를 사용하여 시뮬레이션에 참여시킨 후, 시뮬레이션을 시작한다. 컨트롤러는 현재 시뮬레이션에 참여한 모델과 데이터에 대한 정보와 시뮬레이션의 상태 정보 등을 표시해 준다.

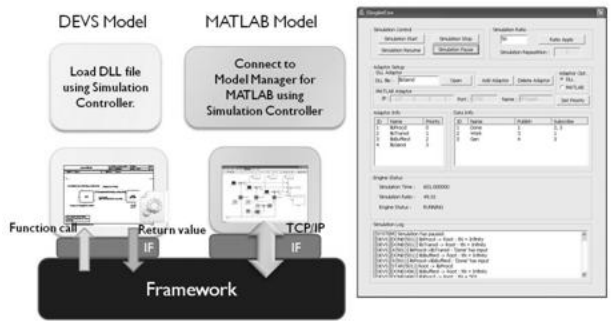

그림 7. 프레임워크를 이용한 시뮬레이션

5. 실험 : 전함의 방어체계

제안된 프레임워크를 검증하기 위해서 그림 8 과 같은 다른 모델간의 연동 시뮬레이션을 실행하였다.

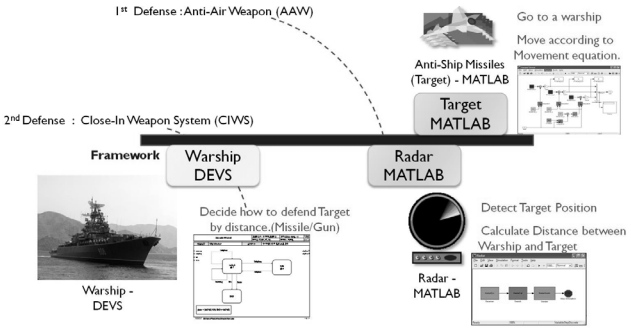

그림 8. 전함의 방어 체계

이 실험의 시나리오는 전함을 향해 날아오는 목표물이 존재하고 전함의 레이더에서는 목표물의 위치를 탐지하여 거리를 계산한다. 전함은 거리정보를 이용해 원거리 시에는 대공 미사일(AAW)를 발사하고, 근거리 시에는 근접 무기 체계(CIWS)를 발동한다. 이러한 방어 체계를 이용하여 전함의 생존률을 높이는 것이 목적이다. 특히 이번 실험에서는 대공 미사일의 표 1 의 특성에 따라 전함의 생존률이 달라지는 것을 실험 한다.

운동방정식을 사용하는 목표물과 거리를 계산하는 레이더는 MATLAB 모델로 구현하고 전함은 DLL 모델로 구현하였다.

표 1. 대공미사일의 특성

	Hit Power	Accuracy
Alternative 1	50%	100%
Alternative 2	75%	75%
Alternative 3	100%	50%

실험에서는 표 1 과 같은 대공 미사일의 특성을 적용하여 시뮬레이션을 진행하였고 결과는 그림 9 와 같다.

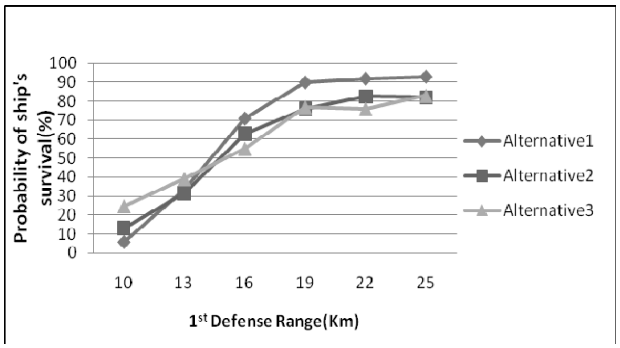

그림 9. 전함의 방어 체계 실험 결과

그림 9 를 통해 실험 결과가 1 차 방어 거리가 가까울수록 파워가 좋은 대공 미사일을 사용한 전함이 생존률이 높고, 1 차 방어 거리가 멀수록 정확도가 좋은 대공 미사일을 사용한 전함이 좋은 생존율을 보이는 경향이 있는 것을 확인해 볼 수 있다. 이런 실험을 했던 다른 연구와 동일한 경향을 보이는 것을 볼 때, 연동 실험이 성공적으로 행해진 것을 확인할 수 있다[7].

6. 결론

본 논문에서는 모델 수준의 연동을 지원하며, 플러그인 방식을 통하여 모델의 교환을 쉽게 할 수 있는 프레임워크를 디자인하고 구현하였다.

제안된 프레임워크는 사용자 모델을 공통된 인터페이스를 통해 구현하여, 플러그인 방식으로 시뮬레이션의 목적에 따라 모델을 쉽게 교환하는 것이 가능하다. 또한 제안된 프레임워크 내부에서 연동 환경을 지원하고 있다. 연동을 지원하기 위해서 프레임워크 내부에서 사용하는 데이터 교환과 시간 동기화 알고리즘을 제시하고 구현하였다. 이런 기능을 사용하여 사용자는 목적에 맞는 시뮬레이션을 더욱 쉽게 실행할 수 있다.

참고 문헌

[1] Ören T.I. and Zeigler B.P., "Concepts for Advanced Simulation Methodologies," *Simulation*, vol. 32, no. 3, pp. 69-82, 1979.

[2] Ören T.I., " Future of Modelling and Simulation: LDSe Development Areas," *Proceedings of the 2002 Summer Computer Simulation Conference*, pp. 3-8, 2002.

[3] Davis K. P. and Anderson A. R., "Improving the Composability of Department of Defense Models and Simulations," *RAND Technical report*, 2003.

[4] Jan Himmelspach and Adelinde M. Uhrmacher ,"Plug'n simulate," *Proceedings of the 40th Annual Simulation Symposium (ANSS'07)*, pp.137 -143, Mar. 2007.

[5] Bernard P. Zeigler, Herbert Praehofer and Tag Gon Kim, *Theory of Modeling and Simulation*, ACADEMIC PRESS, 2000.

[6] ChangHo Sung, JeongHee Hong and Tag Gon Kim, "Interoperation of DEVS Models and Differential Equation Models using HLA/RTI," *Proceedings of the 2009 Spring Simulation MultiConference*, pp. 387 – 392, 2009.