상황 시간 인식 서비스를 위한 프레임워크

A framework for situation and time aware services

이정은, 이지형
성균관대학교 정보통신공학부

Jung-Eun Lee, Jee-Hyong Lee
School of Information and Communication, Sungkyunkwan University
E-mail : papaxi@skku.edu, jhlee@skku.ac.kr

ABSTRACT

최근 우리 생활과 밀접하게 연관되어 있는 유비쿼터스 컴퓨팅 환경에서는 컴퓨터 스스로 사용자가 처한 환경을 인식하는 상황 인식(situation-awareness) 시스템이 필수적이다. 상황 인식(situation-awareness)시스템은 센서로부터 주변 환경 정보의 수집 및 결합을 통해 상황을 인식하고, 해석과 추론 처리과정을 거쳐 사용자에게 적절한 서비스를 제공한다. 주변 환경이 시간에 따라 계속 변화므로 더 좋은 서비스를 제공하기 위해서는 현재의 환경 정보뿐만 아니라 과거의 환경 정보도 고려하여 현재 상황이 어떻게 변화되고 있는지도 파악할 수 있어야 한다.

본 논문에서는 시간에 따라 동적으로 변화하는 유비쿼터스 컴퓨팅 환경에서 시간을 처리할 수 있는 상황 인식 프레임워크를 제안한다. 이 프레임워크는 변화하는 컨텍스트의 시간처리를 JESS와 JAVA를 이용하여 처리하였다.

Key words: 상황 인식(Situation awareness), 시간 논리(Temporal logic), 룰 베이스 시스템 (Rulebase system), JESS

I. 서 론

최근 우리 생활과 밀접하게 연관되어 있는 유비쿼터스 컴퓨팅 환경에서는 컴퓨터 스스로 사용자가 처한 환경을 인식하는 상황 인식(situation-awareness) 시스템이 필수적이다. 상황 인식(situation-awareness)시스템은 주변 환경으로부터 센서의 수집 및 결합을 통해 상황을 인식하고, 해석과 추론 처리과정을 거쳐 사용자에게 적절한 서비스를 제공한다.

상황인식 컴퓨팅은 1994년 Schili와 Theimer에 의하여 최초로 논의되었다. 여기에서 상황인식 컴퓨팅을 '사용 장소, 주변 사물과 물체의 집합에 따라 적응적이며, 동시에 시간이 경과하면서 이러한 대상의 변화까지 수용할 수 있는 소프트웨어'로 정의한다[1].

그러나 현재까지의 대부분의 시스템은 시간 정보를 적절히 반영 하지 못한다. 예를 들면, '방안의 온도가 30도이고 방안에 사람이 있으면 에어컨을 작동 시켜라', 이러한 상황 조건이 주어졌을 때 기존의 상황 인식 시스템은 센서로부터 입력된 값만을 가지고 조건에 맞으면 작동하도록 되어있다. 하지만 이러한 시스템이 실제적으로 구현되어 작동되기 위해서는 사람이 잠깐 들어왔다 가는 상황인지 방안에 머무르는 상황인지를 구분해야 한다. 즉 상황 표현에 사람이 얼마 동안 머물렀는지를 표현을 해줘야 한다. 이렇게 시간이 경과하면서 변화하는 객체를 표현하기 위한 방법 중에서 대표적으로 시간 논리가 사용이 된다.

본 논문에서는 시간에 따라 동적으로 변화하는 시간을 처리할 수 있는 룰 베이스 시스템 기반의 상황 인식 프레임워크를 제안한다. 룰 베이스 시스템은 여러 가지 상황에 따른 행동 규칙을 만들어 놓고 그 규칙에 만족하는 조건을 가지면 결정을 내리는 구조를 가지고 있다. 이러한 구조는 상황인식 시스템에서 입력된 컨텍스트의 이름과 값에 따른 해석 및 추론 처리

과정을 거치는 구조에 적합하다.

본 논문에서는 룰 베이스 시스템에서 시간을 고려한 상황인식 프레임워크를 제안함으로써 이러한 문제점을 보완한다. 룰 베이스 시스템에서 상황 인식 서비스를 사용하기 위해 센서로부터 제공 받은 정보를 Predicate 통해서 컨텍스트를 표현하고, 시간 논리를 적용하여 시간 관계를 표현한다. 또한, 다수의 센서 정보로부터 규칙과 일관성 있는 매칭을 통해서 적합한 사실을 추론하기 위해 JESS의 RETE 알고리즘을 사용한다.

