DEVSim-Java : Internet/WEB을 기반으로 한 DEVS 모델의 시뮬레이션 환경

조 정 훈° , 김 탁 곤
한국과학기술원 전기 및 전자공학과

Web-based Environment for Simulation of DEVS Models

Jeonghun Cho° , Tag Gon Kim
Dept. of EE, KAIST

기존의 이산 사건 시스템 시뮬레이션 환경들은 Interoperability의 문제, Portability 문제로 인하여 Internet과 Web상에서 분산 시뮬레이션이 불가능하다. 본 논문에서는 이러한 제약을 해결하고자 자바를 사용하여 DEVS(Discrete Event Systems Specification) 형식론을 구현하여 Internet/Web 상에서 시뮬레이션이 가능한 DEVSim-Java환경을 설계하고 구현하였다. DEVSim-Java를 사용하여 시뮬레이션 환경을 구현함으로써 원격지에서 개발된 시뮬레이션 모델들을 인터넷을 통하여 재사용 하는 remote model-base(RMB) 개념을 제안한다. DEVSim-Java는 자바의 장점을 이용하여 시뮬레이션 과정을 애니메이션으로 잘 나타낼 수 있고, 시뮬레이션의 결과를 Graphical Analyzer를 통해 분석할 수 있게 된다.

1. 개 요

이산 사건 모델링 시뮬레이션은 Manufacturing 시스템, 통신 네트웍, 그리고 병렬 컴퓨터와 같은 시스템의 설계 및 해석에 사용되어져 왔다. Zeigler에 의해 소개된 DEVS 형식론[Zeigler 1984]은 이산 사건 시스템을 모듈로 나누어서 이를 계층적으로 모델링 할 수 있는 수학적 기반을 제공한다. 이러한 DEVS 모델링과 시뮬레이션을 웹 상에서 할 수 있는 환경을 개발하는 것이 본 논문의 취지이다.

기존의 이산 사건 시스템의 시뮬레이션 환경은 language-based 또는 library-based로 구현되었다. 이러한 환경들은 Interoperability의 결여와 Portability의 결여로 Internet/Web상에서 분산 시뮬레이션이 불가능한 문제점이 있다.

이러한 문제점은 특별한 환경에서만 시뮬레이션 application 개발을 가능케 하여 재사용이 불가능하며 높은 모델 개발비용을 초래하게 된다.

자바는 Machine independent한 byte code를 생성하기 때문에 위의 제약들을 해결할 수 있다. 따라서 자바를 사용해 시뮬레이션 환경을 구성함으로써 웹을 통하여 어떤 컴퓨터에서든지 같은 모델을 사용해 시뮬레이션을 할 수 있게 하고, 자바의 그래픽을 이용해 시뮬레이션의 과정을 애니메이션으로 볼 수 있게 하여 직관적으로 이해할 수 있게 하는 목적이다.

2. DEVSim-java 시뮬레이션 환경의 설계 및 구현

2.1 DEVS 형식론

Zeigler에 의해 소개된 DEVS 형식론[Zeigler 1984]은 이산 사건 시스템을 모듈로 나누어서 이를 계층적으로 모델링 할 수 있는 수학적 기반을 제공한다.

DEVS 형식론으로 가장 작은 모델인 atomic 모델을 기술하고, 이러한 기본 모델들을 모듈화, 계층적인 방법으로 연결하여 복잡한 coupled 모델을 기술하게 된다.

Atomic 모델은 가장 기본적인 모듈로서 입력 사건 집합, 출력 사건 집합, 상태 집합, 모델의 다음 상태를 기술하는 전이함수들과 출력을 내기 위한 출력 함수 및 시간 함수로 구성되며, 수학적으로 표현하면 다음과 같다[Zeigler 1984].

