

# 경성 실시간 시스템 설계에 적합한 Priority 모델링 기법

김광덕, 이종순, 박성한  
한양대학교 전자계산학과

## the Priority Modeling Method for Designing the Hard Real-Time System

Kwang-Deok Kim, Jong-Soon Lee, Sung-Han Park  
Dept. of Computer Science & Engineering, Hanyang University

### 요 약

본 논문에서는 경성 실시간 시스템에서 발생하는 이벤트들을 효과적으로 나타낼 수 있는 개선된 Statecharts 모델링 기법과 Statecharts에 priority를 나타내는 방법을 제안한다. 일반적인 모델링 기법들은 데이터의 흐름이 니 시스템의 분석에는 적합하게 사용된다. 그러나 임의적으로 발생하는 이벤트에 대한 즉각적인 반응을 표현하기 어려운 단점이 있다. 본 논문에서는 기존의 모델링 기법들의 단점들을 보완하고 Statecharts라는 모델링 기법의 장점들을 혼합하여 경성 실시간 시스템에 가장 적합한 개선된 Statecharts를 제안한다. 그리고 Statecharts 모델링 기법의 구조적인 특성에 맞는 새로운 방법의 priority 표현법을 제안한다.

## 1. 서론

실시간 시스템이란 일반적으로 시스템의 작업 수행에 시간 제약과 비 주기적으로 발생하는 이벤트가 동시에 존재하는 시스템을 말한다. 실시간 시스템의 응용분야로는 항공 제어 시스템, 미사일 방어 시스템 등 수없이 많다. 이러한 실시간 시스템은 제한된 시간 내에 기대되어지는 작업을 최대한 빠르게 처리해 주는 것이 중요하다 그리고 무엇보다 실시간 시스템에서 중요하게 고려되어야 할 것은 임의적으로 생성되는 이벤트들에 대한 즉각적인 대응을 제한된 시간 내에 처리해 주어야 한다는 것이다[1].

실시간 시스템은 중요시한에 따라 경성(hard) 실시간 시스템과 연성(soft) 실시간 시스템이 있다. 경성 실시간 시스템은 시스템에 대한 고신뢰도가 요구되는 경우에 해당하며 이는 만약 태스크가 중요시한을 만족시키지 못하면 생명에 치명적 영향을 미치거나 막대한 재산 손실을 입는 경우와 같은 시스템이다. 연성 실시간 시스템은 태스크가 중요시한을 넘기더라도 경성 실시간 시스템에 비해 손실이 치명적이지 않은 경우에 해당된다[2].

실시간 시스템을 설계하기 위해 여러 가지 기법들이 제안되었다. 그 중에서 대표적인 것으로 flowcharts, Structure charts 및 Data Flow Diagrams 등이 있다. 이 기법들은 자료의 흐름을 표현하기 쉽

다는 장점을 가진 반면 프로세서들간의 상호작용이 쉽게 표현되지 않고 임의로 발생하는 이벤트에 대한 빠른 반응도 나타내기 어려운 문제점이 있다. 이러한 문제점을 해결하고 실시간 시스템에 필수사항인 동시성을 쉽게 표현하고, 연쇄적인 실행의 표현도 가능한 Statecharts가 Harel에 의해 제안되었다[3]. Statecharts는 기존의 DFD과 Context diagram에서 표현하기 힘들었던 연쇄반응과 이벤트의 발생을 한눈에 알아보기 쉽게 표현할 수 있는 장점을 가진다. 그러나 Statecharts는 이벤트 발생을 기본으로 한다는 점 때문에 전체 시스템의 자료흐름을 표현하지 못하는 단점을 가진다[1].

본 논문에서는 기존의 Statecharts에서는 표현하지 못하던 데이터 흐름의 내용을 보완해서 경성실시간 시스템을 설계하는데 적합하도록 Statecharts를 개량한 개선된 Statecharts 모델링 기법을 제안한다. 그리고 Statecharts에서 동시에 발생하는 이벤트들의 모호성을 배제하기 위해 새로운 priority 표현법을 제안한다.

본 논문은 제 2장에서 제안된 실시간 모델링 기법인 개선된 Statecharts의 특징과 장점을 설명한다. 제 3장에서는 동시에 발생하는 이벤트들의 모호성 배제를 위해서 제안된 priority 표현법의 특징과 장점을 설명한다. 제 4장에서는 본 논문의 결론과 앞으로 개선해야 할 사항에 대해 기술한다.

## 2. 개선된 Statecharts

### 2.1 개선된 Statecharts의 특성

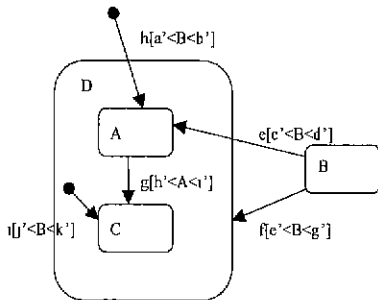
본 논문에서 제안되는 개선된 Statecharts 설계 방법은 DFD, Context diagram 그리고 기존의 Statecharts의 장점들을 최대한 고려하고 단점들을 보완하고 있으며 특성은 다음과 같다.

