추상 시간 기계의 시간 명세

노경주°, 박지연, 이문근
전북대학교, 컴퓨터과학과
{kjuno, jypark, mklee}@cs.chonbuk.ac.kr

Timing Specification in Abstract Timed Machine

Kyoung-ju Noh°, Ji-yeon Park, Moon-kun Lee
Dept. of Computer Science, Chonbuk National University

요 약

본 논문은 ATM(*Abstract Timed Machine*)에서의 시간 명세 방법에 대해서 기술한다. ATM은 임무 위급 시스템과 같은 실시간 시스템을 명세, 분석 및 검증 할 수 있는 LTS(*Labeled Transition System*) 정형 기법이다. 실시간 시스템이 요구하는 많은 속성 중 특히 시간 제약에 대해서는 사용자나 개발자 모두에게 명확하고 간결한 명세기법이 요구된다. ATM에서는 전이와 모드의 시간 제약을 간략하게 표현할 수 있는 방법을 제공한다. 또한 주기적 동작을 패턴인식하여 보여주는 주기 주제 모드를 통하여 주기와 관련된 동작과 시간 제약을 쉽게 파악할 수 있게 하며 주기적 동작을 선택적으로 간략화하는 방법을 제공한다. 본 논문은 ATM의 다양한 시간에 대한 요구사항에 대한 표현방법을 기술하고 예를 통해서 살펴본다.

1. 서 론

최근의 많은 시스템은 신속한 정보와 정확한 제어의 조작을 위해 즉각적으로 반응하는 실시간 시스템으로 개발되고 있다. 더욱 정확한 실시간 시스템에 대한 수요가 있으며, 빠른 기술의 발달과 요구사항의 변화를 반영할 수 있도록 기존의 시스템에 대한 재·역공학을 필요로 한다. 일반 시스템과 달리 실시간 시스템은 규모가 방대할 뿐만 아니라 병렬성, 동기/비동기, 통신, IO, 시간과 같은 복잡한 속성을 가지고 있다. 따라서, 순공학과 재공학의 통합 환경인 순환 공학 환경에서 실시간 시스템이 가진 속성을 명확하게 표현하고, 충분한 동적 정보를 제공할 수 있는 새로운 정형 기법에 대한 필요성이 대두되었다.

ATM은 실시간 시스템의 여러 속성을 위하여 개체들 사이에서 제어의 변환, 프로시저나 함수 개체의 다중 호출을 나타내는 실행 개체의 중복된 이미지 제어, 다수의 다른 이벤트 유형에 대한 제어, 즉각적으로 수행되지 않는 실행 시간을 가진 동기/비동기적 이벤트 처리 등의 동적 행위 등에 대한 표현력을 제공함으로써 실시간 시스템을 명세 할 수 있도록 한다.

실시간적 속성 중 시간은 실시간 시스템의 핵심이 되는 부분이다. 시간에 대한 여러 요구사항을 정확하고 간결하게 명세할 수 있어야 하여 시간에 따른 시스템의 동작을 명세할 수 있어야 한다. 또 시간에 대한 정의로부터 특정 시간에서의 시스템의 동작을 예측 할 수 있어야 한다.

본 논문은 새로운 정형 명세 언어인 ATM을 기술하고 ATM이 가진 여러 실시간적 특성 중 시간성에 관해 자세한 정의와 표현 예를 기술한다. 1장에서는 서론, 2장에서는 관련연구, 3장에서는 ATM에 대한 간략한 기술을 하며, 4장에서는 ATM의 시간 표현을 예제로 살펴본다. 마지막으로, 5장에서는 결론 및 향후 연구를 살펴본다.

2. 관련연구

시스템의 명세를 위해 Statecahrts[3], Modechart[9], CSM (Communicating State Machines)[5], CRSM (Communicating Real time State Machines)[1], Petri Net[10], Timed Petri Net[7] 등과 같은 다양한 정형 기법들이 개발되었다. Modechart와 CRSM, Timed Petri Net과 같은 경우는 실시간 시스템이 가진 실시간성에 대한 명세방법을 제공한다. 시간에 관한 명세가 가능한 기존의 정형 기법들은 시간 제약을 전이상에 레이블로 표현한다. 전이의 발생이 가능한 범위를 나타내는 최소, 최대 경계 시간과 전이가 지속되어야 하는 시간 등의 명세방법을 제공한다. 하지만 위와 같은 시간 명세 방법은 시스템의 전이에 관한 것으로 국한되어 있다. 따라서 특정 사건의 시간 제약 뿐만 아니라 시스템 내 프로세들의 생성, 소멸, 또는 대기 등의 행위를 표현하거나 다른 프로세스와의 상호작용에 있어서의 시간에 대한 제약 조건에 대한 표현을 할 수 있어야 하며 시간 제약을 가진 시스템의 스케줄링 등에 대한 명세 표현 방법이 필요하다