Definition 1. Atomic 모델

$$M = < X, S, Y, \delta_{int}, \delta_{ext}, \lambda, ta >$$

X : input events set;

S : sequential states set;

Y : output events set;

$\delta_{int} : S \rightarrow S$: internal transition function;

$\delta_{ext} : Q \times X \rightarrow S$: external transition function;

$Q = \{(s,e) | s \in S, 0 \le e \le ta(s)\}$: total state of M;

$\lambda : S \rightarrow Y$: output function;

ta : S → Real : time advance function.

Coupled 모델은 여러 모델들의 계층적인 구조를 기술한다. 이것은 여러 개의 모델이 서로 어떻게 연결되어 있는지를 기술하며 다음과 같이 표현된다[Zeigler 1984].

Definition 2. Coupled 모델

$$DN = < X, Y, M, EIC, EOC, IC, SELECT >$$

X : input events set;

Y : output events set;

M : set of all component models in DEVS;

$EIC \subseteq X \times \bigcup_i X_i$: external input coupling relation;

$EOC \subseteq \bigcup_i Y_i \times Y$: external output coupling relation;

$IC \subseteq \bigcup_i Y_i \times \bigcup_i X_i$: internal coupling relation;

$SELECT : 2^M - \varnothing \rightarrow M$: tie-breaking selector.

여기에서 X_i와 Y_i는 각각 i^{th} 컴포넌트 모델들의 입력과 출력 사건집합이다.

2.2 Abstract 시뮬레이터

Zeigler가 제안한 abstract 시뮬레이터[Zeigler 1984] [SBPARK]는 DEVS 형식론에 의해 명세 된 모델을 해석하여 시뮬레이션 하는 알고리듬이다.

이를 위하여, abstract 시뮬레이터는 두 가지 유형으로 이루어져 있는데 하나는 atomic 모델을 위한 simulator이고 또 하나는 coupled 모델을 위한 coordinator이다. 이들은 각각 한 쌍이 되어 시뮬레이션이 이루어지게 되어있다. 모델 개발에서 composition 기술을 통해 모델을 재사용하기 위해 모델은 외부 agent에 의해 동작되어지는 수동적인 모델이 되어야 한다. 따라서 DEVS 모델들의 시뮬레이션 흐름은 모델과 연결되어 있는 abstract 시뮬레이터에 의해 제어된다.

Abstract Simulator는 (x, t), (*, t), (done, t_N), 그리고 (y, t)의 네 가지 이벤트 타입을 사용하여 시뮬레이션에 필요한 정보를 주고받는다. DEVSim++[Ahn 1994; Kim 1994]에서는 이러한 추상화된 시뮬레이터를 C++로 구현하였다.

2.3 DEVSim-java 시뮬레이션 환경

DEVS 형식론을 자바를 사용해 구현한 DEVSim-java 시뮬레이션 환경의 전체 구조는 그림 2.1과 같다. DEVSim-java 시뮬레이션 환경은 앞 절에서 설명한 abstract simulator, utility 클래스, 입출력 분석기, 그리고 모델러가 만든 모델들로 구성되어져 있다.

DEVSim-java 클래스 구조는 그림 2.2와 같다. 여기에서 AtomicModel과 CoupledModel로부터 상속받아서 새로운 atomic model과 coupled model을 만들게 된다. 앞 절에서 설명되었듯이 모델과 abstract 시뮬레이터가 분리되어 있는데, 그림 2.2에서

AtomicModel과 **CoupledModel**에 abstract 시뮬레이터가 포함되어 있다. 모델러는 이 abstract 시뮬레이터로부터 상속을 받아서 자식 클래스를 만들어서 적절한 함수를 override해 줌으로써 하나의 모델이 완성된다.

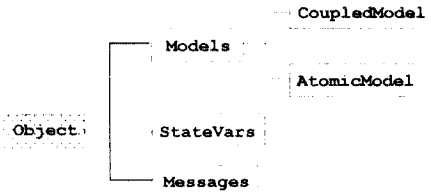

그림 2.2 : DEVSim-java Class Hierarchy

자바 applet을 사용하기 위해 Graphics 라이브러리가 추가되고, abstract 시뮬레이터 부분에서 화면에 출력을 보내는 대신에 applet으로 보내기 위해 override되었다.

