한국시뮬레이션학회 논문지
Vol. 27, No. 1, pp. 91-99 (2018. 3)
http://doi.org/10.9709/JKSS.2018.27.1.091
ISSN 1225-5904

이벤트 제어 모델 템플릿을 사용한 모델 라이브러리 기반 DEVS 시뮬레이션 환경의 확장성 개선

권세중† · 이준희 · 최창범 · 김탁곤

Improving Extensibility of DEVS Simulation Environment with Model Base by using Event Control Model Templates

Se Jung Kwon† · Jun Hee Lee · Changbeom Choi · Tag Gon Kim

ABSTRACT

Discrete event simulation environments often need to be modified because additional questions to systems may become apparent while observing the simulation results repeatedly. It leads to increasing development budget and depreciating the effectiveness of the environment. To avoid the modifications and to generate the altered results, this paper applies an Event Control Model (ECM) with control functions that modulate, delete and generate the events at the simulation time. In addition, this paper suggests an easier approach for domain-users, who do not want to program at source code level, by using ECM templates. The simulators with the ECMs can have better extensibility because it becomes more adaptable to possibly unanticipated changes. It prevents increasing development costs due to modifications or development of new models by M&S experts, and it provides a new alternative step to domain users. To support the effectiveness of this approach, this paper describes a relevant example, which is composed of an initial simulation model based on our empirical studies. It will show that there exist the uncountable benefits because the existing simulator is reused by domain users without new projects.

Key words : DEVS formalism, Event-based simulation, Event control model, Modeling and simulation

요 약

도메인 사용자에게 배포된 이산 사건 시스템의 시뮬레이터는 시뮬레이션 결과를 분석하는 과정에서 발생하는 요구사항의 변화로 인해 수정될 필요가 생긴다. 이로 인해 예상치 못한 개발 비용이 추가적으로 발생하고 시뮬레이션 환경의 효용성이 떨어진다. 본 논문은 이런 문제를 해결하기 위해 이벤트를 변조/삭제/생성하여 이전과 다른 결과를 발생시키는 제어 함수를 지닌 이벤트 제어 모델 템플릿을 제안한다. 이벤트 제어 모델은 이산 사건 시뮬레이션을 이산화된 상태 변화와 대응되는 이벤트의 시퀀스로 보고 실행 시간에 블랙박스 모델 외부로 발생한 이벤트를 제어하여 기존의 모델 행동을 수정한다. 더해서 이벤트 제어 모델 템플릿은 사용자가 프로그램 구현을 하지 않고 모델 행동을 수정할 수 있도록 하여 더 나은 확장성을 가지게 하며 수정에 따른 개발 비용 상승을 막는다. 본 논문은 제안하는 방법의 효용성을 보이기 위해 프로젝트 경험으로부터 가정한 사례 연구를 포함하고 있다. 이를 통해 기존의 시뮬레이터를 재사용함에 따른 이득을 확인할 수 있다.

주요어 : DEVS 형식론, 이벤트 기반 시뮬레이션, 이벤트 제어 모델, 모델링 및 시뮬레이션

* 이 논문은 2017년도 정부(미래창조과학부)의 재원으로 정보통신기술진흥센터의 지원을 받아 수행된 연구임 (No. 2017-0-00461, 사용자 수준 맞춤이 가능한 범용 이산사건 시뮬레이션 SW 개발 플랫폼)

Received: 8 February 2018, **Revised**: 13 March 2018, **Accepted**: 13 March 2018

† **Corresponding Author**: Se Jung Kwon
E-mail: sejungkwon@kaist.ac.kr
School of Electrical Engineering, KAIST

1. 서론

이산 사건 시스템은 실제 시스템을 설계하거나 분석하기 위해 이벤트 단위로 시스템을 추상화 한 것으로 이벤트란 시스템의 상태가 바뀌는 순간을 의미한다. 과거에는 이산 사건 시스템의 시뮬레이션을 위해서 이벤트 기반 실행 알고리즘을 통해 이벤트에 대응하는 함수들을 순차

적으로 실행하도록 했으나, 사용자의 편의성이나 재사용성등을 고려하여 객체 중심으로 정형화되고 구조적인 모델링 및 시뮬레이션 방법론이 발전하였다. 본 논문이 대상으로 하는 DEVS 형식론[1]은 그 대표적인 경우이다.

시스템을 추상화하여 수학적으로 모델을 기술하기까지는 상당한 사전 지식과 진입장벽이 존재하기 때문에 도메인 사용자들이 시스템에 대한 질문(What-if Question, WIQ)과 요구사항을 발주하면 Modeling and Simulation (M&S) 전문가들이 모델을 기술하고 시뮬레이터를 구현하여 배포하는 형태로 프로젝트가 이뤄지는 경우가 많다. 일반적으로 도메인 사용자들은 M&S 기술이나 프로그래밍에 대한 이해가 적기 때문에 파라미터 입력이나 직관적으로 모델을 합성하여 다양한 분석 실험을 하도록 요구한다. 다양한 프로젝트를 진행하면서 사용자들의 요구사항에 맞게 시뮬레이션 환경을 설계/구현하다보니 Fig. 1과 같은 DEVS 모델 라이브러리 기반의 시뮬레이션 환경을 배포하는 형태가 정립되었다.