3. ATM

ATM은 계층성, 분산성, 실시간성, 우선권, 다수에 의한 동기화, 예외처리, 실행성 등 다양한 속성을 가진 실시간 시스템을 역공학 과정에서 명세하기 위해 개발된 LTS(*Labeled Transition System*)이다.

ATM은 모드(mode)의 집합, 가드(guard)된 전이(transition)의 집합, 포트(port), 실행 시작점과 실행의 종료점으로 구성된다. 머신은 ATM의 기본 단위로 내부에 모드를 포함한다. 일반적으로 머신은 태스크, 프로시저와 같은 독립적 프로그램 블록 단위를 표현 한다. 모드는 시스템의 상태를 나타내기 위한 것으로 모드의 역할에 따라 각 모드는 타입화 되어 이동 아이다, 타입, 시간 제약 등 자신을 대표할 수 있는 속성을 갖는다. 모드의 종류로는 계산 모드, 추상화 모드, 주제 모드가 있다. 계산 모드는 실행문을 내부에 직접적으로 포함하고 있어 모드 자체의 실행에 의해 다음전이를 유발한다. 추상화 모드는 소프트웨어 구조 계층의 하위 단계의 머신을 현재단계에서 추상화 시킨다. 주제 모드는 소프트웨어의 특정 기능을 수행하거나 특정 기능에 영향을 받는 모드들을 표현하는 모드이다.

전이는 이벤트, 조건, 시간제약으로 구성된 레이블을 갖는다. 포트는 머신 간의 통신을 위한 것으로 머신의 활성/비활성을 위한 활성화(activation) 포트, 엔트리(entry) 포트, 대체(substitute) 포트가 있다.

ATM은 1) 해석 (Interpretation), 2) 재구성 (Restructuring), 3) 명세언어로의 표현의 단계를 거친다. 해석 단계에서는 소

본 연구는 한국과학재단 특정기초연구(과제번호 1999-2-003-3) 지원으로 수행되었음.

코드로의 블록 단위에 따른 머신과 모드에 대한 타입 ATM이 생성된다. 재구성 단계에서는 시스템의 동적 행위에 따라 타입 ATM의 재구성이 이루어지고, ATM으로의 명세 단계에서 ATM으로 표현된다.

4. ATM에서의 시간 명세

ATM은 시스템이 가진 실시간적 제약을 표현하기 위해 두 종류의 타이머를 가진다. 타이머의 종류로는 시스템 전체에 포함되는 전역 타이머, 각 ATM에 속하는 지역(local) 타이머가 있다. 전역 타이머는 시스템이 활성화 될 때 함께 활성화 되고, 지역 타이머는 자신이 속한 머신이 활성화 될 때 활성화 된다.

타이머는 활성화 포트, 타임 포트, 알람(alarm) 포트를 가진다. 타임 포트는 타이머의 현재시간을 전송하는 포트이다. 알람 포트는 alarm[1], alarm[2], ..., alarm[n] 처럼 다수의 포트가 존재하고, 타이머가 속해 있는 소속 머신으로부터 특정 시간마다 시그널을 보내주기를 원하는 시간을 받아들여 해당 시간마다 소속 머신으로 시그널을 보낸다. 예를 들어 Timer.alarm[1]!(40)의 경우는 타이머의 1번 알람 포트를 통하여 알람 시간을 40초로 설정하고, Timer.alarm[1]?()는 타이머의 1번 알람 포트로부터 해당 시간이 경과 했다는 시그널을 받는 것을 나타낸다.

4.1 전이에서의 명세

전이의 시간 제약은 네 가지로 나눌 수 있다: 1) 전이에 시간 제약이 없는 경우, 2) 전이의 발생 가능한 기간에 대한 제약, 3) 전이 실행이 지속되어야 하는 제약, 4) 최소 시간(lower-bound)과 최대 시간(upper-bound) 사이에 일정한 전이가 지속(duration) 되어야 한다.