3. 원격 모델 베이스 및 원격 시뮬레이션

3.1 원격 모델 베이스의 개념

DEVSim-java 환경은 실행 위치에 따라 서버에서 실행되는 것과 클라이언트에서 실행되는 것이 있다. 서버에서 실행되어지는 것이 원격 실행이 되고, 클라이언트에서 실행되는 것이 일반적인 자바의 동작이다. 또한, DEVSim-Java를 자신의 로컬에 가지고 시뮬레이션 할 수 도 있고, 원격지에 위치하고 자신의 로컬에는 시뮬레이션하기 위한 모델들만 가지고 시뮬레이션을 할 수 있다. 물론 모델들 중 몇 가지는 다른 위치에 놓고 시뮬레이션이 가능하다. 여기서 원격지에 있는 모델의 일부 또는 전체를 원격지 모델베이스(Remote Model Base)라고 정의하고 그것들을 이용해서 시뮬레이션을 수행하는 것을 원격 시뮬레이션(Remote Simulation)이라고 정의한다.

3.2 원격 실행

네트웍을 이용한 원격 실행에는 두 가지 방법이 있는데, 그 첫 번째는 CGI(Common Gateway Interface)의 도움으로 실행할 수 있는 방법과

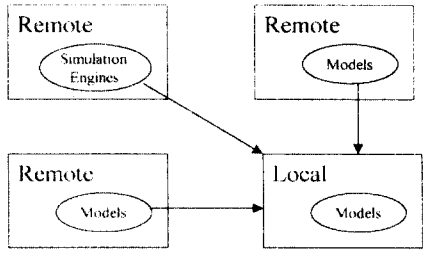

그림 3.1: 원격지에 분산되어진 DEVS 모델들의 시뮬레이션

RMI(Remote Method Invocation)를 이용하는 방법이다.

Internet/Web상에서의 원격 실행에서 CGI와 RMI의 차이는 다음과 같다. 즉, CGI는 실행한 결과 값을 HTML(Hyper Text Markup Language)파일로 보내주는 반면에, RMI는 호출된 method의 return 값을 보내주는 것이고, CGI는 클라이언트에서 호출이 되었을 때 프로그램을 실행하기 시작하는 반면에, RMI는 이미 실행이 되어서 sleep상태에 있다가 요청이 들어오면 깨어나게 된다. 즉, daemon이 상주하는 것이다.

3.3 원격 시뮬레이션

이 시뮬레이션(Remote Simulation) 방식은 여러 원격지에 모델들이 흩어져 있을 때 이것들을 로컬로 모아서 시뮬레이션을 하는 방법이다. 전체적인 구성은 그림3.1과 같다. 그림에서 보듯이 각각 다른 원격지에 이미 만들어진 모델들이 있을 때 그것들을 다운로드 받아서 로컬 머신에서 실행을 하게 된다. 따

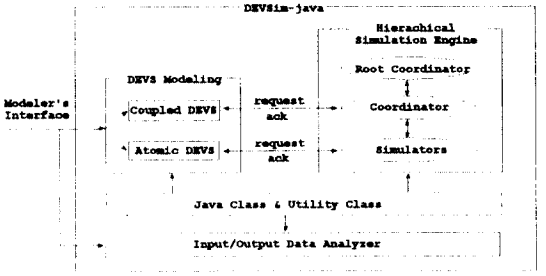

그림 2.1: DEVSim-java 시뮬레이션 환경

라서 다른 사람이 이미 만들어 놓은 모델이 있을 경우에 그것들을 재사용이 가능하게 되는 것이다.

4. DEVSim-Java의 검증 및 성능 측정

DEVSim-java 환경의 검증과 활용 예를 보이기 위해 예제로 Hyper Cube Multiprocessor를 시뮬레이션 한다. Hyper Cube Multiprocessor를 간단하게 모델링하고, 시뮬레이션을 하며 시뮬레이션 되는 과정을 애니메이션을 통해 보인다. 같은 예제를 DEVSim++을 사용한 예와 성능에서 비교를 한다.