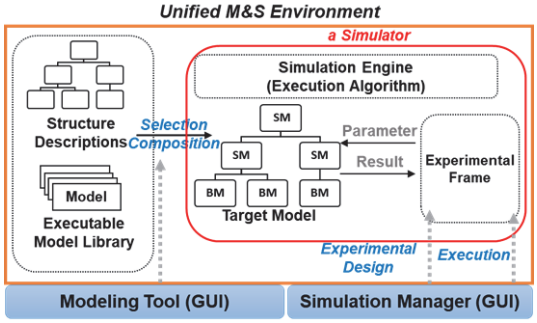

Fig. 1. Target M&S environment

DEVS 모델 라이브러리는 기초 행동을 기술하는 원자 모델의 컴파일된 라이브러리를 담고 있다. 사용자들이 Visual C++와 같은 코딩 툴이나 컴파일러를 통하지 않고 실행하기를 원하다보니 복잡한 과정을 생략한 채로 사용자의 다양한 요구사항들을 만족시킬 수 있는 라이브러리를 C++ DLL과 같은 형태로 제공한다. 사용자는 모델링 툴을 통해 요구사항에 맞춰서 모델을 선택/합성하고 시뮬레이션 매니저를 통해 파라미터를 입력하여 필요한 시뮬레이션을 다양하게 수행한다.

그러나 배포된 라이브러리 기반의 시뮬레이션 환경은 초기에 제기된 WIQ들에 제한된 확장성을 가질 수밖에 없다. 약간의 변화가 필요한 상황에서도 기존의 실험 틀(Experimental Frame, EF)이나 모델 라이브러리가 반영할 수 없는 내용이라면 개발자에게 새로운 프로젝트를 발주하여 수정 개발하게 할 수밖에 없다. 그러나 이미 프로젝트가 종료되었기 때문에 소스 코드가 존재하지 않거나 심지어 다른 개발자에 의해 해석되어 재개발될 필요가 생기기도 한다. 이것은 책정하지 않았던 상당한 예산 부담을 추가로 발생시키기 때문에 도메인 사용자들은 기존의 있는 모델들로 최대한 가까운 시뮬레이션을 진행하고 그 결과를 토대로 유추하는 선택을 할 가능성이 높다. 이것은 결국 시뮬레이션 결과의 정확도와 신뢰성을 크게 훼손하게 되고 M&S 공학 자체의 효용성에 대한 폄훼로 이어지곤 했다.

따라서 본 논문은 그 동안 프로젝트 경험을 토대로 모델 라이브러리 기반 시뮬레이션 환경의 확장성을 향상시키는 연구를 진행했다. 이를 통해 사용자가 변화하는 요구사항에 맞춰 그들이 배포 받은 시뮬레이션 환경에 더 높은 자유도를 가질 수 있도록 했다. 이를 위해서 본 논문은 이벤트 제어 모델 템플릿을 제안한다. 이벤트 제어 모델(Event Control Model, ECM)은 이벤트 기반 실행의 관점에서 모델 객체가 아닌 입출력에 집중하여 시뮬레이션 동작 시에 발생하는 이벤트를 수정/삭제/생성하는 추가적인 이산 사건 모델 기술 방법이다[2]. 본 논문은 한 발 더 나아가 프로그래밍을 피하고 싶은 도메인 사용자들을 위하여 ECM 템플릿을 제안한다. 모델링 툴에서 필요한 곳에 ECM을 삽입하고 그 내용을 UI를 통해 기술함으로써 약간의 지식만으로 사용자가 그들에게 필요한 수정을 할 수 있도록 지원한다.

본 논문의 구성은 다음과 같다. 2장에서는 기존의 환경이 가지는 확장성에 대해 말하고 3장에서 ECM에 대해서 소개한다. 4장에서는 이것을 템플릿으로 발전시킨 적용 프로세스를 정리한 다음 5장에서 사례 연구를 통해 효용성을 보인다. 마지막으로 6장에서 결론을 맺는다.

2. 관련 연구

확장성(Extensibility)은 요구사항에 있을법하나 예측하지 못한 변화에 대해서 최소한의 노력으로 적응할 수 있는 능력[3]이다. 앞서 말한 시뮬레이션 환경에서의 확장성은 주로 실험 틀과 동적 구조를 통해 이뤄졌다.

첫째로, 실험 틀은 시뮬레이션 모델을 실험/분석하기 위해서 파라미터를 입력하고 결과를 수집하며 시뮬레이션을 제어하는 역할을 한다. 확장성 측면에서 실험 틀을 통해 입력되는 파라미터의 범위, 제어할 수 있는 범위가 확장될 수 있는 범위에 해당한다. 가장 기초적인 실험 틀[1]로 시작해서 이를 통해 시뮬레이션을 제어하고자 하는 관

련된 다양한 확장[4-5]이 연구되어 왔다. 하지만 이런 연구들은 결국 시뮬레이션 모델 외부로 공개된 인터페이스의 정의에 한정되어 있으며 제어의 범위를 시뮬레이션 한 번의 실행으로 한정하고 있다는 한계가 있다.

둘째로, 동적 구조란 시뮬레이션 모델이 실행된 후에 어떤 조건에 의해서 모델 구조나 메시지 전달의 경로가 변화하는 것을 의미한다. 동적 형식론을 통해 많은 접근이 이뤄졌으며 상당수의 DEVS 시뮬레이션 엔진이 다양한 방법으로 이 기능을 제공하고 있다[6-8]. 하지만 이미 모델을 설계할 때 동적 구조에 대한 기술이 존재해야하며 결합 모델 수준에서의 제어라는 한계가 존재한다.

