센서 네트워크에서의 Rule-Base 기반 상황 인식 미들웨어*

김금란°, 김진아, 김세웅, 김창화, 김상경, 박찬정
강릉대학교 컴퓨터공학과
{orange0420°, noanoa98, sewoong98}@nate.com, {kch, skkim98, cjpark}@kangnung.ac.kr

A Rule-Base based Context-aware Middleware in Sensor Network

GeumLan Kim, J.A. Kim, S.W. Kim, C.H. Kim, S.K. Kim, C.J Park
kangnung National University Dept. of Computer Science &Engineering

요 약

상황인식 처리기술은 사용자의 컨텍스트에 근거하여 사용자와 장치간의 상호 운용성을 지원해 줌으로써, 사용자로 하여금 정보 획득 및 실행을 보다 용이하게 하도록 해주는 기술이다. 여러 상황변화가 발생할 경우 이에 따라 서비스 전달 방식을 동적으로 적응시키기 위하여 상황에 대한 명시적인 요구사항을 정의하고 이 정의 된 상황을 각 노드에 전파하고 노드에 포함된 미들웨어는 전파된 상황에 적합한 센싱된 정보를 분석하여 특정한 상황의 발생과 이에 따른 액션을 수행하며 또한 센서에서 취득한 컨텍스트로 부터 통합된, 추론된 컨텍스트를 생성한다. 본 논문에서는 센서로부터 다양한 타입의 컨텍스트를 처리 할 수 있는 미들웨어를 제안한다. 이 미들웨어는 변화하는 주변 환경에서 센서로부터 센싱된 컨텍스트 뿐 아니라 통합된, 추론된 컨텍스트를 생성할 수 있도록 설계 되었다. 제안한 상황인식 미들웨어를 기반으로 사용자 질의 요청과 이벤트 상황 질의를 설계하였으며, 이벤트 상황 질의의 추론 DB를 명시하기 위하여 Clips 언어를 사용하였다.

1. 서 론

상황(Context)은 사용자와 시스템간의 상호 작용에 연관된 사람, 장소, 사물의 특징을 형성하는 가능한 모든 형태의 정보이며 사용자의 현재 상황에 따라 적절한 정보 혹은 서비스를 제공하기 위해 상황을 이용하는 것을 상황인식(Context-awareness)라 한다[1].

센서네트워크를 이용하여 사용자가 원하는 현재의 상황의 데이터를 수집만 하는 것이 아니라 주변공간의 상황을 자동적으로 인지하고 또한 사용자들이 물리적으로 떨어져 있더라도 원하는 곳의 사물과 주변 환경의 변화(어떤 지역의 지진, 화재, 위험한 물질 또는 적군의 감지등의 상황 정보)를 인식하거나 추적하여 그에 따른 적절한 정보와 서비스를 제공할 필요가 있다[2].

상황인식 시스템은 다양한 센서들로부터 얻어진 의미없는 정보들을 사용자가 원하는 정보로 가공처리 하여 정보를 제공하며 센싱된 정보를 사용자가 원하는 형식의 정보로의 변환만이 아니라 센싱된 정보 혹은 단순히 가공된 정보들을 이용하여 정의된 특정한 상황을 인식하는데 이러한 정보를 사용하며 더 나아가 여러 가지 상황을 조합하여 상위 레벨의 컨텍스트를 추론하도록 한다[2].

본 논문에서는 Rule-Base의 추론 엔진에 미리 패턴과 액션으로 구성된 상황을 정의하고 패턴에 따라 센싱된 정보를 가공하여 가공된 정보가 패턴에 부합되는지 감시하게 된다. 만약 정보가 상황에 정의된 패턴을 모두 만족한다면 상황에 정의된 액션에 따라 사용자에게 상황을 알리거나 센서노드 스스로 상황에 맞는 조치를 취할 수 있는 미들웨어를 제안한다.

본 논문에서의 구성은 1장 서론에 이어, 2장에서는 현재 사용되고 있는 상황인식 미들웨어와, 컨텍스트에 대하여 살펴보고, 3장에서는 질의 요청 방법에 대하여 살펴보고, 4장에서는 본 논문에서 제안하는 상황인식 미들웨어에 대하여 살펴보고, 5장에서는 상황인식 미들웨어의 처리 과정을 살펴보았으며, 마지막으로 6장에서는 결론으로 끝 맺었다.