4.1 Hyper Cube Multiprocessor 모델링

실시간 작업을 위해 여러 개의 프로세서가 동시에 작업을 할 때, 각각의 node 컴퓨터사이의 메시지 전달은 작업의 deadline에 영향을 주는 중요한 요소가 된다. 만약에 메시지의 도착이 늦어진다면 작업 완료 시간이 늦어질 수가 있고, 또한 deadline을 놓칠 수 가 있기 때문이다. 따라서, 각각의 node 컴퓨터간에 routing policy를 결정하는 것은 중요한 일이다. 최적의 routing policy를 결정하는 것이 Hyper Cube Multiprocessor 시뮬레이션의 목적이다[Ahn 1993].

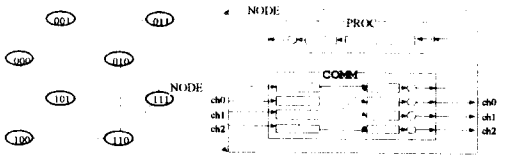

그림 4.1 3차원 Hyper Cube 네트웍과 노드모델

4.2 Hyper Cube Multiprocessor 시뮬레이션

Remote model-base 시뮬레이션은 client에는 시뮬레이션 환경만을 가지고 다른 서버에 시뮬레이션을 위한 모델들이 존재할 때 그 모델들을 download 받아서 시뮬레이션을 하게 된다. 실행하는 도중에 클래스가 필요하게 되면 network을 통해서 load하게 된다.

Graphical 시뮬레이션은 자바의 그래픽 기능을 이용해서 만들어진 applet이다. 초기화면은 그림 4.2와 같다. Start 버튼이 눌려지면 각각의 message의 전달과정을 animation으로 볼 수 있다.

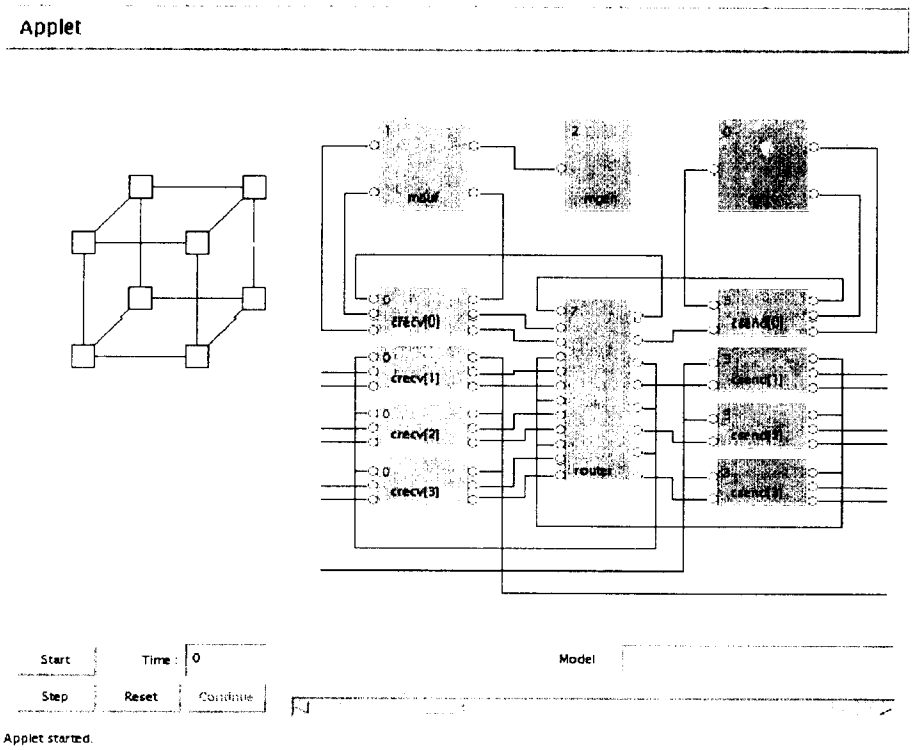

그림 4.2: Graphical 시뮬레이션 초기화면

4.3 동작 검증 및 속도 비교

DEVSim++는 1993년 Internet에 공개 소프트웨어로 등록한 이후 국내외 적으로 많은 응용에 적용되어 그 동작은 이미 검증되었다. DEVSim-java와 DEVSim++는 같은 DEVS 형식론을 자바와 C++을 통해 각각 구현했으므로 이들 결과는 동일하게 나와야만 한다. 실제로 4.1절에 소개한 모델로 두 환경에서 시뮬레이션 한 결과는 동일하였다.