앞선 말한 환경에서 두 가지 확장성 모두 이미 M&S 전문가에 의해 구현된 내용에 제한을 받는 확장성이다. 즉, 주어진 WIQ들에 대해서 M&S 전문가가 어느 정도의 확장성을 가질 것인지를 결정하게 되며 변화된 요구사항이 그 범위를 넘어설 경우에는 이를 해결할 수 없다.

본 연구에서는 이런 한계를 넘어서기 위해서 제어 대상의 범위를 확장했다. 기존 연구들의 제어 대상을 확장하면서 환경이 배포된 이후에도 추가적으로 행동을 쉽게 추가 기술하거나 고칠 수 있는 방법을 제안한다.

3. 이벤트 제어 모델

이산 사건 시뮬레이션은 이벤트 기반 시뮬레이션으로 추상화 할 수 있는데 이벤트는 다음과 같이 정의된다.

Definition 1. In a set of events, E, an event is defined as E_k =< tg, src, t_N, v >, where

1. tg is a target of the event.
2. src is a source of the event.
3. t_N is an execution time of the event.
4. v is a set of variables.

이산 사건 시뮬레이션은 크게 메시지 전달과 스케줄링으로 이뤄진다. 이때 각각의 과정을 상태 변화를 일으키는 이벤트라는 관점으로 추상화하여 살펴보면, 모델은 메시지 전달을 위해 메시지를 담은 데이터 이벤트가 어디로 전달될지를 기술하여 시뮬레이션 엔진(이벤트 스케줄러)에 등록한다. 등록된 이벤트는 정해진 시간(일반적으로 동시간)에 도착지 모델로 전달된다. 따라서 데이터 이벤트의 tg에는 도착지 모델에 대한 정보가, v에는 메시지 정보가 저장되며 t_N값에는 일반적으로 현재 시간이 담기게 된다. 스케줄링을 위해서는 모델들이 각각 자신이 언제 실행될지를 알리는 정보를 담은 시간 이벤트를 엔진에 등록하고 엔진은 등록된 이벤트를 정렬하여 시간 순서대로 실행한다. 따라서 이산 이벤트의 tg는 자기 자신을 가리키며 tN에는 모델이 언제 실행될지에 대한 정보가 담기게 된다.

이벤트 관점에서 볼 때 시뮬레이션이란 이러한 이벤트 시퀀스를 만들어내는 과정이다. 이벤트 제어란 어떤 모델의 이벤트 집합으로부터 만들어지는 이벤트 시퀀스에 런타임에 접근하여 이벤트를 삭제/변조/생성 하는 것을 의미한다. 그 결과 제어된 이벤트가 이벤트 스케줄러에 삽입되고 기존의 모델을 고치지 않은 상태에서 의도하는 다른 결과가 발생하게 된다. 이를 위한 ECM[2]은 다음과 같이 정의된다.

Definition 2. The Event Control Model (ECM) is defined as $ECM =< \{E_k\}, S_{ecm}, \{f_i\}, CR, SELECT >$, where

1. $\{E_k\}$ is a set of target events in a simulator.
2. S_{ecm} is a set of states for the ECM.
3. $\{f_i\}$ is a set of event-oriented control functions.
   : $(E_k \cup \varnothing) \times S_{ecm} \times t \rightarrow (\{E_k\} \cup \varnothing) \times S_{ecm}$
   - t : A current time when the f_i is executed.
4. CR is a set of relations between E_k and f_i.
   : $CR \subset \{E_k\} \times \{f_i\}$.
5. $SELECT$ is a tie-breaking selection function.
   : $2^{\{fi\}} - \varnothing \rightarrow f_i$

ECM과 함께 이산 사건 시뮬레이션을 실행하면 런타임에 발생한 이벤트에 대하여 CR을 통해 해당되는 이벤트 제어 함수를 실행하고 변화된 이벤트를 삽입한다. S_{ecm}은 이벤트 함수 간의 상태 공유를 위해서 사용되며 SELECT 함수는 같은 이벤트에 여러 개의 함수가 연결되어 있을 때 우선순위를 정한다. ECM을 통해 변화시킬 수 있는 모델 행동의 범위는 ECM을 제안한 논문[3]에서 자세히 다루고 있다.

ECM을 적용함에 있어서 여러 가지 제약 사항이 존재한다. 첫째, ECM은 모델 외부로 발생한 이벤트를 제어 대상으로 보고 있기 때문에 모델 내부의 상태에는 접근할 수가 없다. 즉, 내부 상태를 입력으로 하는 제어 함수는 만들 수 없다. 둘째, 변조된 데이터는 반드시 도착지 모델에 의해 해석이 가능해야한다. 즉, 고친 이벤트의 값이 모델의 입력으로 허용한 범위를 넘어서거나 아예 새로운 이벤트를 만들어버리면 제대로 된 실행을 담보할

수 없다. 셋째, 출력과 내부 상태 사이의 관계가 없어야 한다. 모델은 외부로 내보낸 출력에 대한 기억을 하지 말아야 한다. 만약 외부로 보낸 데이터를 기억하고 있다가 그 데이터를 토대로 다른 작업을 수행한다면 외부에서 데이터가 ECM에 의해 수정된 것을 알 수 없기 때문에 정상 동작을 보장할 수 없다.

