유비쿼터스 환경을 위한 상황인지 및 학습, 추론 모델

지동준° 양정진
가톨릭대학교 컴퓨터정보공학부
{j_jun16°, jungjin}@catholic.ac.kr

Context-aware and Reasoning Model for Ubiquitous

Dong-Jun Ji°, Jung-Jin Yang
Catholic Univ. of Korea Dept., CSIE

요 약

유비쿼터스는 인간의 일상생활에 깊이 스며들어 삶을 풍요롭게 만들어 주는 기술이다. 즉, 여러 형태의 센서가 인지하는 상황정보를 바탕으로 인간을 위한 다양한 목적을 이루어 낼 수 있다. 각각의 유비쿼터스 시스템은 각자의 구조를 가지지만 상황인지(Context-aware), 학습(Learning), 추론(Reasoning) 의 요소는 대부분 필수적으로 갖추고 있다. 본 연구에서는 위 세가지 기본요소를 조합해서 구현할 수 있는 프레임워크를 제시하고 시나리오를 통해 그 적용 가능성을 살펴본다.

1. Introduction

유비쿼터스는 인간의 일상생활에 깊이 스며들어 삶을 풍요롭게 만들어 주는 기술이다. 즉, 여러 형태의 센서가 자동으로 인지하는 상황정보를 바탕으로 인간을 위한 다양한 목적을 이루어 낼 수 있다. 각각의 유비쿼터스 시스템은 각자의 구조와 구성요소로 이루어지지만 상황인지(Context-aware), 학습(Learning), 추론(Reasoning)의 3가지 요소는 대부분이 필수적으로 갖추고 있다. 시스템은 이들 요소를 이용하여 사용자가 시스템을 의식하지 않고 사용할 수 있도록 하는, 유비쿼터스 본연의 목적을 도울 수 있다. 본 논문이 제시하는 상황인지 및 학습, 추론 모델은 위의 3가지 요소로 이루어져있으며, 센서들의 상황인지 결과를 축적하여 학습의 자료로 사용한 뒤, 그 결과물을 근거로 하여 새롭게 인지되어오는 상황정보나 기존 데이터의 결과를 추론하는 규칙기반추론(Rule-Based Reasoning : RBR)을 수행한다. 이처럼 일련의 과정으로 이루어지는 상황인지 및 학습, 추론은 하나의 프레임워크로 묶일 수 있으며 그 세부적인 구조는 다양할 수 있다. 설계자가 어떤 유비쿼터스 시스템을 구상할 때 이들의 흐름을 잘 이해하고 목적에 따라 효율적으로 재구성하는 것은 매우 중요하다 할 수 있다. 본 연구에서는 이를 위한 기초적인 프레임워크 모델을 제시하고 이해를 돕기 위해 시나리오에 적용하여 설명한다.

본 논문은 다음과 같이 구성된다. 2장에서는 온톨로지에 대한 간략한 설명과 학습, 추론을 위한 실제 어플리케이션을 소개하고 3장에서는 이를 바탕으로 설계한 상황인지 및 추론모델을 제시하며 적용 가능한 시나리오를 설명한다. 4장에서는 결론과 향후과제를 제시한다.

2. Related Works

2.1 Ontology

온톨로지의 개념을 설명할 때 자주 인용되는 Tom Gruber 의 정의에는 온톨로지는 개념의 명세(An ontology is a specification of a conceptualization.)로써 에이전트나 에이전트 커뮤니티의 추상개념 혹은 관계의 표현이라고 풀이되어 있다. 즉, 표현의 논리적 형식보다는 개념이나 관계 등의 의미를 중시하여 인간이 갖고 있는 추상적 개념을 컴퓨터 상에 표현하고, 이런 특성을 이용한 지식의 공유와 재사용에 그 목적이 있다. 온톨로지 표현 형식으로는 W3C 에서 표준으로 제안하는 RDF[1] 와 그보다 확장된 표현력을 지니는 OWL[2] 등이 대표적이다.

2.2 WEKA

학습을 위한 어플리케이션으로 WEKA[3]를 사용할 수 있다. WEKA는 machine learning(ML) techniques 를 실현하기 위해 개발되었다. JAVA로 작성되었으며 ARFF 이라는 자체 데이터파일 포맷을 사용하는데 이것은 MS Excel 의 CSV 포맷과 같은 comma-separated format 의 일종이다. WEKA는 [표1]과 같은 데이터셋을 자료로 학습을 진행하며 이를 위한 여러 가지 학습알고리즘들이 준비되어 있다. 그 중 사용자가 선택한 방법으로 데이터셋을 분석하고, 그 결과로 얻은 규칙형태의 지식을 이용해서 데이터셋의 결과를 예측할 수 있다.

```
@relation weather.symbolic

@attribute outlook {sunny, overcast, rainy}
@attribute temperature {hot, mild, cool}
@attribute humidity {high, normal}
@attribute windy {TRUE, FALSE}
@attribute play {yes, no}

@data
sunny,hot,high,FALSE,no
sunny,hot,high,TRUE,no
overcast,hot,high,FALSE,yes
rainy,mild,high,FALSE,yes
rainy,cool,normal,FALSE,yes
rainy,cool,normal,TRUE,no
overcast,cool,normal,TRUE,yes
sunny,mild,high,FALSE,no
sunny,cool,normal,FALSE,yes
rainy,mild,normal,FALSE,yes
sunny,mild,normal,TRUE,yes
overcast,mild,high,TRUE,yes
overcast,hot,normal,FALSE,yes
rainy,mild,high,TRUE,no
```

