STSR의 실시간 내장형 시스템의 명세+

*김진현○ *이수영 **손한성 *최진영
*고려대학교 컴퓨터학과, **원자력 연구소
*{ jhkim ○, ,sylee, choi}@formal.ac.kr, **hsson@kaeri.re.kr

Specification of Real-time Embedded System using STSR

* Jin-hyun Kim ○ * Su-Young Lee **Han-Sung Son *Jin-Young Choi
*Dept. of Computer Science Korea University
**Korea Atomic Energy Research Institute

요 약

원자력 발전소 안전계통이나 의료 시스템과 같은 실시간 내장형 시스템의 설계는 그 안전성을 분석하기 위한 정형 명세가 요구된다. 이러한 실시간 내장형 시스템이 명세를 위해 본 논문에서는 Statecharts를 확장하여 시간적 명세 및 분석에 용이하고 하드웨어/소프트웨어 통합 설계에 유리한 언어를 제시한다. 그리고 그 언어의 의미론을 기술하고, SyncCharts라는 정형명세 언어로 그 행위의 의미론을 부여한다. 이렇게 하여 기존의 Statecharts를 실시간 내장형 시스템에 적합하도록 수정하고 그 의미론을 부여한다.

1. 서론

원자력 제어 계통이나 항공 시스템 혹은 의료 시스템과 같은 시스템은 일반적으로 시간적 정확성이 요구 되는 실시간 시스템이다. 근래 들어 이러한 실시간 시스템은 내장형 시스템으로 개발되고 있다. 실시간 시스템의 정확성은 결과의 정확성 뿐 아니라 시간적 제약 조건에 달려 있다. 따라서 이러한 시스템의 설계는 시간적 명세가 반드시 포함되어야 한다. 최근에 이러한 고안전성 실시간 시스템을 명세하기 위해 다양한 정형 명세 언어가 개발되고 있다. 특히 설계자 및 구현자 혹은 그 외 시스템의 설계와 관련된 사람들 간의 이해 및 의사소통을 원활하게 하기 위해 도식적 언어가 개발되고 있다. 이러한 도식적 언어 가운데 Statecharts는 가장 잘 알려진 반응형 시스템 명세 언어로서 State-diagram에 계층성과 평행성을 포함하여 반응형 시스템을 보다 쉽게 설계하도록 고안된 언어이다. 하지만 Statecharts는 실시간 시스템을 명세할 수 있는 명확한 문법을 지니고 있지 않다. 즉 시간적 흐름 및 자원의 사용 등에 관한 명확한 문법 및 의미론을 지니고 있지 않다. 따라서 시스템의 시간적 분석에 상당한 어려움을 지니고 있다. 특히 근래에 들어 하드웨어 및 소프트웨어의 통합 설계에 관심이 더해지고 있는 시점에서 시간적 분석에 어려움을 지닌 Statecharts는 적합하지 않다.

본 논문에서는 이러한 문제를 해결하기 위해 Statecharts with Timed Shared Resource(이하 STSR)를 제안하고 이를 정의하고 실시간 시스템의 명세 및 내장형 소프트웨어 설계의 능력을 보인다.

본 논문의 구성은 다음과 같다. 1장에서는 STSR의 언어에 대해 기술하고 2장에서는 이를 통한 실시간 명세 기법을 기술한다. 3장 STSR의 언어 및 의미론을 제시하고 4장에서는 본 연구의 결론 및 향후 연구 과제를 기술한다.

2. STSR 언어

I-Logix 사의 STATEMATE MAGNUM은 Statecharts를 설계언어로 사용하는 반응형 시스템 개발 도구이다. 이 도구의 Statecharts는 실시간 시스템의 시간적 특성을 명세하기 위한 의미론으로 Synchronous time scheme 와 Asynchronous time scheme을 갖는다. 첫번째 Synchronous time scheme는 하나의 time-unit에 하나의 전이 스텝을 수행하는 시스템을 가리킨다. 두번째 Asynchronous time scheme은 time-unit 내의 하나 이상의 스텝을 수행하는 시스템을 가정한다. 본 논문의 STSR은 하드웨어/소프트웨어 통합 설계를 위해 이 두 가지 의미론 중 하나의 Asynchronous time scheme을 확장한다. 본 논문에서는 STATEMATE MAGNUM의 Asynchronous time scheme을 지닌 Statecharts을 다음과 같이 확장한다.