하지만 이런 제약 사항들은 반대로 ECM을 적용하기 위해 세 가지 가정 사항으로 볼 수 있다.

첫째로, ECM을 적용하기 위해서는 잘 나뉜 원자 모델들로 이루어져 있어야 한다. 더 많은 데이터가 모델 밖으로 공개된다는 뜻이기 때문에 ECM을 적용할 지점이 늘어난다. 둘째로, 잘 정의된 Input/Output (I/O) 명세가 필요하다. 어떤 모델이 어떤 인터페이스를 가지는지에 대해 잘 정의되어 있다면 외부에서 어떤 입력을 넣었을 때 모델이 정상 동작할 수 있는지 잘 알 수 있다. 셋째로, 모듈성이 중요하다. 모델은 외부와 단절된 상태로 인터페이스를 통해서만 동작해야한다.

이 세 가지 가정 사항들은 모두 정형화된 이산 사건 시뮬레이션을 함에 있어서 중요하게 다뤄진다. 특히, 모델 라이브러리를 만들 때 잘 나뉜 모듈성 있는 모델들을 세세한 I/O 명세와 함께 만들어야 이들을 조합하여 다양한 모델을 합성해낼 수 있기 때문에 필수적이다. 즉, 본 논문이 대상으로 하고 있는 모델 라이브러리 기반 시뮬레이션 환경은 ECM을 적용하기에 적합하다.

더해서, 시뮬레이션 엔진 측면에서도 중요한 가정 사항이 있는데, 시뮬레이션 엔진이 이벤트 제어를 위한 API를 제공해야한다. 이벤트를 모델과 엔진이 공유하는 정보라고도 볼 수 있는데 이를 제어할 수 있어야 한다. 본 연구에는 이벤트 기반 시뮬레이션으로 DEVS 모델을 실행하는 E-DEVSim++[9]를 사용하였다.

4. 대상 환경에 맞는 적용 프로세스

대상 시뮬레이션 환경에 ECM을 적용하는 사람은 도메인 사용자와 M&S 전문가로 나눌 수 있다. WIQ의 변화가 발생하였을 때 도메인 사용자는 기존의 환경이 이를 허용할 수 있는지를 검토하여 가능하다면 기존 연구에서 밝힌바와 같은 방법을 통해 확장한다. 이것이 불가능하다면 새로운 프로젝트를 발주하여 M&S 전문가에 의해서 진행해야 하는데, 많은 비용과 시간이 필요하다.

여기에 ECM을 적용하게 되면 프로젝트를 발주하기 전에 한 번 더 ECM을 활용할 수 있는지 탐색하게 된다. 도메인 사용자는 M&S에 대한 지식은 부족하지만 도메인 지식은 충분하며 이미 요구사항, 시나리오 등을 제공한 사람이기 때문에 모델 구조를 놓고 ECM을 어디에 적용할지 잘 이해할 수 있다. 다만, 복잡한 ECM은 만들 수 가 없는데, 원자 모델의 내부 상태를 유추해야하는 경우는 M&S 지식이 없는 사용자가 구현하기 쉽지 않기 때문이다. ECM을 만들기 복잡하거나 신뢰할 수 없는 상황이 되면 결국 M&S 전문가에 프로젝트를 의뢰할 수밖에 없는데 물론 이 상황에서도 ECM의 효용성은 충분히 존재한다.

M&S 전문가는 도메인 지식이 부족하기 때문에 도메인 사용자의 요구사항을 해석하여 새로운 프로젝트를 진행하기에는 시간이 많이 걸리지만 기존의 모델 명세를 충분히 이해할 수 있다. 따라서 시간이 조금 걸리더라도 ECM을 필요한 만큼 구현할 수 있으며 기존의 명세를 토대로 내부 상태를 해석할 수 있기 때문에 제한이 없다.

Fig. 2는 이를 정리하여 나타낸 도메인 사용자의 ECM 적용 프로세스이다. WIQ의 변화에 대응해서 사용자는 우선 기존의 환경이 해결할 수 있는지 확인하고 그것이 불가능할 때 필요한 모델의 행동이 ECM을 통해 기존의 모델을 확장함으로써 해결할 수 있는지 확인한다.

확장이 가능하다고 판단된다면 I/O 흐름을 보고 적절한 ECM을 삽입하여 행동을 바꿀 수 있는지 확인하고 ECM을 구현한다. 그리고 그 결과가 믿을만한 것인지 확인한다. 이때, I/O 명세로부터 자동으로 ECM이 올바르게 동작하는지를 확인하는 탐지 함수를 생성할 수 있다. 모델의 정상 동작을 보장하는 I/O의 범위를 알고 있기 때문에 추가 생성된 제어 함수들로 하여금 값을 벗어나는지 여부를 확인할 수 있다. 모델들이 잘 정의되어 있으며 I/O에 대한 명세가 잘 정의되어 있기 때문에 이를 잘 확인하는 것을 통해 모델의 정상 동작을 확인 할 수 있다.