2. 관련연구

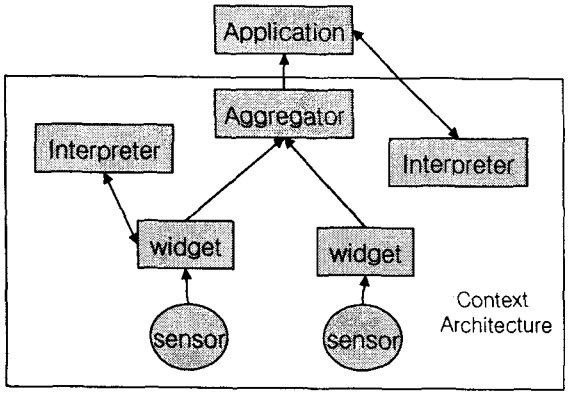
2.1 상황인식 미들웨어

컨텍스트에 대한 연구는 몇 년 전부터 계속되어 왔다. 그 중 가장 중요한 연구는 Context Toolkit[3]이라고 할 수 있다. Context Toolkit 에서는 컨텍스트를 사람, 장

* 본 연구는 정보통신부 및 정보통신연구진흥원의 대학 IT연구센터 지원사업의 연구결과로 수행되었음(IITA-2005-C1090-0501-0010)

소, 물체 이 세 가지 개체의 상황을 설명하는 정보로 정의하고, 컨텍스트를 Identity, Location, Status, Time의 4개의 타입으로 분류한다. 이 미들웨어의 특징은 센서로부터 취득한 컨텍스트 정보를 표현 및 통합을 한 후 애플리케이션에서 제공한다[4].

그림 1 은 Context Toolkit 의 컨텍스트 컴포넌트들을 보여준다.



[그림 1] 컨텍스트 컴포넌트

Context Toolkit 은 컨텍스트 정보를 이용하는 어플리케이션을 개발함에 있어서 출발점을 제공하지만 컨텍스트의 구체적인 표현 방법이나 컨텍스트에 대한 추론 방법을 제공해 주지 못하고 있다.

2.2 Data Aggregation

센서네트워크에서 노드의 배치는 매우 조밀하게 하여 같은 지역을 여러 노드가 중복해서 감시하도록 한다. 이는 하나의 이벤트에 대해 여러 노드가 동시에 감지하여 중복된 데이터를 발생할 수 있으며 특히 인접한 노드들은 같은 데이터를 반복적으로 전송할 수 있다. 이를 효과적으로 병합하지 않는다면 네트워크 전체의 데이터 전송량이 증가되어 에너지 소모가 많을 수밖에 없다.

이를 해결하기 위해 센서네트워크에서는 네트워크 내부에서 데이터를 병합하여 중복된 데이터를 배제하고 구분되는 이벤트만을 싱크로 전송하기 위한 방법이 필요하다. 이때 중요한 것은 어디에서 데이터 병합을 수행할 것인가 하는 것이다. 데이터 병합 방법은 발생하는 센서 데이터와 애플리케이션에 따라 다르기 때문에 이를 일반화하여 적용시킬 수 있는 있어야 한다. 데이터 병합 점(point) 에 대한 연구는 그룹이나 클러스터를 구성하여 그 내부에서 발생하는 데이터를 병합하거나 싱크로 전송되는 경로 상에서 병합을 이룰 수 있는 최적 트리를 구성하는 방법 등이 있다[5].

3. 센서네트워크에서의 질의 요청

3.1 질의 요청 방식

사용자에 의한 질의 요청이 발생했을 경우 미들웨어에서 질의를 분석하고 분할하여 query저장 DB에 질의를 저장한다. 이때 사용하는 질의 방식으로는 공간 질의, 시간 질의, 변화하는 센싱 정보를 위한 질의로 구분하여 질의를 할 수 있으며, 질의에 맞는 형태로 센싱된 정보는 최하위 노드에서 상위 노드로 전달되면서 데이터 병합을 하게 된다.

이때 질의 방법에 대해 살펴보면 어느 특정지역의 특정시간 혹은 실시간적으로 질의 요청이 있을 수 있다. 이를 위해 (실)시간 공간 질의가 필요하며 공간 연산이 전체 실행 시간에서 차지하는 시간비중이 크기 때문에 실시간 공간 데이터의 유효성을 유지하면서 (실)시간 공간 질의의 시간제약조건을 만족시킬 수 있는 질의 처리 기법이 필요하다[5].