[표 1] WEKA의 ARFF 포맷

[그림1]의 actual 항목은 실제 데이터셋의 값이고 predicted 항목은 WEKA 에서 예측한 결과이다.

06:20:45 - rules.DecisionTable

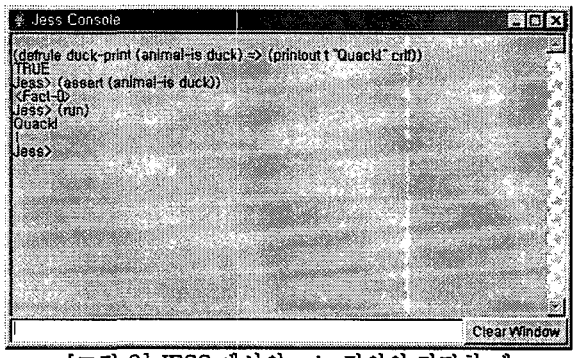
```
=== Predictions on training set ===

inst#,     actual, predicted, error, probability distribution
    1       2:no     2:no          0     *1
    2       2:no     2:no          0     *1
    3       1:yes    1:yes         *1      0
    4       1:yes    1:yes         *0.5    0.5
    5       1:yes    1:yes         *0.667  0.333
    6       2:no     1:yes       + *0.667  0.333
    7       1:yes    1:yes         *1      0
    8       2:no     2:no          0     *1
    9       1:yes    1:yes         *1      0
   10       1:yes    1:yes         *0.667  0.333
   11       1:yes    1:yes         *1      0
   12       1:yes    1:yes         *1      0
   13       1:yes    1:yes         *1      0
   14       2:no     1:yes       + *0.5    0.5
```

[그림 1] 학습결과를 이용하여 결과를 예측한 모습

2.3 JESS

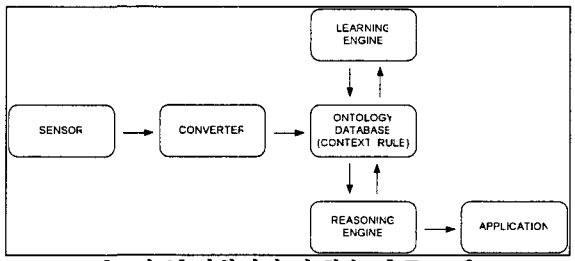
JESS[4]는 Rete 알고리즘을 사용하는 JAVA기반의 Rule 엔진이다. 자체적으로 지원하는 Rule 언어나 XML[5]을 이용하여 Rule을 정의하고 추론한다. 추론을 위해 JESS는 3가지 기본 요소를 제공한다. 첫째, 메모리에 적재될 데이터들인' fact list and instance-list' . 둘째, rule의 모음인 ' knowledge-base' . 셋째, rule의 실행을 제어하는 ' inference engine' 이다. JESS는 콘솔형태로 단독으로 실행될 수 있으며 JAVA관련 API를 이용하여 Java 코드 안에서도 사용이 가능하고 반대로 JESS 코드 안에서 JAVA를 사용할 수도 있다. 즉, 규칙을 바탕으로 추론을 할 수 있는 JAVA 프로그램의 작성이 가능하다. JESS는 버전7부터 Eclipse 기반의 개발환경에서 편집기, 디버깅 , Rete Network View 기능을 지원한다.



[그림 2] JESS 에서의 rule 정의의 간단한 예

3. Context-aware and Reasoning Model

본 논문이 제시하는 상황인지 및 학습, 추론모델의 구조는 [그림3]과 같다. 상황인지를 위한 센서와 학습엔진, 추론엔진, 그리고 센서가 인지하는 정보를 적절히 변환하는 컨버터와 변환된 정보들을 저장하는 온톨로지 데이터베이스로 구성된다.



[그림 3] 상황인지 및 학습, 추론 모델

Sensor

상황인지(Context-aware)란 디바이스 단위, 혹은 센서 단위의 개체가 주변 상황의 정보를 자동적으로 인지하는 것이다. 그 대상은 주로 온도, 습도, 시간, 사용자의 위치 좌표와 같이 외부에서 바로 가져올 수 있는 수치 등이 있다.