모델을 확장하는 것으로 부족하고 아예 새로운 모델을 만들어야한다면 그 모델의 행동이 몇 개의 제어함수로 추상화가 가능한지 확인해본다. 가능하다면 앞서 말한 과정을 통해서 진행할 수 있다.

이것이 불가능하거나 제어함수를 만들기가 어렵거나, 신뢰할 수 없는 결과를 얻을 수 없다면 결국 새로운 프로젝트를 발주해야 할 텐데 M&S 전문가에게 있어서도 ECM의 효용성은 의미가 있다.

하지만 이런 프로세스에서 큰 맹점이 하나 있는데, 앞에서 언급하였듯이 ECM을 기술하기 위해서는 프로그래밍이 필요하다. 비록 모델을 고치는 것보다는 적은 비용으로 이룰 수 있겠으나 도메인 사용자가 프로그래밍 수준을 거치지 않고자 하는 경향과 맞지 않는다.

따라서 본 연구에는 템플릿을 통해서 도메인 사용자가

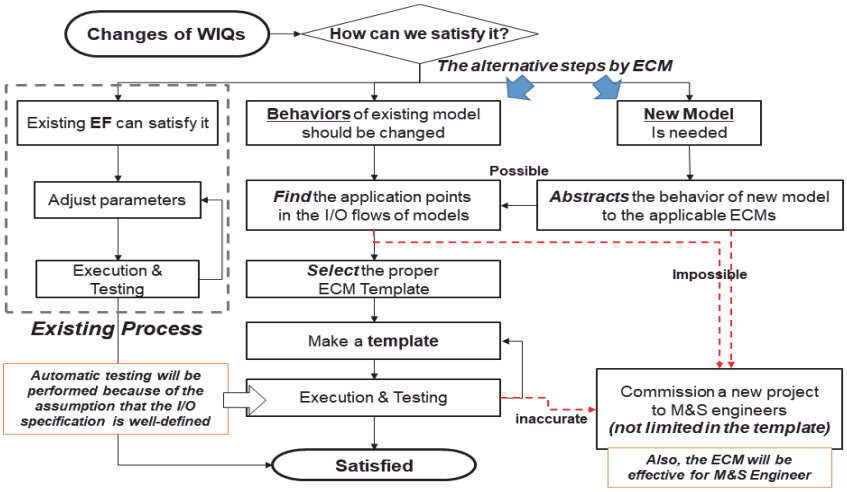

Fig. 2. Overall Process of the proposed work

ECM을 사용하는 것을 지원하고자 한다. 템플릿은 프로그래밍을 하지 않고 사용자에게 어느 정도 확장 가능성을 제공한다. 기존에 있던 모델링 UI에서 모델 구조에 ECM을 추가하면 활성화되는 조건을 입력하고 어떤 이벤트의 변수를 대상으로 할지 결정하여 그 변수를 고치는 간단한 식을 입력한다.

입력된 ECM은 모델링 UI가 저장한 모델 구조와 함께 저장되어서 시뮬레이션 엔진이 실행될 때 해석되어 삽입된다. 자세한 내용은 5장에서 계속해서 다룬다.

5. 사례 연구: 템플릿을 사용한 도메인 사용자의 워게임 모델 확장

사례연구는 수상 전투체계 시뮬레이션 모델[10][11]을 대상으로 했다. 해당 체계를 분석하기 위한 모델 라이브러리와 시뮬레이션 환경이 존재할 때 추가된 WIQ들을 가정하고 ECM이 어떤 역할을 할 수 있는지 보인다.

5.1 User Interface (UI)

모델링 UI는 제공되는 DLL (Dynamic-link Library) 모델들을 토대로 사용자가 결합 모델을 구성할 수 있도록 한다. XML으로 출력을 내보내면 시뮬레이션 엔진에 의해 해석되어 실행된다. 본 연구는 프로젝트를 위해 개발되었던 모델링 UI를 수정하여 ECM을 기술할 수 있도록 했다. 기술된 ECM은 XML 파일로 함께 출력된다. Fig. 3은 실제 모델링 UI의 그림이다. 기존에 있던 결합 모델을 기술하는 창과 모델의 속성을 기술하는 창에 ECM을 기술하는 창을 추가했다. ECM 창의 모습은 Fig. 4와 같다.

Fig. 5는 결합 모델에 ECM을 추가한 모습이다. 데이터 이벤트는 모델 구조에서 커플링 관계에 해당하기 때문에 모델과 모델을 이어주는 메시지 전달에 연결된 것을 확인할 수 있고 시간 이벤트는 모델에서 발생하기 때문에 모델에 시간 이벤트가 연결된 것을 확인 할 수 있다.

5.2 대상 워게임 모델 : 대어뢰전 시뮬레이션

사례연구를 위해 수상 전투 체계를 다루는 다양한 모델로 부터 대어뢰전의 대응 체계를 평가하는 시나리오를 위한 모델을 구성하였다고 가정한다. 대어뢰전은 기만기를 대응체계로 가지는 아군 수상함, 적 잠수함, 적 어뢰로 이루어져 있으며 시나리오는 Fig. 6과 같다. 잠수함은 아군 수상함을 발견하고 어뢰를 발사한다. 어뢰와 아군 수상함은 서로를 초음파 센서로 탐지하는데 수상함이 어뢰를 발견하면 기만기를 발사하고 회피 기동을 실시한다. 대응 체계를 통해 아군 수상함이 얼마나 생존할 수 있는지를 평가하는 것이 시나리오의 목적이다. 이를 시뮬레이션하기 위한 구조는 간단히 Fig. 7과 같이 나타낼 수 있다.