첫 번째 공간질의(윈도우 쿼리)를 위하여 SQL의 where절 이후에 다음과 같이 연산식을 사용한다. 연산식은 SQL 문의 where절 이후에 공간을 지정할 수 있는 조건식을 추가한다.

```
SELECT AVG(TEMP, LIGHT)
  FROM sensors
  WHERE  "공간 범위, 시간"
```

```
Circle (Point, R)
rectangle (Point, Point)
polygon (Summit)
Summit : Point | Point, Summit
Point : (X,Y)
R : 반지름
```

두 번째 시간 질의를 위하여 SQL의 where절 이후에 다음과 같이 연산식을 사용한다.

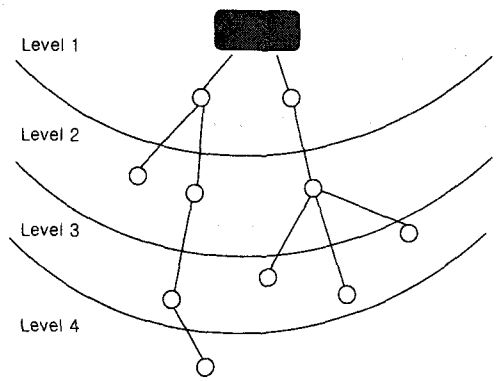
(Max | Min) attribute Start Time End Time
(Average) attribute Interval clock Start Time End Time
attribute operator threshold Start Time End Time
time : 시간
operator : <, >, =, !
threshold : 속성 값

마지막으로 변화하는 센싱 정보를 위하여 다음과 같은 연산식을 사용한다.

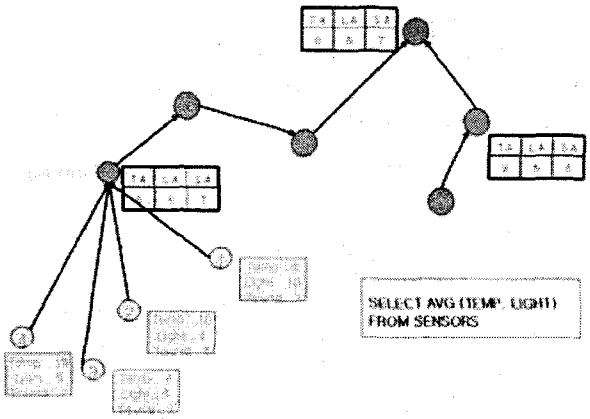
Select attribute From Sensor Where expr Situation S
S : (ascend | descend | sameness) Neighbor (near | all | point | me) Base Time During Minute
S : S (and | or) S | S

3.2 질의 요청에 따른 질의 최적화

센서의 배치가 매우 조밀하여 한 지역을 여러 개의 센서가 중복하여 감시하는 경우가 발생하기도 한다. 이렇게 중복된 센싱 정보를 아무런 가공을 하지 않고 상위 노드로 전송 할 경우 데이터 전송양이 증가하여 센서들의 Life-Time을 단축하는 경우가 발생한다. 따라서 센싱된 정보를 각각의 노드는 사용자 요구에 적합한 데이터로 병합하여 상위 노드로 전송해야 한다. 이러한 방식을 취하기 위하여 [그림 2]와 같이 노드들을 트리 형태로 구성되며 상위 노드로부터 얻어진 질의는 하위노드에 적합하게 분해하여 전송하게 된다. 다음 질의에 맞는 결과를 상위 노드로 전송하기 위해서는 수신된 결과를 [그림 3]과 같이 병합하여 전달하게 된다.



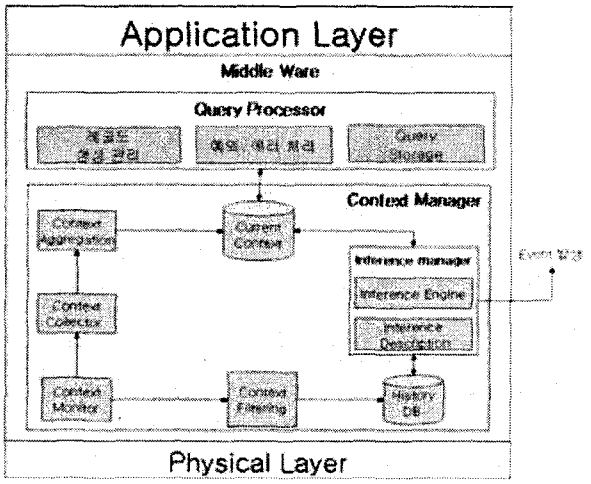
[그림 2] 노드들의 트리 구조



[그림 3] 트리형태로 구성된 노드의 데이터 전송

4. 제안한 상황인식 미들웨어

4.1 미들웨어 구조



[그림 4] 미들웨어 구조

본 논문에서 제안한 상황인식 처리를 위한 미들웨어는 [그림4]와 같이 크게 Query processor와 Context Manager 2부분으로 나뉘게 되고, Context Manager는 다시 사용자의 질의 요청에 따라 질의를 처리할 수 있는 부분과, 여러 현상에 대한 Event 상황이 발생했을 경우 이에 대해 사용자에게 빠르게 알려줄 수 있는 Inference Manager로 구분된다.