Enhanced Statecharts = state diagrams + orthogonality + broadcast communication + data flow diagram

위의 특성 중 orthogonality는 AND state에 의해 구분이 되고, broadcast communication은 전이에 의해 일어나며, 그리고 DFD(Data Flow Diagram)은 자료의 흐름을 표현한다. 즉 전이가 일어나면 이벤트가 발생되고 발생한 이벤트는 전체 시스템으로 broadcast된다. 그리고 자료의 흐름은 각 이벤트 발생이 있을 때 [ ] 안에 표현을 한다. 이는 어떤 상태에서부터 어떤 상태로 자료가 전이되는지, 전이되는 자료는 어떤 것인지를 나타낸다. 전이에 따른 상태들 사이에서의 전이 형태는 다음과 같다.

event[condition · data\_flow]/action

여기서 event는 상태에 따라 순간적으로 나타나는 발생이며, condition은 다음에 일어날 전이를 위해 만족되어야 하는 조건이고, data\_flow는 상태들 사이에서의 자료 흐름이고 action은 다른 이벤트를 발생케 한다



(그림 1) Enhanced Statecharts의 예

예를 들어 그림 1에서 이벤트 h가 발생할 때 전이되는 자료의 흐름은 B상태의 결과로써의 자료 a'가 현재 발생된 이벤트 h의 입력이 됨을 나타낸다. 즉, h라는 이벤트가 발생되면 상태 B에서의 결과 자료 a'가 현재의 상태 A로 전이된다. [a'<B<b']는 이벤트 h가 발생할 때 현재 발생한 상태 A에서 상태 B로는 자료 b'. 상태 B에서 상태 A로는 자료 a'가 전이됨을 나타낸다.

### 2.2 개선된 Statecharts의 장점

개선된 Statecharts를 이용해서 실시간 시스템을 디자인할 경우의 장점은 다음과 같다. 첫째, concurrent 프로세스들을 표현하기 용이하며 이벤트 발생에 따른 상태 전이 표현이 가능하므로, 시스템에 필요한 전반적인 이벤트의 발생을 효과적으로 나타낼 수 있다. 둘째, 기존의 Statecharts에서는 표현하지 못하던 각 이벤트의 발생에 따른 자료의 흐름을 나타낼 수 있다. 특히 제안된 기법에서는 기존의 Statecharts에서는 표현하지 못했던 자료의 흐름을 표현함으로써 전체적인 시스템의 자료 흐름을 한눈에 알아 볼 수 있다는 장점을 가진다. 특히 본 논문에서 제안하는 모델링 기법은 이벤트 발생과 동시성 그리고 주기적 시간 설정에 중점을 두었기 때문에 기존 Statecharts의 장점을 극대화시키고, 각 이벤트 발생에 따르는 데이터의 흐름을 모델링 기법에 추가함으로써 경성 실시간 시스템을 디자인하는 기법으로는 최적의 조건을 갖춘다.

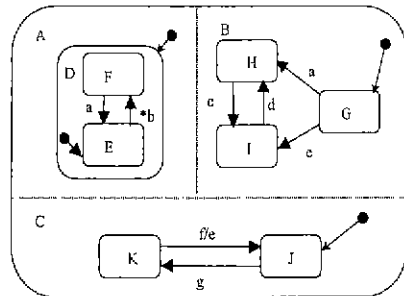
## 3. Priority

Statecharts에서 이벤트들이 동시에 발생하는 경우에 나타나는 반응은 비결정적이며 모호하다. 이러한 모호성을 해결하기 위한 기존의 방법에는 크게 두 가지가 있다. 첫째, 모델링에 사용되는 모든 이벤트들간의 우선 순위를 미리 결정해주는 방법이 있다. 이 방법은 시스템이 복잡해질수록 미리 정해두어야 하는 우선 순위의 관계가 많아진다는 단점이 있다. 둘째, Statecharts의 구조적인 특성을 이용한 방법이 있다. 이 방법은 이벤트들 간의 상호 연관성을 가지고 우선 순위를 정해주어 첫째 방법보다 표현의 간단함이 장점이다. 그러나 상호연관성이 없고 orthogonality가 다른 이벤트들 간의 모호성을 해결하지 못하는 문제가 있다[4]

본 논문에서는 기존 방법들의 장점을 가지면서도 표현하기 쉬운 방법을 제안한다. 기본적인 방법은 다음과 같다.

