한국시뮬레이션학회 '00춘계학술대회 논문집 2000.4.15 영남대학교

실시간 시뮬레이션용 RT-DEVS Kernel 개발

조 성 면, 김 탁 곤
한국과학기술원 전자전산학과
시스템 모델링 시뮬레이션 연구실
E-mail : smcho@smslab.kaist.ac.kr, tkim@ee.kaist.ac.kr

Development of RT-DEVS Simulation Kernel

Seong Myun Cho, Tag Gon Kim
SMSLAB, Dept. of EECS, KAIST

DEVS 형식론[Zeigler 1984]은 이산 사건 시스템을 기술하기 위해 Zeigler에 의해 제안된 모델링 형식론이다. 본 논문에서는 DEVS 형식론을 실시간 시뮬레이션 모델을 기술하기 위한 RT-DEVS 형식론으로 확장하고 이를 통해 기술된 시뮬레이션 모델을 실행하기 위한 실시간 수행 알고리즘을 제안한다. 또한 기술된 모델을 실시간에 기반하여 수행시키기 위한 시뮬레이션 커널의 구조와 구현에 대하여 다룬다.

1. 서론

이산 사건 시뮬레이션은 단순한 성능 분석뿐만 아니라 복잡한 시스템의 동작을 분석하는 데에도 널리 사용된다. 실시간 시뮬레이션은 시뮬레이션 도중 외부 프로세스 -소프트웨어, 하드웨어 또는 사용자-와 상호 작용을 갖는 모델을 실시간에 기반하여 시뮬레이션하는 것이다. 이러한 실시간 시뮬레이션에서는 시뮬레이터가 모델이 행하는 입출력 사건(event)을 제시간에 맞추어 처리해 주어야한다. 기존의 성능 분석을 위해 개발된 시뮬레이션 방법은 입출력 사건 처리에 있어서 이러한 능력을 갖고 있지 않다. 왜냐하면 이러한 시뮬레이터들은 입출력 사건 처리에 있어서 실시간이 아닌 가상시간을 사용하기 때문이다.

본 논문에서는 모델과 외부 환경과의 상호 작용이 있는 이산 사건 시스템을 시뮬레이션할수 있는 방법론을 제시한다. 제안한 방법론 하에서 실시간 시스템은 RT-DEVS 형식론으로 기술되고 이와 상호작용을 갖는 외부 프로세스는 있는 그대로 사용할 수 있다. 실시간 시뮬레이션 환경은 RT-DEVS 모델을 위한 실시간 커널 상에서 구현되었다.

본 논문은 다음과 같이 구성되어있다. 2장에서는 실시간 시스템을 모델링하기 위한 RT-DEVS 형식론에 대해서 설명한다. 3장에서는 실시간 시뮬레이션 환경 전반에 관하여 설명하고 4장에서 결론을 맺는다.

2. RT-DEVS 형식론

RT-DEVS 형식론은 실시간 시뮬레이션을 위하여 기존의 DEVS 형식론을 확장한 형식론이다. RT-DEVS 형식론에서 atomic 모델은 다음과 같

이 정의된다.

$RT\text{-}AM = < X, S, Y, \delta_{ext}, \delta_{int}, \lambda, ta, \psi, A >$

, 여기서

X : 입력 사건의 집합

S : 상태 변수의 집합

Y : 출력 사건의 집합

$\delta_{ext} : Q \times X \rightarrow S$,

외부 상태 전이 함수,

여기서 $Q=\{(s,e) \mid s \in S \text{ and } 0 \le e \le ta(s)\}$

$\delta_{int} : S \rightarrow S$, 내부 상태 전이 함수

$\lambda : S \rightarrow Y$, 출력 함수

$ta : S \rightarrow I^{+}_{0,\infty} \times I^{+}_{0,\infty}$, 시간 진행 함수,

여기서 $I^{+}_{0,\infty}$ 는 음이 아닌 정수

$\psi : S \rightarrow A$, 작업 매핑 함수

A : 작업 집합 $A = \{a \mid t(a) \in I^{+}_{0,\infty}, \text{ and } t(a) \le ta \mid_{max}\} \cup \varnothing$

기존의 DEVS 형식론에서는 시뮬레이터가 atomic 모델의 시간 진행 함수를 부를 때마다 가상의 시뮬레이션 시간이 경과하였다. 그러나 RT-DEVS 형식론에서는 가상 시간 진행 함수를 실제의 시간으로 대체한다. 즉, 시간 진행 함수가 실제 시간의 경과를 나타낸다.