4.2 Query processor

Query processor는 Query를 처리하기 위하여 분석하는 기능을 하는 부분으로 레코드 생성관리, 예외·에러 처리, Query Storage로 구성된다.

4.2.1 레코드 생성 관리

Context Manager의 상황 추론에 따른 Action 메시지 혹은 사용자 질의에 의해 얻어진 센싱 정보 결과를 상위 노드 혹은 사용자에게 전달 할 수 있도록 레코드를 생성한다.

4.2.2 예외·에러 처리

Application으로부터 상황을 정의하거나 질의가 들어올 경우 질의를 분석하여 잘못된 정보를 요구하거나 질의에 오류가 있을 경우 오류를 처리 하고 센서 노드의 상태가 원래 상태로 복귀 할 수 있도록 한다.

4.2.3 Query Storage

상위 노드에서 질의 요청이 발생했을 경우 발생된 질의가 각 하위 노드로 전달되기 위해서는 질의를 분석하고, 분해 하게 된다. 이때 분해 되어진 질의를 저장한다.

4.3 Context Manager

본 논문에서는 상황 인식 미들웨어 시스템을 제공하기 위해 Context manager를 제안한다.

· Context Monitor : 센서로부터 추출한 정보를 받아들인다.

· Context Collector : Context Monitor를 통해 얻어진 현재의 상황 정보를 모아 일시적으로 저장하고 관리한다.

· Context Aggregation : Context collector가 가지고 있는 센싱 정보를 이용하여 사용자가 원하는 형태의 정보로 병합한다.

· Context Filtering : Context Monitor로부터 제공받은 context 정보 중 중복된 데이터 또는 불필요한 데이터를 제거한다. 그리고 유용한 context만을 선택하여 History DB에 저장한다.

· History DB : Node 위치 정보와 Sensor로부터 얻어진 상황정보(온도, 습도, 조도 등)를 가지고 있다.

· Inference Manager : 각 상황에 맞는 Event를 발생할 수 있도록 하기 위하여 상황에 대한 정보와 상황에 대한 대처를 할 수 있도록 유저에게 전달할 메시지를 저장하고 있다.

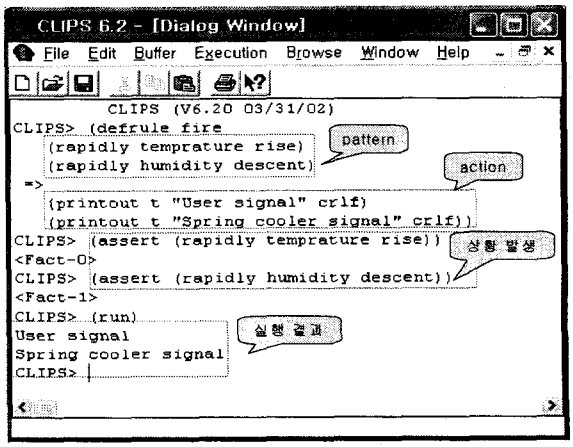
· Inference Engine : 환경에서 일어날 수 있는 여러 상황 정보를 사용자에게 빠르게 알려줄 수 있도록 만들어진 추론 엔진으로 Inference Description에 정의된 상황정보와 비교하여 현재의 상황을 추론할 수 있도록 한다.

· Inference Description : 여러 정의된 상황 정보를 저장하는 곳으로 Clips 언어를 사용하여 상황정보를 등록하였으며, Clips 언어의 구성요소는 Rule, Fact, Object로 구성된다.

Rule : Rule은 패턴과 액션으로 구성되어 있으며 룰에 정의된 모든 패턴이 Fact로 존재할 경우 정의된 액션이 실행된다.