IF [ L(LCA(T1)) < L(LCA(T2)) ] THEN  
T1 has a higher priority than T2

여기에서 T1과 T2는 Transition이다. LCA(Least Common Ancestor)는 현재 발생한 Transition을 포함하는 바로 위의 상태이다[4]. 그리고 L은 각 상태의 단계를 나타낸다. 여기서 한가지의 예외가 있는데 '\*'를 가지는 Transition은 가장 높은 priority를 갖는다



(그림 2) Priority표현법의 예

그림 2에서 기본적인 상태인 A,B,C는 L이 1이다 그리고 직사각형의 상태로 하나씩 들어갈 때마다 L도 하나씩 증가한다.

그림 2에서 만약 이벤트 a가 F에서 발생했다면 Transition((G),(H))가 일어날 것이다. 그 이유는 LCA((F),(E))는 D가 된다. 그리고 이때의 L은 2이다. 반면에 LCA((G),(H))는 B가 되고 이때의 L은 1이다. 즉 Transition((G),(H))가 수행된다. 만약 이벤트 b와 f가 동시에 발생한다면 Transition((E),(F))가 수행된다. 비록 Transition((K),(J))의 L이 1로 Transition((E),(F))의 L인 2보다 낮지만 '\*'가 붙은 이벤트는 모든 것에 우선하기 때문에 Transition((E),(F))가 수행된다.

기존의 priority표현법들은 시스템의 구성이 복잡해짐에 따라 상호 관계 설정이 많아지고 상호 연관성이 없는 이벤트들의 발생에는 priority를 설정할 수 없는 단점을 가졌다. 본 논문에서는 기존의 방법들을 보완해서 동시에 발생하는 이벤트들의 모호성을 해결하는 새로운 priority표현법을 제안한다.

#### 4. 결론

본 논문에서는 기존의 Statecharts기법에서는 표현하지 못하던 자료 흐름을 모델링 기법에 추가한 개선된 Statecharts라는 새로운 디자인 기법을 제안한다. 제안된 기법은 DFD이나 Context diagram과 같은 실시간 시스템 모델링 기법들이 표현하기 어려웠던 주기적, 비 주기적 이벤트 발생, 동시적 태스크들의 수행 등을 나타낼 수 있는 기존의 Statecharts기법의 특성과 자료흐름을 표현할 수 있는 장점을 가진다. 그리고 동시에 발생하는 이벤트들의 모호성을 해결하기 위해 제안된 priority표현법은 상호연관성이 없는 이벤트들의 우선 순위를 결정할 수 있고 기존의 방법들에 비해 간단하다는 장점을 가진다.

개선된 Statecharts를 이용해서 정성 실시간 시스템을 디자인 할 때 이벤트 발생에 따른 자료흐름을 시스템 디자인시에 표현할 것인지의 여부를 결정하여야하는 문제점이 있다. 향후 개선된 Statecharts 모델링 기법을 기반으로한 실시간 시스템 디자인이 최적화 된 안전한 디자인인지를 실제 시스템 구성 전에 미리 보여줄 수 있는 방법이 연구되어야겠다.

#### 참 고 문 헌

[1] Phillip A. Laplante, "Real-Time Systems Design And Analysis," IEEE Press, 1997  
 [2] 김태웅, 신현식, "최악 경우의 여유 시간에 기초한 테이블 구동 비주기 실시간 태스크 스케줄링 알고리즘," 정보과학회 논문지 제 22권 제 12호, 1995. 12  
 [3] D. Harel, "Statecharts. A Visual Formalism for Complex Systems," Science of Computer Programming, 8, 3, pp 231-274, June, 1987.  
 [4] Hyoung Seok Hong, Jeong Hyun Kim, Sung Deok Cha and Yong Rae Kwon, "Static Semantics and Priority

Schemes for Statecharts," Proceedings of Computer Software and Application Conference '95, pp. 114-120, Dallas, Texas, Aug, 1995.  
 [5] D. Harel, A. Pnueli, J.P. Schmidt and R. Sherman, "On the Formal Semantics," Proceedings of Symposium on Logic in Computer Science, pp 54-64, New York, June, 1987  
 [6] A. Pnueli and M. Shalev, "What is in a Step: On the Semantics of Statecharts." Theoretical Aspects of Computer Science, LNCS 298, Springer-Verlag, pp 244-264, 1991.  
 [7] Harel, D. and A. Pnueli, "On the Development of Reactive Systems," in Logics and Models of Concurrent Systems, (K. R. Apt, ed), NATO ASI Series, Vol F-13, Springer-Verlag, New York, pp. 477-498, 1985.  
 [8] von der Beek, M., "A Comparison of Statechart Variants," in Formal Techniques in Real-Time and Fault-Tolerant Systems (Langmaack, de Roever and Vytupil, eds.), Lecture Notes in Computer Science, vol. 863, Springer-Verlag, New York, pp. 128-148, 1994