시뮬레이션이 시작되면 잠수함 모델이 어뢰를 발사하는 메시지를 통해 어뢰를 활성화 시킨다. 어뢰와 수상함은 위치 메시지 (POS)를 통해 서로의 위치를 공유한다. 수상함이 어뢰를 탐지하면 Fire 메시지를 통해 기만기들을 활성화 하고 기만기 또한 수상함에게 자신의 위치 정보를 보내서 어뢰가 기만당하는 행동을 모의하도록 한다.

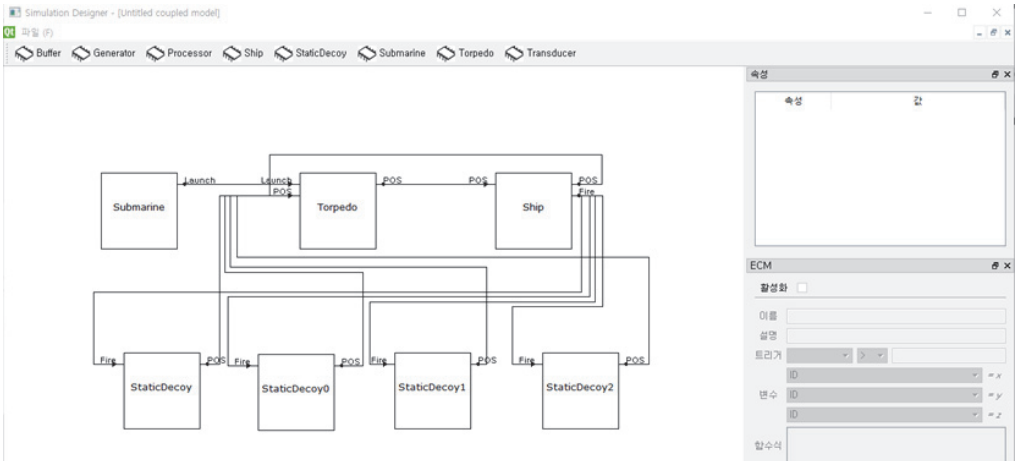

Fig. 3. Modeling UI

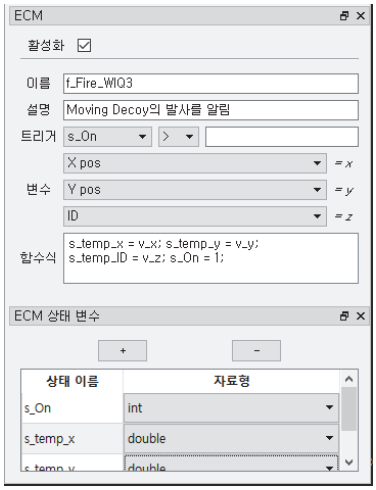

Fig. 4. ECM window

본 사례연구에서는 이 상태에서 기존의 확장성으로는 해결할 수 없는 세 가지 WIQ가 발생하였다고 가정하였다. 본래는 개발자에게 새로운 프로젝트를 발주해야하는 상황이지만 도메인 유저가 간단한 ECM을 기술함으로써 해결하는 것을 보여주려고 한다. 새로운 WIQ들은 해당 워게임 모델로 진행하였던 실제 프로젝트에 기반하여 제기 되었으며 다음과 같다.

- WIQ1 : 기만기를 발사할 때 발사 위치의 오차를 고려하면 어떻게 될 것인가?
- WIQ2 : 대응체계에 재머를 추가한다면 어떤 변화를 보일 것인가?
- WIQ3 : 기만기가 일정확률로 고장 난다면 어떤 결과를 보일 것인가?

가정된 WIQ들은 사실 모델이 크게 변화하는 문제는 아니지만 실제 개발 과정에서 제기되었던 변화들이다. 이로 인해서 모델 라이브러리의 모델을 수정해야하는 상황이 발생할 텐데, 이에 대해 ECM 템플릿을 적용하여 기존의 모델을 수정하지 않고 변화를 수용하였다.

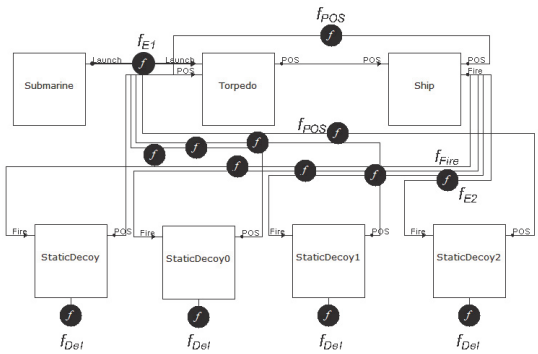

Fig. 5. Models with control functions

5.3 ECM 템플릿 기술

ECM 템플릿 창을 통해 기술하는 순서는 다음과 같다.

a. S_{ecm} 기술 : 공용 변수 창을 통해서 ECM 기술을 위해 필요한 상태를 기술한다. 이 변수의 값은 사용자 또는 함수 표현식을 통해 고쳐질 수 있다.
b. 이벤트 선택: 이벤트 제어 함수를 삽입할 이벤트를