본 논문은 다음과 같이 구성되어 있다. 2장에서는 관련연구로 상황인식과 시간논리의 연구를 살펴본다. 3장에서는 제안한 상황 인식 프레임워크를 기술하고, 각각의 모듈의 역할에 대해 설명한다. 마지막으로 4장에서는 향후 연구 방향에 대해 언급하고, 본 논문에 대해 결론을 맺는다.

II. 관련 연구

2.1 상황 인식

본 장은 유비쿼터스 컴퓨팅 환경에서 사용자가 처한 환경을 인식하는 상황인식 기술에 대해 소개한다.

상황 인식(situation-awareness)은 단지 상황 정보를 기반으로 하여 향후 시스템 상태가 어떻게 변화 할 것인가에 대한 예측으로 정의된다[2]. 또한, 앤슬리는 "상황 인식을 다량의 시간과 공간 안에서 환경 요소를 인지하고, 그것들의 의미를 이해하며 가까운 미래에 일어날 상태를 예상하는 것"으로 정의를 하였다[3].

본 논문에서는 상황(situation)을 사용자주변 환경이 특정 조건을 만족하며 일정 시간 유지 혹은 변화 하는 것이라고 정의한다.

$$\text{Context} = \frac{\partial\, \text{Situation}}{\partial T}$$

그림 1. 상황 인식 표현

따라서, 상황인식(situation awareness)를 위해서는 센서로부터 얻어지는 환경정보 뿐 아니라 시간까지도 함께 표현하고 다룰 수 있는 기법 이 필요하다.

상황 처리를 위한 연구로 SA-CSL이 있다. SA-CSL에서는 컨텍스트와 액션의 집합을 상황으로 정의하고 상황을 표현하기 위해서 컨텍스트 튜플, 액션 튜플, 분석 결과를 사용한다. 표현 방법은 아래와 같다[3].

- ◆ 컨텍스트 튜플: <t, c1, c2…, cn>
- ◆ 액션 튜플: <t, a1, a2,…, an>
- ◆ 상황 표현: [∀, ∃] t in <time range> [context, analysis result, action, situation] <compare> <value>

T는 시간을 나타내고, c1, c2, …, cn은 컨텍스트 애트리뷰트를 표현하며, a1, a2… an은 액션 애트리뷰트라고 한다. SA-CSL은 컨텍스트를 구체적으로 표현하는 방법을 제시하고 있지만 컨텍스트 정보를 애트리뷰트로만 나타내어서는 실제 복잡하고 다양한 컨텍스트를 모두 표현할 수 없다.

2.2 시간 논리

본 본문에서 시간을 표현하고 시간 관련 추론을 하는데 있어서 시간 논리(temporal logic)을 사용하였다.

시간 논리는 기본적인 논리식에서 사용 되는 연산자들을 시제 논리 연산자를 추가한 형태로 구성된다. 기본적으로 사용되는 알렌이 정의한 시제 논리 연산자는 다음 표 1과 같다[5].

◇	□	○	U	P	∀	∃	¬	⇒	⇔
결국	항상	다음	까지	수행	모두	존재	부정	함축	동등

표 1. 시간 논리 연산자

알렌(Allen)은 1983년에 간격의 개념을 기초로 하여 Time Logic Interval(TLI)을 제안했으며, 그림 2는 알렌의 TLI 관계도로서 아래 표2와 같이 정의할 수 있다[6].

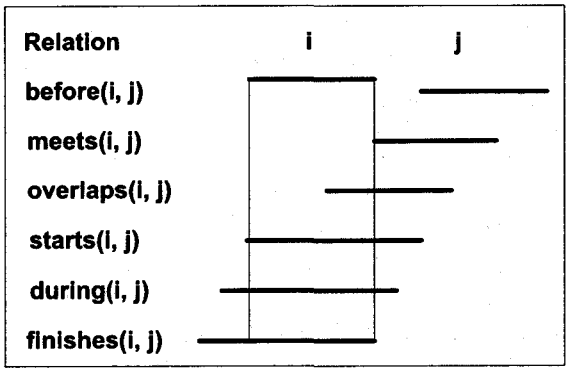

그림 2. 알렌 TLI 관계도

본 논문에서는 알렌의 TLI 중에서 During를

사용하여 현재 시스템에 적용하여 시스템에 관계된 사건이나 상태들이 어떤 순서로 일어나야 하는지를 기술하는데 사용하였다. TLI의 다른 관계들까지도 점차적으로 시스템에 적용할 것이다.