RT-DEVS 형식론에서 coupled 모델은 DEVS 형식론에서의 그것과 한가지 다른 점이 있다. 그것은 DEVS 형식론에서 동시에 발생하는 사건에 대한 순서를 결정하는 SELECT 함수가 없다는 것이다. 왜냐면 실시간 시뮬레이션에서는 그러한 경우가 일어나지 않기 때문이다. 하나의 프로세서 상에서 시뮬레이션을 수행하므로 설령 외부에서 두 개 이상의 사건이 전해지더라도 오직 한번에 하나의 사건만이 처리된다.

3. 실시간 시뮬레이션 환경 구현

3.1 시뮬레이션 방법론

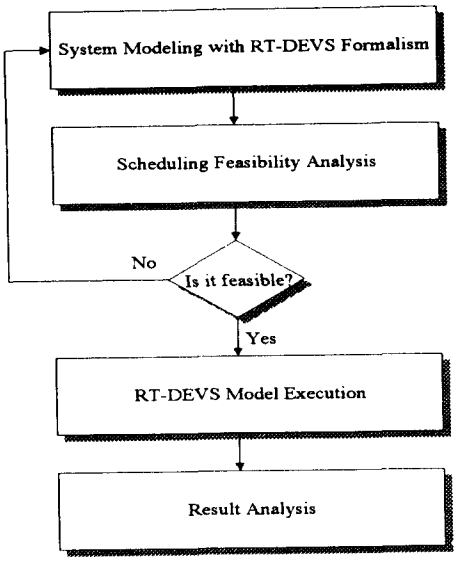

그림 1 실시간 시뮬레이션 절차

그림 1은 개괄적인 실시간 시뮬레이션 절차를 보인 것이다. 모델러는 RT-DEVS 형식론을 통해서 시스템을 구성하는 모델을 기술한다. 그런 다음 주어진 모델이 실시간 시뮬레이션이 가능한지를 알아보는 스케줄링 가능성 분석을 거친다. 이 단계에서 스케줄링 테스트 그래프가 만들어진다. 만약 주어진 시뮬레이션 모델이 실시간 시뮬레이션에 적합하지 않다고 판명되면 모델러는 실시간 스케줄링에 적합하도록 모델을 수정하는 작업을 하게 된다. 이 작업은 시간을 많이 소모하는 수행 함수를 더 작은 단위로 쪼개는 작업등을 말한다. 수정된 모델들이 실시간 스케줄링에 적합한 상태가 되면 시뮬레이션을 진행한다. 우선 스케줄러는 시뮬레이션 진행시 메시지 전달에 대한 오버헤드를 최소화하기 위하여 시스템에 포함된 모든 coupled 모델을 atomic 모델간의 연결만으로 재구성한다. 다음 각각의 모델에 해당하는 쓰레드를 생성하고 그들간의 메시지 채널을 확립한다. 실시간 제약조건을 만족된 상태에서 시뮬레이션이 마쳐지면 모델러는 얻어진 시뮬레이션 결과를 분석한다.