선택한다. 데이터 이벤트를 위해서 연결 관계를 선택하거나 시간 이벤트를 위해서 모델을 선택한다.

c. 이벤트 트리거 선택: ECM이 항상 활성화되는 것은 아닐 때에 트리거를 위한 옵션을 선택한다. 이 옵션은 간략하게 구성되어 있는데, 더 복잡한 옵션이 필요할 때에는 함수 표현식을 통해 구현할 수 있다.

d. 변수 선택: 해당 메시지 또는 스케줄링 정보 중에 필요한 내용을 선택한다. 선택된 변수는 다음의 함수 표현식에서 v_x, v_y, v_z로 사용된다.

e. 함수 표현식 기술: 마지막으로 표현식을 기술한다. 표현식으로는 C++ 기반의 표현식을 입력하면 오픈소스 기반의 해석기가 내용을 해석한다. 그림 8과 같은 형태로 함수가 동작한다. 이런 과정을 통해서 기술된 ECM은 5.3.1-5.3.3에서 기술하였다.

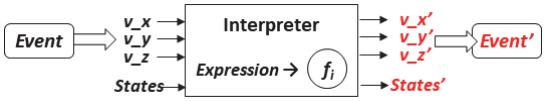

Fig. 6. Interpreter operation for ECM templates

5.3.1 WIQ1

- f_{error}
  - Target Variable : x and y in Fire Message
  - Expression
    v_x := genNormal(v_x, 5.0);
    v_y := genNormal(v_y, 5.0);

WIQ1를 만족시키기 위해서는 Ship에서 Decoy로 가는 발사 명령의 위치 정보를 변조시켜야한다. 따라서 해당 메시지의 x, y 좌표를 대상으로 정하고 일정 확률 변수를 적용하여 변화시키는 함수를 통해 고치도록 한다.

5.3.2 WIQ2

- f_{Del}
  - Target Variable : t_N in Decoy Model
  - Expression
    if(rand() < S_prob) v_x := Infinity;
- States
  - S_prob : 기만기가 실패할 확률

WIQ2를 만족시키기 위해서는 인위적으로 기만기 모델의 동작을 멈추면 된다. 스케줄링 관점에서 보면 스케줄링 정보를 삭제(=무한대로 수정) 하면 된다. 따라서 기만기에서 발생하는 시간 이벤트를 타겟으로 하여 그 값을 일정 확률로 무한대로 변화 시킨다.

5.3.3 WIQ3

- f_{Fire}
  - Target Variable : PosX and PosY in Fire Msg.
  - Expression
    S_On := 1; S_posX := v_x; S_posY := v_y;
    S_st := GetCurrentTime();
- f_{POS}
  - Trigger Option : S_On / = / 1
  - Target Variable : PosX and PosY in POS Msg.
  - Expression
    if(GetCurrentTime() - S_st < S_lt and CalDist(S_posX, S_posY, v_x, v_y) < S_range and rand() < S_prob) v_z := Infinity;
- States
  - S_On : Trigger State
  - S_posX : X position of initial ring
  - S_posY : Y position of initial ring
  - S_st : Starting time of a jammer
  - S_lt : Life time of a jammer
  - S_range : Effective range of a jammer
  - S_prob : Deletion probability

재머는 신호를 발생시켜 어뢰가 수상함을 찾아내는 것을 방해한다. 이를 추상화하여 POS 메시지를 삭제하는 재머 모델을 새로 개발해서 그 역할을 하도록 해야 하지만 본 연구에서는 제어 함수를 통해 그 역할을 하도록 했다. f_{Fire} 함수는 대응체계가 발사되는 시점을 잡아내는 역할을 하며 S_On 상태를 1로 바꾼다. f_{POS} 함수는 S_On이 1일 때 동작하며 여러 상태를 점검 하여 POS 메시지를 삭제해야할 때 v_z로 선택된 t_N값에 무한대 값을 넣는다.

이상의 ECM들로 인해 매번 새롭게 프로젝트를 발주해야하는 비용을 줄이고 사용자가 직접 모델을 확장할 수 있도록 했다. 해당 ECM들을 통해 실행된 결과는 기존의 프로젝트[10][11]를 통해 얻어진 과거 결과와 비교를 통해 정상 동작함을 확인하였다.

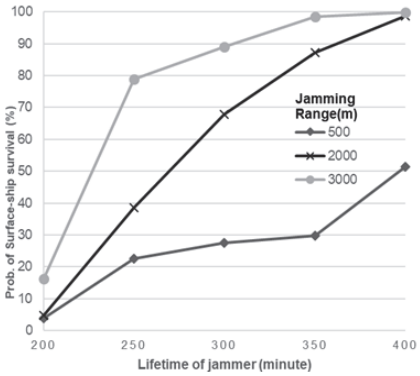

Fig. 9. Actual Simulation Results of WIQ3

그림 9는 그중 가장 복잡한 WIQ3에 대해서 실행한 결과이다. S_lt 값을 200초에서 400초까지 S_range 값을 500m에서 3000m까지 변화시켜가면서 100번씩 실험한 평균 생존율을 그래프로 나타내고 있다. 이 결과는 실제 Jammer 모델을 수정하여 시뮬레이션 했던 결과[11]와 비교하여 평균 1% 이내 오차를 보였으며 약 100번 수행하고 여러 가지 난수에 기초한 오차가 포함된다는 것을 고려하면 같은 실험 결과를 얻었다고 볼 수 있다. 실행 횟수를 늘려가면서 비교하면 오차가 점점 줄어드는 것을 확인할 수 있다. 나머지 WIQ에 대해서도 같은 결과를 얻었다.