관 계	정 의
before(i, j)	i의 끝은 j가 시작하기 이전을 나타낸다.
meets (i, j)	j의 시작이 정확하면 i는 끝난다.
overlaps(i, j)	i는 j전에 시작하고 i는 j전에 끝난다.
starts(i, j)	i는 j와 동시에 시작을 했지만 먼저 끝난다.
during(i, j)	i가 시작한 후에 끝나는 시점이 j 이전 이다.
finishes(i, j)	i의 시작은 j보다 늦지만 같은 시간에 끝난다.
equals(i, j)	모두 같은 간격을 가진다.

표 2. 알렌의 TLI 관계에 대한 정의

III. 상황인식 프레임워크

본 장에서는 센서로부터 입력된 상황 인식 모듈 과정을 프레임워크로 제안을 하고 각 모듈이 담당하는 기능을 설명을 한다. 또한 작업 흐름도를 통해 시스템의 흐름을 보다 쉽게 이해 할 수 있도록 하였다. 먼저 제안하는 프레임워크는 아래 그림3과 같다.

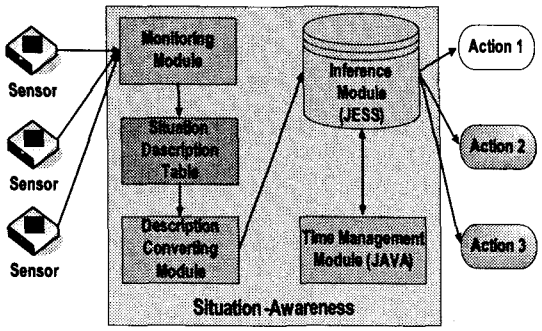

그림 3. 상황 인식 프레임워크

3.1. 각각의 모듈 설명

1) Monitoring Module

Monitoring 모듈은 주위 환경에서 센서를 가진 디바이스로부터 센서 정보가 저장되는 부분을 담당한다. 센서로부터 입력되는 값을 표 3에서 볼 수 있듯이 세 개의 속성으로 정하였고, JESS의 Deftemplate을 사용하여 새로운 센서 값이 입력 될 때마다 fact로 추가되어 저장되어 관리 된다.

속성	설명
name	센서로부터 입력된 컨텍스트 이름
value	센서로부터 입력된 컨텍스트 값
time	컨텍스트 값이 입력된 시간

표 3. 상황 인식 템플릿의 속성

```
(deftemplate situation "sensed data"
        (slot name)
        (slot value)
        (slot time))
```

그림 4. 상황 인식 템플릿 구조

2) Situation Description Table

Situation Description Table은 아래 그림 5와 같이, 센서로부터 제공 받은 정보를 Predicate 기반으로 표현하였고, 알렌의 시간 논리를 적용하여 시간 관계를 포함 한다. 그림 5의 ①은 온도가 30이상이면서 5초 동안 유지되면 에이컨을 작동한다". 또한, ②는 "습도가 40이상이면서 6초 동안 유지되면 보일러를 작동 한다"라는 상황정보를 Predicate 표현 방식이다. 그림에서 CT는 현재 시간을 의미하며, Interval(t1, t2)는 시각t1부터 시각 t2까지의 시간구간에서 참인 가상의 사건을 의미한다. 따라서 During(Interval(t1, t2), p)는 사건 p가 시간 t1부터 t2 동안 계속 유지되었음을 의미한다.

```
① During(Interval(CT-5, CT), Temperature(>30))
->turn on air-condition
② During (Interval(CT-6, CT), Humidity(>40))
-> turn on boiler
```

그림 5. Predicate 기반 상황 인식 표현

3) Description converting Module

Description converting 모듈은 룰 베이스 시스템에 저장되기 전에 JESS 규칙에 맞게 룰의 형태로 변환하는 모듈이다.

```
①-1 (defrule check-temp
(not (situation (name temp- 30) (value true)))
(situation (name temp) (value ?v))
(test (>= (?v 30)))
=> (assert (situation (name temp-30) (value true)
                (time ?*ct*))))
①-2 (defrule air-conditioner
(situation (name temp-30) (value true) (time ?t))
(test (>= (- ?*ct* ?t)))
=> (air-conditioner on))
```