Converter & Ontology Database

센서의 상황인지를 통해 얻은 데이터들은 보다 효율적인 사용을 위하여 적절한 변환을 통해 온톨로지 컨텐츠화 된 후, 온톨로지 데이터베이스에 저장될 수 있다. 이 때, 데이터는 크게 두 가지 목적으로 나뉠 수 있다. 첫째는 학습을 위한 데이터인 경우로써, 데이터는 컨버터를 통과하며 같은 종류의 Context끼리 분류되어 온톨로지 데이터베이스에 축적된 후, 학습엔진의 자료가 된다. 둘째는 추론을 위한 데이터인 경우로써, 컨버터에 의해 적절히 변환된 후, 추론엔진에 의해 분석된다. 이러한 데이터들 외에 온톨로지 데이터베이스는 학습엔진이 만들어내는 Rule 을 저장하여 추론엔진에 제공한다. 그 밖에도 시스템의 목적에 따라 여러 용도로 사용될 수 있다.

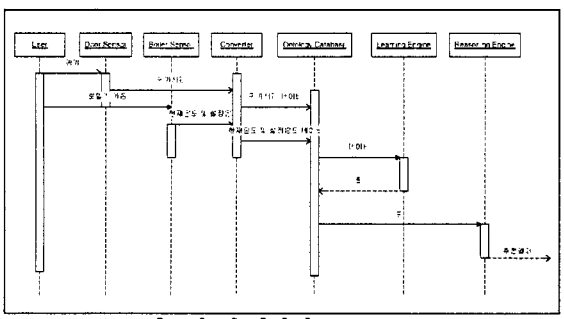
Learning Engine

온톨로지 데이터베이스에 일정량 이상의 데이터들이 축적되었을 때, 학습엔진을 이용해 Rule 을 만들어낼 수 있다. Rule 들은 추론엔진 등에 사용되기 위해 다시 온톨로지 데이터베이스로 보내어진다.

Reasoning Engine

추론엔진은 학습엔진에 의해 생성된 Rule 을 이용하여 센서로부터 인지되어오는 데이터나 기존의 데이터에 대한 결과를 추론할 수 있다. 그 결과는 시스템의 행동에 영향을 미치게 된다.

Adoptable scenario



[그림 4] 시나리오 흐름도

위에서 제안한 상황인지 및 학습, 추론모델을 간단한 스마트 홈 시스템에 적용하여 설명한다. 먼저, 처음 설치된 스마트 홈 시스템은 거주자의 정보를 파악하기 위해 센서를 통해 들어오는 정보들을 온톨로지 데이터베이스에 저장한다. 이 정보들은 거주자의 출입시간, 실내 온도에 따른 보일러 작동여부 등이 될 수 있다. 즉, 출입문의 센서는 며칠간 거주자의 출입시간을 감지하고 보일러의 온도센서는 실내온도에 따른 설정온도를 감지하여 데이터베이스에 기록한다. 축적되는 이 정보들은 주기적으로 학습엔진에 의해 처리되고, 그 결과 해당 정보들에 대한 Rule 이 생성된다. 이제 거주자는 스마트 홈 시스템을 가동시킬 수 있다. 스마트 홈 시스템은 거주자의 패턴을 분석한 Rule 을 바탕으로 거주자의 귀가시간을 예상하고 예상시간의 실내온도에 따라 보일러의 동작여부와 그 적정온도까지 예상하여 설정해 놓을 수 있다.

4. Conclusion and Future works

상황인지, 학습, 추론의 요소는 유비쿼터스 시스템에서의 핵심적 요소이다. 본 논문에서는 이들이 일련의 과정으로 만들어낼 수 있는 기본적이고 포괄적인 유비쿼터스 시스템 모델을 제안하였다. 그리고 이를 기초로 다양한 유비쿼터스 시스템 모델을 구상할 수 있을 것으로 기대한다. 하지만 이는 기본적이고 넓은 의미의 프레임워크 수준의 제안이므로 실제 구현을 위해서는 보다 구체적인 후속연구가 필요하다. 특히, 각 요소간의 통신을 위한 프로토콜과 데이터 변환과정, 온톨로지 데이터베이스의 효율적인 활용 등에 대해서 보다 심층적인 연구가 필요할 것이다.

후 기

본 연구는 문화관광부 및 한국문화콘텐츠진흥원의 지역문화산업연구센터(CRC)지원사업의 연구결과로 수행되었음

참 고 문 헌

[1] RDF Primer W3C Recommendation 10 February 2004, http://www.w3.org/TR/2004/REC-rdf-primer-20040210/

[2] OWL Web Ontology Language Overview W3C Recommendation 10 February 2004, http://www.w3.org/TR/2004/REC-xml-20040204/

[3] WEKA, http://www.cs.waikato.ac.nz/ml/weka/

[4] JESS, http://herzberg.ca.sandia.gov/jess/

[5] Extensible Markup Language (XML) 1.0 (Third Edition) W3C Recommendation 04 February 2004, http://www.w3.org/TR/2004/REC-xml-20040204/