STSR = Statecharts with Asynchronous time scheme + Shared Resource + Priority + Time

STSR은 asynchronous time scheme에 자원 및 우선순위를 포함시킴으로 다음과 같은 특징을 지닌다.

첫째, 시간을 사용한 상태와 시간을 사용하지 않는 상태를 명확히 구분함으로 실시간 시스템의 설계에 대한 시간적 분석을 용이하게 한다. 둘째, 자원 사용에 대한 우선순위를 기술함으로 공유 자원에 대한 경쟁관계를 명확히 명세하며 스케줄러와 공유자원 분배에 관한 기술을 단순화하여 시스템 설계를 단순화 한다. 셋째, 하드웨어/소프트웨어 통합 설계를 위해 시간을 사용하는 부분을 명확히 함으로 통합 설계 이후, 하드웨어와 소프트웨어 부분을 나누기가 보다 분명하고 수월해진다.

+ 본 논문은 원자력 연구개발 중장기사업인 원전계측제어 사업단" 에 의해 지원되었음.

STSR 언어는 다음과 같이 정의된다.

Def 2.1 (State) : *S = {BS,AS,OS,TRS}*

State는 각각 label을 가지며, 다음과 같이 정의한다.

Base state (*bs∈BS*) : instantaneous 한 상태이다.

And-state (*as∈AS*) : Orthogonal state, 두 개 이상의 동시에 수행되는 상태를 의미한다.

Or-state (*os∈OS*) : 두 개 이상의 orthogonal하지 않은 상태들을 의미한다.

Timed-resource state (trs∈TRS): 우선순위를 갖는 자원을 갖거나 혹은 가지지 않고 시간만을 소모하는 상태를 의미한다.

Def 2.2 (전이 : *Transition*)

Transition, *T*는 다음과 같이 정의 한다.

$$T: s_1 \xrightarrow{\delta} s_2 \quad (s_1, s_2 \in S,\ \sigma \in \Sigma)$$

Def 2.3 (*Timed Transition*)

전이 가운데, 시간적 제약이 가해지는 전이를 *Timed Transition*, 라 하고, *timed transition* 의 집합을 *TT*라 한다.

Def 2.4 (*Atomic Time Constraint*)

실수 값을 갖는 *atomic time constraint*, *tc*는 전이 t ($t \in T$) 에 대한 시간적인 제약으로 정의한다. *atc*는 다음과 같이 표시한다.

$atc ::= \sim n$, for $\sim \in \{\geq, \leq, =\}$, $n \in R$.

*TC(TT)*는 모든 *TT*에 대한 time constraint의 집합을 가리킨다.

Def 2.5. (*Clock variable*)

Clock variable, *CV* 는, *atomic time constraint*, *tc*를 만족하는 시간이 할당되는 clock 변수의 집합이라 한다.

Def 2.6 (*Time interpretation*)

Time interpretation, *v*는 *tc*를 만족하는 시간 t ($t \in \boldsymbol{R}$)을 *cv*(*cv* ∈*CV*) 를 할당하는 함수라 한다.

$V : cv \rightarrow t \qquad (cv \in CV,\ t \in \boldsymbol{R})$

Def 2.7 (*Assignments*) : *A(I)*

I를 정수 값을 갖는 변수의 집합이라 할 때, I에 대한 정수 *assignment*는 $\langle v, c_1, c_2 \rangle$ 의 tuple로 $v = c_1 \cdot v + c_2$, $v \in I$이고 $c_1, c_2 \in Z$이다. *A(I)*는 *I*에 대한 모든 assignment의 power-set 를 가리킨다.

Def 2.8 (*Action*)

Action 은 이벤트를 내보내는 것과 *A(I)*를 포함한다.

Def 2.9 (*Act*)

전이 시, 받는 이벤트 및 수행되는 *Action*의 집합을 *Act*이라 하자. *ACT* 다음과 같이 표현된다.

e[c]/a, for e ∈ E(Event), a ∈ A(Action), c ∈ condition.

Def 2.10 (*Resource Label*)

*R*을 *resource*의 집합이라고 하고, *P*를 정수 값의 우선 순위라 할 때, *Resource label*, *rl*은 다음과 같은 형태를 갖는다.