시간 제약이 없는 경우 전이는 시간과 관련된 레이블을 갖지 않는다. 최소 시간과 최대 시간 경계로 표현되는 경우 전이는 *[lb, ub]*와 같은 형식의 레이블을 갖으며 이는 전이가 *lb*와 *ub* 시간 사이에 발생해야 함을 의미한다. 전이의 일정 시간 지속에 관한 경우는 *[duration]*과 같은 레이블을 갖으며 이는 특정 전이가 *duration* 시간 동안 지속되어야 한다는 것을 의미한다. 최소 시간과 최대 시간 경계 사이에 일정한 전이의 지속은 *[lb, duration, ub]* 형식의 레이블을 갖는다.

<그림 1>은 전이의 시간 제약에 대한 예를 보여준다.

전이	설명
M.Act!()	머신 M에 활성화 시그널을 보내며 시간 제약은 없다.
M.Act!() [0, 10]	머신 M에 활성화 시그널 전이 가능한 순간부터 10 단위 시간 이내에 보내야 한다.
wait[10]	10 단위 시간 동안 wait 동작을 지속한다.
Pot.boil!()[0.4,10]	전이 가능하게 된 10 단위 시간 사이에 순간부터 Pot 머신에 boil이라는 시그널을 4 단위 시간 동안 보냄

<그림 1> 전이 상에서의 시간 제약 표시

4.2 모드에서의 시간 제약 표시

ATM의 모드 계산 모드, 추상화 모드, 주제 모드가 있다. ATM의 모드 또한 전이와 같이 시간 제약에 대한 표현을 제공하며 표현 방법은 전이에서의 시간 제약 표현의 법칙을 그대로 포함한다. 모드 중 주기적 동작을 나타내는 주제 모드는 주기 시간 제약의 표현방법을 가진다.

4.2.1 주기적 작업에 대한 시간 제약 표시

실시간 시스템은 일정 시간마다 작업을 반복적으로 수행하는 주기적 동작을 가진다. ATM은 이러한 주기성에 관련하여 간결한 표현을 제공한다.

주기적 동작에 대한 실행은 프로시저나 함수 내에 주기적 동작이 포함되는 경우, 주기적 동작을 포함하는 프로시저나 함수를 주기적으로 호출하는 경우 등 다양한 실행 사례가 있다. 주기적 동작에 대한 표현은 동일 하지만 후자와 같이 주기적 동작이 독립적인 ATM 머신에 속해 있는 경우는 주기적 동작에서 요구되는 시간 속성을 파라미터로 전달하는 호출 전이 이벤트가 선행된다.

주기적 동작은 먼저 일반적인 전이와 모드로 타이머를 사용하여 구체적으로 표현된다. 전이와 모드, 타이머를 사용하여 명세된 주기적 동작은 일련의 패턴을 지니고, 패턴 인식과정을 통하여 주기적 동작에 해당하는 범위를 파악할 수 있다. 파악된 범위는 주기적 동작에서 요구되는 다양한 시간 속성을 가진 단일 주기 주제 모드로 표현된다. 사용자가 주제 모드 안에 포함되는 것에 대한 간단한 형식을 원한다면 주기 주제 모드에 포함되는 영역이 주기적 시간 제약을 가진 추상화 모드로 대체될 수 있다. 주기적 동작이 요구하는 시간에 대한 속성은 실행이 가능하게 되는 대기시간(ready time), 주기를 나타내는 주기시간(period time), 주기적 동작이 실행을 지속해야 하는 실행시간(execution time), 실행의 데드라인(deadline)으로 구성된다. 추가적으로, 전체 주기 작업이 실행 할 수 있는 시작 시간과 종료 시간을 가진다.

<그림 2>는 주기적 작업의 시간 제약에 관한 레이블 형식을 보여준다.

```
[lower-bound, (ready-time, period,
   execution-time, deadline), upper-bund]
```

<그림 2> 주기적 작업의 시간 표기

독립된 주기적 작업은 호출되어 실행될 때 주기적 시간 제약에 대한 시간 값을 전달 받거나, 이미 저장되어 있는 값에 의해 수행된다.