그림 6. JESS 형식의 규칙

예를 들어, 그림 5의 ①은 그림 6의

①-1과 ①-2의 두 규칙으로 표현될 수 있다. ①-1은 온도가 30도 이상인가를 확인하는 규칙이고 ①-2는 30도 이상인 상태로 5초 이상 유지되었는가를 확인하는 규칙이다. 즉 온도가 시간에 따라 유지/변화하는 상황을 표현한 것이다. 규칙에서 ?*ct*는 현재시각을 갖고 있는 변수이다.

4) Time management Module

그림 6과 같은 규칙이 JESS에서 적절히 작동하기 위해서는 JESS 안에서 계속하여 현재시각을 나타내는 변수 ?*ct*를 계속 갱신해야 한다. 그러나 JESS에서는 기본적으로 현재시각을 확인할 수 있는 방법이 없기 때문에 외부에서 이를 처리해 주어야 한다. 이를 위해서 JESS와 JAVA를 연결하여 JAVA 쪽에서 현재 시각 정보를 JESS에 넣어준다. 이러한 작업을 하는 모듈이 Time management Module이다.

5) Inference Module

Inference 모듈은 조건 제어 규칙(If-Then Rule)을 가진 구조로써, 센서로부터 입력된 정보를 규칙에 맞게 매칭, 추론하여 사용자 상황이나 주변 상황이 변할 때마다 적응적인 서비스를 제공한다. 본 논문에서는 추론을 하기 위해 룰 베이스 시스템 기반의 JESS를 사용한다.

IV. 결 론 및 향후 연구

본 논문에서는 유비쿼터스 컴퓨팅 환경에서 시간에 따라 동적으로 변화하는 시간을 처리할 수 있는 상황 인식 프레임워크를 제안했다. 또한, 룰 베이스 시스템 기반의 JESS를 추론 모듈로 적용 했으며, 현재의 시간 값을 처리할 수 없는 부분을 자바를 통해 처리하였다. 따라서, JAVA를 통해 현재의 시간 정보를 기준으로 센서 정보가 들어 올 때의 시간과의 차이에 따라 매 1초마다 상황 정보의 시간 값을 업데이트 하였다. 또한 수많은 센서 정보로부터 추론을 하여 규칙과의 일관성 있는 매칭을 통해서 사실을 찾아내기 위해 JESS의 RETE 알고리즘을 사용했다. 그 결과로, 센서 정보에 일정 시간 동안 지속된 상황을 표현함으로써 그 동안 상황 인식에서 현재 시간만 고려한 방법과 다르게 일정 시간 동안 시간까지 고려함으로써 룰 베이스 시스템에서 효율적으로 상황에 맞는 추론을 하였다.

향후 본 논문에서 제안한 방법을 유비쿼터스 환경에서 변화하는 환경과 그 안에서 변화하는 시스템에 적용할 계획이다.

감사의 글: 본 연구는 정보통신부 2005년 유비쿼터스 환경에서의 환경/사용자의 적응 서비스 기술개발의 연구비 지원으로 수행되었으며 연구비 지원에 감사 드립니다.

V. 참고문헌

[1] 임신영, 허재두, 박광로, 김채규, "상황인식 컴퓨팅 기술 동향", IITA 주간기술동향, 제1142호, pp.1-15, 2004.

[2] M.R. Endsley, D.J Garland, "Situation awaress Analysis and Measurement", pp.1-24, 2000.

[3] S.Yau, Y.Wang, F.Karim, "Development of Situation-Aware Application Software for Ubiquitous Computing Environments", 26th Annual International Computer Software and Applications Conference, pp. 233-238, 2002.

[4] M.R. Endsley, "Theoretical underpinnings of situation awareness: A critical review", Lawrence Erlbaum Associates, Publishers, Mahwah, New Jersey, pp. 3-32, 2000.

[5] A. Ranganthan, R.H. Campbell, "An infrastructure for context-awareness based on first order logic", Personal and Ubiquitous Computing, pp.353-364, 2003.

[6] A. Artale, E. Franconi, "A survey of temporal extensions of description logics", Annals of Mathematics and Artificial Intelligence, pp.171-210, 2000.