3.2 사건 구동 스케줄링

실시간 시뮬레이션 진행시 주어지는 시간 제약 조건을 만족하기 위해서 atomic 모델의 실행 우선 순위는 최근 시간 진행 함수의 결과에 따라 결정되어야 한다. 한가지 고려할 점은 RT-DEVS 모델의 t_N(다음번 스케줄링 시간)은 실행 도중에 계속 변한다는 점이다. 따라서 모델의 실행 우선 순위는 실행도중에 동적으로 변경되어야하며 상대적으로 작은 t_N을 갖는 모델은 더 높은 실행 우선 순위를 가져야 한다. 또한 시뮬레이션 모델은 외부로부터 입력을 받을 때마다 자신의 t_N을 변경하게되므로 스케줄러는 외부 사건이 전해지는 순간 해당 모델의 실행 우선 순위를 가장 높게 정한다. 왜냐하면 모델은 외부 사건이 전해지면 자신의 t_N을 변경하고 이에 맞게 자신의 실행 우선 순위를 재조정 받아야 되기 때문이다. 따라서 내부 사건 또는 외부 사건이 모델의 실행 우선 순위 변화시키게 되므로 이러한 스케줄링 방식을 "사건 구동 스케줄링"이라 한다. 표 1에 이 스케줄링 방식의 특징을 정리하였다.

	사건 구동 스케줄링
태스크 특성	비주기적인 사건 구동 태스크
스케줄링 목표	사건의 인과관계 유지
실행우선순위 결정요소	내부사건/외부사건
태스크간의 관계	입출력관계가 모델의 실행순서에 영향을 미침

표 1 사건 구동 스케줄링 방식

3.3 실시간 시뮬레이션 환경

그림 2는 RT-DEVS 시뮬레이션 환경을 나타낸다. RT-DEVS 마이크로 커널은 다수의 쓰레드를 동시에 수행하기 위한 기본적인 기능을 제공

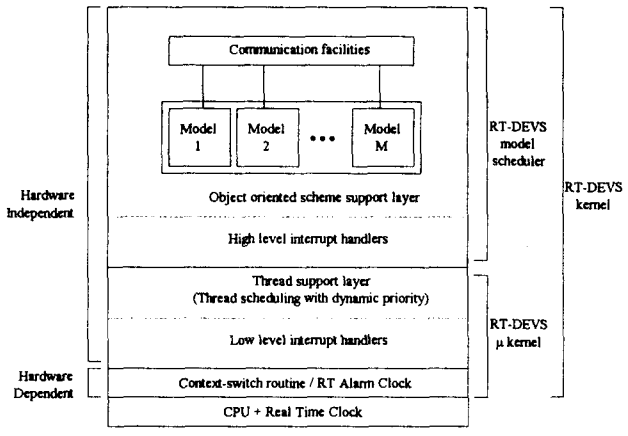

그림 2 실시간 시뮬레이션 환경

한다. 마이크로 커널의 하부에는 쓰레드 컨텍스트 스위칭 루틴과 리얼타임 클럭을 다루는 루틴이 존재한다. 리얼타임 클럭 관련 루틴은 현재 시간을 알려주는 루틴과 타이머 서비스를 제공하는 루틴 등이 있다. 쓰레드 지원층은 CPU가 다수의 모델 쓰레드간에 할당될 수 있도록 하며 또한 외부 환경으로부터 사건을 받아들이기 위한 인터럽트 처리 루틴을 가지고 있다. 구현된 실시간 시뮬레이션 환경의 주목적은 외부와의 상호작용이 있는 RT-DEVS 모델을 실행하려는 것이다. 그러기 위해서는 쓰레드 지원 층에서 제공하는 기능을 이용하여 각각의 시뮬레이션 모델을 실시간 태스크로 변환하여 실행시키는 스케줄러가 필요하다. 이 스케줄러는 제안된 "사건 구동 스케줄링"에 따라 RT-DEVS 모델의 실행 우선 순위를 조정한다. 마이크로 커널은 커널 쓰레드를 실행하는 엔진이며 스케줄러는 RT-DEVS 모델을 실행하는 엔진이라고 볼 수 있다. 그림 3에 스케줄러와 마이크로 커널 사이의 상호 작용을 나타내었다.