4.2.2 주기적 작업에 대한 예

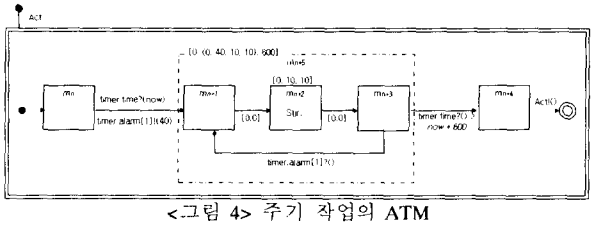
주기적 작업에 기술한 사항에 대해 예를 들어 살펴본다. <그림 3>은 주기적 작업을 표현한 프로시저이다.

```
.....
from now to now + 10 min every 40 sec
execute 10 sec within 10 sec do
begin
    stir;
end;
.....
```

<그림 3> 주기적 작업 예

<그림 3>은 해당 부분이 수행 가능하게 된 시간부터 10분 후까지 매 40초마다 10초씩 10초의 데드라인을 가지고 "stir"라는 동작을 수행하도록 하는 코드이다. 이를 ATM 형식으로 표현하면 <그림 4>과 같다. <그림 4>에서 점선으로 표현된 사각형은 주기 주제 모드를 나타낸다. 이 주기 주제 모드는 <그림 2> 형식의 시간에 대한 레이블을 가지고 있다. m_{n+2} 모드는 "stir"의 동작을 대기시간 0과 데드라인 10초 사이에 10 초 동안 수행한다. 40초의 주기는 타이머의 alarm[1] 포트를 통해 타이머에 설정되고 같은 포트를 통해 시그널을 수신함으로써 하나의 주기 종료를 알 수 있다. 전체 주기적 작업은 타이머를 통해 현재 시간과 주기적 작업의 종료 조건, 즉, 최대 경계 시

간을 비교함으로써 작업의 종료 여부를 결정한다.



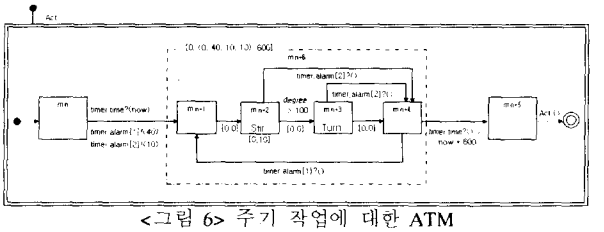
<그림 4> 주기 작업의 ATM

<그림 4>의 경우는 실제 주기적 동작을 포함한 모드가 하나뿐이어서 간략화된 모드의 시간 제약 표기법을 통해 주기적 동작의 한 주기안에 수행되는 동작을 쉽게 표기할 수 있었다. 반면에, <그림 5>의 코드를 표현한 ATM인 <그림 6>에는 m_{n+2}, m_{n+3}의 실행 시간의 합이 10초간 이루어 져야 한다. ATM은 이러한 경우에 타이머에 다른 알람 포트를 사용하여 전체 주기와 동작의 실행 시간의 종료를 파악할 수 있도록 하였다.

```
.....
from now to now+10 min every 40 sec
execute 10 sec within 10 sec do
begin
   stir;
   if degree > 100 then
      turn;

end;
.....
```

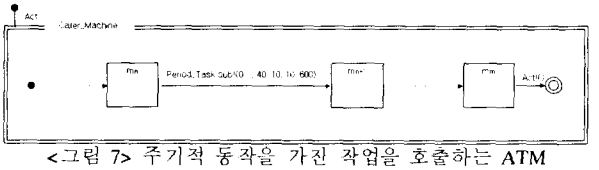
<그림 5> 주기적 작업에 대한 코드



<그림 6> 주기 작업에 대한 ATM

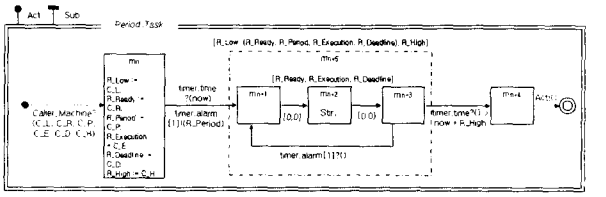
<그림 6>에서 "stir"와 "turn"은 10초 동안 수행되어야 한다. 동작의 수행에 대한 시간제약은 timer.alarm[2] 포트를 통해 이루어 진다. 즉 10초의 시간이 흐르면 타이머로부터 알람 시그널을 수신하여 제어가 주기의 끝을 나타내는 모드 m_{n+4}로 이동한다. 또한 타이머에 설정된 40초의 주기마다 timer.alarm[1] 포트를 통하여 시그널을 수신함으로써 반복의 초기 모드로 제어가 진입하여 주기적 작업이 지속된다.