$rl ::= \{(r_0, p_0), \cdots (r_n, p_n)\}$, $r \in R,\ P \in Z$

Def 2.11 (*Statechart with Timed Shared Resource*)

STSR = {*S*, *Σ*, *R*}

S : A set of State.

Σ : *S* x *TC(TT)* x *Act* x *S* : A set of transition labels.

R : A set of resources

이를 통한 실시간 시스템의 명세는 [그림 2-1]과 같을 수 있다. 이 그림은 Timeout interrupter를 명세한 STSR이다. 이것은 PR 프로세스가 r 이라는 자원은 적어도 2 time-unit 동안 사용하고 a라는 이벤트를 받아 프로세스를 종료하는 것을 의미한다. 만약 2 time-unit이 지나고 5 time-unit이 지났음에도 a가 오지 않으면 timeout을 통해 PR 프로세스를 종료 시킨다. 이 명세에서의 특징은 시간을 사용한 부분이 명확히 드러난다는 것이 특징 될 것이다. 다음 절에서는 STSR의 의미론을 제시하고 실시간 모델체커 벤치마킹 프로토콜인 Fischer의 프로토콜을 STSR로 명세한다.

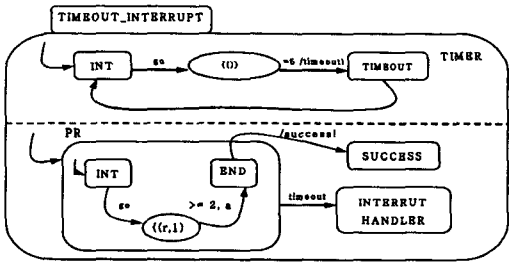


그림 2-1 STSR로 명세한 TIME INTERRUPTER

3. STSR의 의미론 정의

STSR은 그 행위에 대한 의미론으로 그 의미를 부여한다. STSR은 Synchrony Hypothesis 따라 이벤트에 대한 반응은 instantaneous하게 일어난다고 가정한다.

STSR은 STATEMATE 의 Statecharts 의미론 정의와 유사하게 가능한 run의 집합으로 정의한다. Run이란 외부환경으로부터 주어지는 일련의 외부에 대한 시스템의 반응을 의미한다. Run은 시스템이 가진 상태의 일련의 snapshot으로 이루어지는데 이를 status라 한다. 이러한 status 가운데 첫 status를 initial status라 한다. 이러한 status 간의 전이를 step이라 한다.

[그림 3,1] 과 같이 STSR의 run들로 행위를 정의한다.

STSR은 두 가지 형태의 Step을 갖는다. $step_t$는 시간을 소모하는 step으로서 이 step에서는 적어도 1 time-unit의 시간을 소모한다. 또 다른 step의 형태로서 $step_{ut}$가 있다. $step_{ut}$는 시간을 소모하지 않고 진행하는 step이다.

행위에 대한 의미론을 부여하기 전에 기본적인 원칙(principle)은 다음과 같다.

- 시간의 소모는 Timed Resource State에서만 소모 하고 Timed Resource State외에 시간을 소모하는 state는 없다.
- 하나의 이벤트는 하나의 step에서만 유효하며, 다음 step에서는 기억되지 않는다.

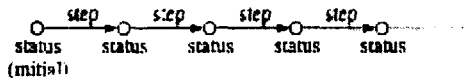


그림 3.1

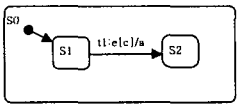
- 특정의 step에서의 내부 및 외부 이벤트에 대한 반응과 변화들은 그 step이 완료되고 난 후, 감지된다.
- 하나의 step의 처리는 그 step이 시작하기 전의 상태를 기반으로 처리된다.

3.1 기본적은 시스템의 행위.

Configuration은 시스템이 동시에 있을 수 있는 state의 집합이다. Configuration은 다음과 같은 rule을 지니다.

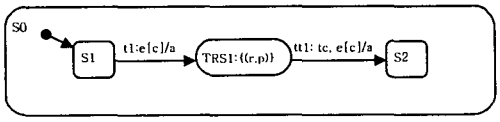
- C는 Root State를 포함하고 있다.
- C가 특정 OR-State A를 포함하고 있다면, C는 A의 하위 states 중 하나만을 가져야 한다.
- C가 특정 AND-state A를 포함하고 있다면, A의 하위 states를 모두 포함하고 있다.