각 atomic 모델은 자신의 특성 함수를 실행하기 위한 모델 쓰레드를 만들게 되고 실시간 스케줄러는 이를 다음 장에서 설명될 CME(Concurrent Model Execution) 알고리즘을

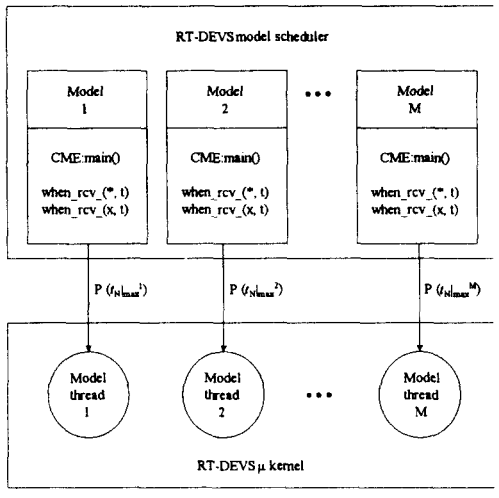

그림 3 모델 스케줄러와 마이크로 커널

통하여 실행하게 된다. 이 알고리즘은 각 모델이 현재 가지고 있는 $t_{N|max}$ 값에 따라 실행 우선 순위 함수를 통해 실행 순서를 조정하며 이는 하부층에 있는 마이크로 커널에 전달된다. 마이크로 커널은 CPU를 모델들간의 실행 순서에 맞게 할당하게 되므로 가장 높은 실행 우선 순위를 갖는 모델이 먼저 실행된다. 그림 3에 나타난 것처럼 각 모델은 자신의 특성 함수를 실행시켜주는 독립적인 쓰레드를 갖으며 모든 모델 쓰레드들이 동시에 수행된다. 이러한 실행 방식은 모든 사건을 순차적으로 정렬하여 처리하는 기존의 DEVS 시뮬레이터와는 다른 방식이다.

3.4 실시간 시뮬레이터 알고리즘

이 장에서는 각 atomic 모델 쓰레드의 동작에 대하여 설명한다. 모델 쓰레드의 일반적인 동작은 **CME:main()** 함수로 처리하고 이 함수는 내부사건을 담당하는 **CME:when_rcv_(*,t)**와 외부사건을 담당하는 **CME:when_rcv_(x,t)**을 부른다

```
1. CME:when_rcv_(*, t):
2.  if tN|min ≤ t ≤ tN|max then
3.      y := (s);
4.      send message(y, t) to associated ports;
5.      s := δ int(s);
6.      tL := t;
7.      tN := [tL + ta(s)|min, tL + ta(s)|max];
8.      P(tN|max)
9.      ψ(s)
10.  else
11.     error;
12. end if
```

```
1. CME:when_rcv_(x, t):
2.  if tL ≤ t ≤ tN|max then
3.      e := t - tL;
4.      s := δ ext(s,e,x);
5.      tL := t;
6.      tN := [tL + ta(s)|min, tL + ta(s)|max];
7.      P(tN|max)
8.      ψ(s)
9.  else
10.     error;
11.  end if
```

```
1. CME:main():
2.  s := s0;                    /* initialize */
3.  tN := [ta(s0)|min, ta(s0)|max];
4.  concur_forever for each RT-DEVS model
                                /* main loop */
5.      wait for an event;
6.      if an external event then
7.          when_rcv_(x, t);
8.      else if an internal time out event then
9.          when_rcv_(*, t);
10.     end if
11. end concur_forever
```

알고리즘 **CME:when_rcv_(*,t)** 는 DEVS atomic 모델의 그것과 다음의 차이점을 갖는다. 우선, 스케줄링된 시간이 모델 명세와 일치하는지 검사하고(2번째 줄) 다음 스케줄링 시간의 간격을 설정한다.(7번째 줄) 또한 모델의 다음 스케줄링 시간에 맞추어 실행 우선 순위를 조정하며