<그림 6>에서 점선으로 표현된 모드는 사용자의 선택에 따라 간략화 된 모드로 대체될 수 있으며 이 모드는 주제 모드가 패턴 인식을 통하여 파악한 주기의 시간 제약을 레이블로 갖는다. 한 프로시저나 함수 내에서 표현되는 주기적 작업은 <그림 5, 6>과 같이 표현되지만 호출 관계에 의해 독립적으로 수행되는 작업은 함수 호출 형식과 주기적 시간 제약을 파라미터로써 표현된다. <그림 7>은 이에 대한 표현 방법을 보여준다.



<그림 7> 주기적 동작을 가진 작업을 호출하는 ATM

<그림 7>은 주기적 동작을 지닌 ATM을 호출하는 ATM으로서 호출에 대한 시그널에 주기적 동작의 시간 제약을 파라미터로 보낸다.



<그림 8> 독립된 주기적 동작을 가진 ATM

<그림 8>의 ATM은 호출자의 파라미터 값에 따라서 주기의 시간 제약이 결정 되어 이에 따라 주기적 작업이 명세된다. <그림 7>에서와 같이 40초의 주기로 10초의 데드라인을 가지고 10초의 실행을 나타내는 파라미터를 받았다면 주기적 작업을 포함하는 Period_Task 머신은 파라미터로 전달된 값에 의해 시간 속성이 명세되고 동작 하게 된다.

5. 결론 및 향후 연구

본 논문에서는 순환공학 과정에서 실시간 시스템을 명세하기 위해 개발된 ATM의 기본 개념과 ATM의 속성 중 시간을 명세하는 방법에 대하여 기술하였다. 특히, 실시간 시스템이 가진 주기적 성질을 명확히 정의하고 표현하는 방법을 보였다.

ATM은 원시코드 상의 시간 제약뿐 아니라 시스템의 실행 과정에서 포함되는 여러 프로시저나 함수 또는 물리적 장치에 대하여 각각이 가진 시간 제약과 전체 시스템으로서의 시간 제약을 관리하는 스케쥴링 전략을 필요로 한다. 또한 ATM의 추후 연구 주제인 우선순위와 스케쥴링의 정책, 시간과 전이에 대한 확률의 정책들에 대하여 연구를 지속한다. ATM이 실시간 시스템에 대한 이해 정보와 명확하고 간략한 명세를 제공할 수 있도록 앞으로 지속적인 연구가 이루어질 것이다.

참고문헌

[1] A. Shaw, "*Communicating Real-Time State Machines,*" IEEE Transactions on Software Engineering, Vol. 18, No. 9, pp. 805816, September 1992.

[2] C.A.R.Hoare, "*Communicating Sequential Processes,*" Prentice-Hall MD.

[3] D. Harel, "Statecharts: A Visual Formalism for Complex System," Science of *Computer Programming*. Vol. 8., pp. 231-274, 1987.

[4] Feldman and Koffman, "Ada95," Addison-Wesley, 1996.

[5] I. Kang and I. Lee, "State Minimization for Concurrent System Analysis Based on State Space Exploration," *Proceedings of Conference on Computer Assurance,* Gaithersburg MD, June 1994.

[6] Insup Lee, "Language constructs for Distributed Real-Time Programming," Dec, 1985.

[7] Joachim fischer, Evegeni Dimitrov, Udo Taubert, "*Analysis and formal Verification of SDL'92 Specifications using Extended Petri Nets*".

[8] Moon Lee, "*An Environment for Understanding of Real-time Systems,*" Ph.D. Thesis The University of Pennsylvania, 1995.

[9] S. jahanian and A. Mok, "*Modechart : A Specification Language for Real Time Systems,*" IBM Technical Report: RC 1514(November, 1989.

[10] Stuart Bennett, "*Real-Time Computer Control - An introduction*" Prenticce Hall, 1994.