Fact : 현재의 상황 혹은 통합된 정보를 입력

```
Rule의 삽입
(defrule RuleName  "optional_comment"
  (pattern_1)
  (pattern_2)
  (pattern_n)
=>
  (action_1)
  (action_2)
  (action_n)
```



[그림 5] Clips 사용 예

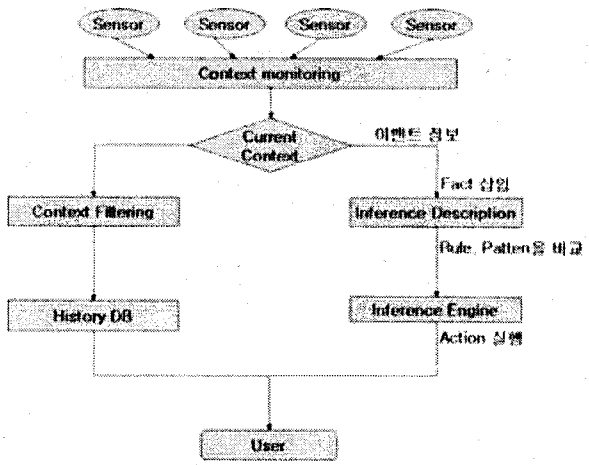
[그림5]는 Clips언어를 사용하여 pattern을 삽입하는 예를 보여주고 있다.

Inference Description에 정의되어있는 상황정보와 센서로부터 얻어진 상황 정보가 모두 만족 할 경우 정의된

액션을 수행하며 이 액션은 사용자 혹은 상위 센서노드 등에 상황을 전파하게 되고 또한 다른 상황을 추론하는 데 쓰일 패턴을 삽입하는데 사용된다.

5. 미들웨어 처리 과정

미들웨어의 전체적인 처리 과정은 [그림6]과 같으며 동작은 크게 두 부분으로 구분할 수 있다. 첫 번째로 사용자에 의해 질의 요청이 발생했을 경우는 [그림6]의 왼쪽 부분과 같이 센서로부터 환경에 대한 정보를 추출하여 상위 노드로 보내게 되면 Context Monitor에 의해 정보를 수집하게 되고 수집된 정보들 중 중복된 정보 또는 불필요한 정보가 있을 경우 Context Filtering을 거쳐 유용한 정보만을 추출하게 된다. 이 추출된 정보를 History DB에 저장하고, current Context에서는 추출된 정보를 질의에 맞는 형태로 변경하여 사용자 또는 상위 노드에 전달하게 된다. 두 번째로 이벤트 상황이 발생했을 경우는 [그림6]의 오른쪽과 같이 센서로부터 센싱된 정보를 Inference Description에 정의된 Rule과 Pattern을 비교 · 분석하여 현재의 상황을 추론하게 되고 Inference Engine에서는 이에 해당하는 상황 정보를 사용자 또는 상위 노드에게 빠르게 알려주기 위하여 Action을 실행하게 된다.



[그림6] 미들웨어 처리 과정

6. 결론

본 논문에서는 추론 DB를 이용한 상황인식을 위한 미들웨어를 제안 한다. 제안된 미들웨어는 동적으로 변화하는 주변 환경에서 센서로부터 센싱된 컨텍스트 뿐 아니라 통합된, 추론된 컨텍스트를 생성할 수 있도록 설계하여 사용자에게 센서 스스로 판단하여 상황정보를 빠르고 정확하게 알려주며 여러 가지 상황에 알맞은 조치를 취할 수 있다.

7. 참고문헌

[1] 최진영 외4인, "위치정보 및 시간정보 기반 상황인식 서비스 모델", 한국정보처리학회 춘계학술발표대회 논문집 제12권, 2005. 5.

[2] 정덕진 외3인, "상황인지 센서네트워크 기술동향"

[3] A. K. Dey, D. Salber, and G. D. Abowd, "A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications", Human-Computer Interaction, 16(2-4):97-166, 2001.

[4] 이상학, "센서네트워크 기술동향", 전자부품 연구원, 2005.

[5] 임정옥 외2인, "시간적 제약을 갖는 공간 질의 처리를 위한 실시간 연산 후배치 기법", 정보과학회 논문지, 2001. 6.

[6] Joseph C. Giarratano. Ph.D. Abowd "Clips User's Guide", 1998.

[7] 심춘보, 태봉섭 외 3인, "상황인식 처리를 위한 미들웨어 및 컨텍스트 서버를 이용한 응용시스템의 구현" 2005.

[8] 박현정, 이지형, "유비쿼터스 컴퓨팅 미들웨어를 위한 상황 인식 프레임워크"

[9] A. Ranganathan, R. H. Campbell, "A middleware for context-Aware Agents in ubiquitous Computing Environments", international Conference EUC 2004, Vol.3207, pp. 672-681, 2004.

[10] 장호진, 염근혁, "적응형 소프트웨어를 위한 컨텍스트 모델링", 2004.