그러나 이들 두 환경에서의 시뮬레이션 속도는 표 4.1에서 보듯이 DEVSim-Java가 DEVSim++보다 대략 15배정도 늦었다. 현재는 속도 면에서 큰 단점이 있지만 이것은 계속 보완해져 가고 있기 때문에 점차 개선되리라고 본다.

5. 결 론

본 연구에서는 자바를 이용해서 DEVS 형식론을 구현하여 웹에서 시뮬레이션이 가능하도록 하게 한다. 이를 위해서 그림 2.1에서 보는 바와 같이 abstract simulator, utility 클래스, 그리고 데이터 입출력 분석기 등 통합적인 시뮬레이션 환경을 제공한다.

DEVSim-java는 다음과 같은 특징을 가지고 있다. 첫째로, 원격 모델에 기반한 시뮬레이션이 가능하다. 둘째로 원격 시뮬레이션이 가능하다. 셋째로 그래픽컬 시뮬레이션이 가능하다. 활용분야는 통신망, 생산시스템, 교통시스템, 컴퓨터 시스템 시뮬레이션 등 기존의 모든 이산사건 시뮬레이션 응용 분야에서 활용되어질 수 있다.

No. of Packets	DEVSim++	DEVSim-java	Ratio
100	34.5	528.69	15.32
100	34.6	529.69	15.31
100	34.32	526.42	15.34
100	34.19	525.00	15.36
100	34.25	526.12	15.36
100	34.86	528.51	15.16
100	34.99	528.22	15.10
100	34.93	527.45	15.10
100	34.9	534.2	15.31
100	34.83	530.69	15.24
평 균	34.83	528.50	15.17

표 4.1: DEVSim++과 DEVSim-java의 속도 비교

앞으로 추가되어야 할 부분으로는 요즈음 들어 부각되고 있는 여러 언어간에 분산 시뮬레이션을 가능하게 해주는 HLA(High Level Architecture) 구조와 호환성을 가지도록 DEVSim-java를 확장함으로써 다른 언어로 만들어진 시뮬레이션 모델과의 분산 시뮬레이션이 Web환경에서 가능하도록 하는 것이다.

참고문헌

[Zeigler 1984] B. P. Zeigler, *Multifacetted Modeling and Discrete Event Simulation*, Academic Press, Orlando. (1984)

[Buss 1996] Arnold H. Buss and Kirk A. Stork, "Discrete Event Simulation on The World Wide Web Using JAVA", *Proceedings of the 1996 Winter Simulation Conference*, pp. 780-785. (1996)

[Fishwick 1996] Paul A. Fishwick, "Web-Based Simulation: Some Personal Observations", *Proceedings of the 1996 Winter Simulation Conference*, pp. 772-779. (1996)

[Kim 1994] T.G. Kim, *DEVSim++ User's Manual*, CORE Lab. Dept. of EE KAIST, pp. 8-9. (1994)

[Ahn 1993] Myoung S. Ahn and Tag G. Kim, "DEVS Methodology for Evaluating Time-constraint Message Routing Policies", Discrete Event Dynamic Systems, vol.3 pp. 173-192. (1993)

[Ahn 1994] 안명수, 박성봉, 김탁곤, "DEVSim++: 의미론에 기반한 이산사건 시스템의 객체지향 모델링 및 시뮬레이션 환경", 한국정보과학회 논문지, 제21권, 제9호, pp 1652-1664.(1994년 9월)