특정 step에서는 네 가지 타입의 연산을 수행한다. 즉 전이, 특정 상태로 전이 될 때의 action 및 특정 상태에서 나갈 때 수행되는 action이다.



기본적인 전이 행위는 다음과 같이 정의된다.

- 시스템의 control 이 s1에 있고, trigger 이벤트 e가 들어오고 조건 c가 만족한다면, 전의 t1이 활성화 된다.
- 이때, 특정한 event exited(S1), entered(S2)이벤트가 발생한다.
- 특정 조건 in(S1)와 in(S2)가 각각 false와 true로 변한다.
- state S1에서 나올 때 수행되는 action이 수행되고, S2로 들어갈 때 수행하는 action이 수행된다.
- S0에서의 static reaction의 trigger가 true라면 이 SR이 수행된다.



기본적인 전이 외에도 basic state에서 timed resource state로의 전이 행위는 다음과 같이 이루어진다.

- S1에서 TRS1으로의 전이는 앞의 기본 룰을 따른다.
- TRS1에서는 다음과 같은 룰을 따른다.
  - 자원 r을 우선순위 p를 가지고 사용한다.
  - 만약 우선순위가 높은 프로세스가 r을 선전했을 경우, r을 점유하지 않고 시간을 소모하며 기다린다.
  - 자원 r에 대한 스케줄링은 매 clock마다 진행하여, 가장 높은 우선 순위를 가진 프로세스가 자원을 점유한다.
- 시간에 대한 제약조건 tc를 만족하고, trigger 이벤트 e가 들어오고 조건 c가 만족할 경우, 일반적인 전이 규칙을 따라 S2로 진행한다.

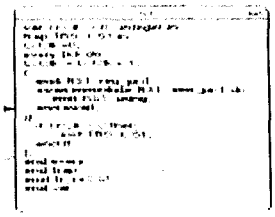


그림 3.2 SyncCharts로 정의된 Timed Resource State

위와 같은 룰에 따라 정형명세 언어인 SyncCharts로 임계구역에서 경쟁관계에 있는 자원과 시간을 사용하는 행위를 정의한 결과가 [그림3.2]와 같다. 이와 같이 STSR을 SyncCharts로 그 행위의 의미론을 정의했다.

이러한 정의를 바탕으로 Fischer' s protocol을 명세한 결과가 [그림3.3]과 같다.

4. 결론 및 향후 연구 과제

본 연구에서 제시된 STSR은 실시간 내장형 시스템의 하드웨어 및 소프트웨어 통합 설계를 위해 I-Logix 사의 Statecharts의 asynchronous time scheme을 확장하고 그 의미론을 제시한다. 그에 더하여 정형명세 언어인 SyncCharts를 통해서도 그 행위적 의미론을 제시한다.

STSR를 이용하여 명료한 시간 및 자원 사용을 명세하고 있으며 또한 우선순위의 명세를 통해 자원 경쟁관계를 명세함으로 시스템의 시간 사용을 분명히 파악할 수 있으며, 하드웨어와 소프트웨어의 통합 설계 및 실시간 운영체제와 같은 시스템의 명세에도 유용하게 사용될 것으로 사려 된다.

향후 연구 과제로는 이에 대한 도구 개발 및 실시간 모델체킹과 같은 정형 검증 도구를 통한 분석기법이 개발되어야 할 것으로 사려된다.

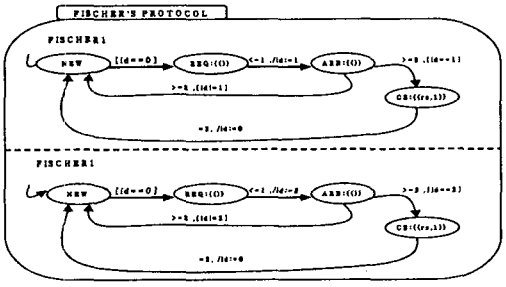


그림 3.3 STSR로 정의된 Fischer's Protocol

5. 참고문헌

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

[2] D.Harel " The STATEMATE Semantics of Statecharts." ACM Transaction of Software Engineering and Methodology, Vol 5, No. 4, pp. 293-333, Oct 1996,

[3] Charles Andre, "SyncChart: A Visual Representation of Reactive Behavior", Technical report RR-95-52, 13S, 1995