6. 결론

본 논문은 모델 라이브러리 기반의 시뮬레이션 환경을 배포 받은 도메인 사용자들의 확장성을 개선하는 방법을 다루고 있다. 기존의 모델들에 수정 요구사항을 반영하기 위해 이벤트 제어 모델을 사용할 수 있으나 여전히 진입 장벽이 남아 있는 도메인 사용자들을 위한 GUI를 통한 템플릿을 제안하였다. 사용자들은 제안된 프로세스에 따라 ECM의 적용 위치를 선택하고 이에 대한 빈 칸을 채워 넣어 모델을 확장할 수 있다. 실제로, 개발 프로젝트를 진행하다보면 약간 변화된 WIQ임에도 불구하고 기존의 환경이 해결할 수 없어 큰 비용이 낭비되거나 문제 해결을 포기하는 상황이 자주 발생한다. 이러한 상황에서 본 연구로 인해 도메인 사용자들은 새로운 프로젝트를 발주하는 것으로 인해 발생했던 비용을 줄일 수 있는 대안을 선택할 수 있으며 더 나은 M&S 공학의 효용성을 얻을 수 있다.

References

[1] Bernard P Zeigler, *et al. Theory of modeling and simulation*. Academic press, 2000.

[2] Se Jung Kwon, *et al*. "Adaptive Discrete Event Simulation Systems to Embrace Changes of Requirements Using Event Control Models." *IEEE Transactions on Systems, Man, and Cybernetics: Systems,* 2017.

[3] Matthias Zenger. *Programming language abstractions for extensible software components*. PhD thesis, ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE, 2004.

[4] Saurabh Mittal and Bernard P Zeigler. Dynamic simulation control with queue visualization. *In Summer Computer Simulation Conference*, 2005.

[5] James Nutaro and Phil Hammonds. Combining the model/view/control design pattern with the devs formalism to achieve rigor and reusability in distributed simulation. *JDMS*, 1(1):19-28, 2004.

[6] Adelinde M Uhrmacher. A system theoretic approach to constructing test beds for multi-agent systems. *In Discrete event modeling and simulation technologies*, pp. 315-339. Springer, 2001.

[7] Jang Won Bae, *et al.* Simulation-based analyses of an evacuation from a metropolis during a bombardment. *Simulation,* 90(11):1244-1267, 2014.

[8] Saurabh Mittal. Extending dodaf to allow integrated devs-based modeling and simulation. *JDMS*, (2):95-123, 2006.

[9] 권세중, 김탁곤, "이벤트 지향 DEVS 실행 환경의 설계, 구현 및 성능 비교," 한국 시뮬레이션 학회 논문지, Vol. 20, No. 1, pp. 87-96, 2011년 3월.

[10] Kyung-Min Seo, *et al.* Measurement of effectiveness for an anti-torpedo combat system using a discrete event systems specification-based underwater warfare simulator. *JDMS*, 8(3):157-171, 2011.

[11] Se Jung Kwon, *et al.*, "Effectiveness Analysis of Anti-torpedo Warfare Simulation for Evaluating Mix Strategies of Decoys and Jammers," AsisSim '2011, Seoul, Korea, Nov., 2011.

권 세 중 (sejungkwon@kaist.ac.kr)

2009 한국과학기술원 전산학과 학사
2011 한국과학기술원 전기 및 전자공학과 석사
2018 한국과학기술원 전기 및 전자공학과 박사

관심분야 : 모델링&시뮬레이션, 시뮬레이션 엔진 구현, 하이브리드 시뮬레이션

이 준 희 (the78910@kaist.ac.kr)

2016 한동대학교 컴퓨터공학 학사
2016~ 현재 KAIST 전기 및 전자공학부 석사과정

관심분야 : DEVS 형식론, 모델링 & 시뮬레이션

최 창 범 (cbchoi@handong.edu)

2005 경희대학교 컴퓨터공학 학사
2007 KAIST 전산학 석사
2014 KAIST 전자공학 박사
2014~ 현재 한동대학교 ICT창업학부 조교수

관심분야 : DEVS 형식론, 소프트웨어 품질 보증, VV/A

김 탁 곤 (tkim@ee.kaist.ac.kr)

1975 부산대학교 전자공학과 학사
1980 경북대학교 전자공학과 석사
1988 Univ. of. Arizona, 전기 및 컴퓨터공학과 박사
1989～1991 Univ. of Kansas, 전기 및 컴퓨터공학과, 조교수
1991～현재 KAIST 전기 및 전자공학과, 교수
- 한국시뮬레이션 학회 회장, 국제시뮬레이션학회(SCS) 논문지(Simulation) EIC 역임
- SCS Fellow, 모델링 시뮬레이션 기술사(미국)
- *Who's Who in the World* (Marguis 16thEdition, 1999) 등재
- KIDA Fellow, 연합사, 국방부/합참, 국방과학연구소, 기품원 자문위원 역임

관심분야 : 모델링/시뮬레이션 이론, 방법론 및 환경개발, 시뮬레이터 연동