(8번째 줄) 모델의 현재 상태 변수에 의해 정해지는 작업을 실행한다.(9번째 줄) 알고리즘 **CME:when_rcv_(x,t)**이 갖는 차이점도 위와 비슷하다. 실행 우선 순위 함수 $P(t_{M\,max})$가 불려질 때마다 커널 쓰레드 엔진은 가장 높은 실행 우선 순위를 갖는 모델 쓰레드에 CPU를 할당한다. 모델이 출력을 발생시킬 때에도 모델 스위칭이 일어난다. 왜냐하면 입력을 받는 모델은 가장 높은 우선 순위를 갖게 되기 때문이다. **CME:main()** 함수는 CME의 메인 함수이다. 각 atomic 모델은 스케줄링된 내부 사건이 발생할 때까지 외부 입력 사건을 기다린다. 내부 사건이 발생하기 전에 외부 사건이 발생하면 **when_rcv_(x,t)** 함수를 수행한다. 만약 외부 사건없이 내부 사건이 발생하면 **when_rcv_(*,t)**를 수행하고 새로운 내부 사건 시간을 설정한다. 시뮬레이션 모델과 외부 환경의 실행 시간을 동기화하기위하여 모델 스케줄러는 atomic 모델이 해당 작업을 마치면 다음 스케줄링 시간에 이를 때까지 실행을 중지시킨다. 그러나 이 기간동안에 외부 사건이 발생하면 즉시 실행을 재개하여 해당 사건을 처리할 수 있게 한다. 이는 단순한 라운드 로빈 방식을 택했던 기존의 연구[Hong 1997]와 다른 점이다.

4. 결론

실시간 시뮬레이션은 시뮬레이션을 통한 운전자 교육 또는 실시간 플랜트 모델과의 연결을 통한 컨트롤러의 동작을 분석하는 작업등에 사용할 수 있다. 실시간 시뮬레이션을 수행하기 위해서는 단순 입출력뿐 아니라 시간에 따라 변화하는 모델을 기술할 수 있는 형식론과 주어진 모델을 실시간에 기반하여 실행할 수 있는 시뮬레이션 환경이 필요하다. 시뮬레이션 수행시 적용되는 실시간 스케줄링 알고리즘은 모델을 기술하는데 사용된 형식론에 따라 달라진다. 본 논문에서는 기존의 DEVS 형식론을 실시간 시뮬레이션에 적합하도록 확장한 RT-DEVS 형식론에 기반하여 실시간 시뮬레이션을 진행할 수 있는 사건 구동 스케줄링 알고리즘을 제안하고 이에 맞는 실행 환경을 구현하였다.

참고 문헌

[Cho 1998] Seong M. Cho and Tag G. Kim, Real-time DEVS Simulation: Concurrent Time-selective Execution of Combined RT-DEVS and Interactive Environment, *SCSC-98*, pp 410-415, Reno, Nevada, U.S.A., 1998.

[Coen 1997] Alberto Coen-Porisini, Carlo Ghezzi and Richard A. Kemmerer, Specification of real-time systems using ASTRAL, *IEEE Transactions on Software Engineering*, Vol. 23, No. 9, 1997.

[Hong 1997] Jun S. Hong, Hae S. Song, Tag G. Kim and Kyu H. Park, A Real-time Discrete Event System Specification Formalism for Seamless Real-time Software Development, *Discrete Event Dynamic Systems*, vol. 7, pp. 355-375, 1997.

[Kim 1991] Tag G. Kim, Hierarchical Class Development in The DEVS-Scheme Simulation Environment, *Expert Systems with Applications*, vol. 3, no.3, pp. 343-351, 1991.

[Motus 1994] Leo Motus and Michael G. Rodd, *Timing analysis of real-time software*, Redwood Books, Oxford, 1994.

[Krishna 1994] C.M. Krishna and Kang G. Shin, *REAL-TIME SYSTEMS.* McGraw-Hill Book, Oxford, 1994.

[Ramamrithan 1994] Ramamrithan, K., and

Stankovic, J. Scheduling Algorithms and Operating Systems Support for Real-Time Systems, *Proceedings of the IEEE*, January 1994.

[Burns 1997] Alan Burns and Andy Wellings, *Real-Time Systems and Programming Languages*, Addison-Wesley, 1994.

[Zeigler 1984] Zeigler, B.P., *Multifacetted Modelling and Discrete-Event Simulation*, Academic Press, New York, 1984.

[Zeigler 1993] Bernard P. Zeigler and Jinwoo Kim, Extending the DEVS-scheme knowledge-based simulation environment for real-time event-based control, *IEEE Transactions on Robotics and Automation*, vol. 9, no. 3, 1993.