유비쿼터스 환경을 위한 온톨로지기반 상황인지 시스템

권순현○ 박영택
숭실대학교 컴퓨터학과
kwonshzzang○@naver.com, park@computing.ssu.ac.kr

Ontology-Based Context Aware System for Ubiquitous Environment

SunHyon Kwon○ Youngtack Park
Dept.of Computing, Soongsil University

요 약

유비쿼터스 컴퓨팅이란 사용자에게 지속적인 서비스를 제공해주는 컴퓨팅 환경을 말한다. 끊임없이 동적으로 변하는 유비쿼터스 환경에서 수많은 상황데이터가 발생을 하고 상황정보로 추상화하는 과정이 필수적이다. 상황인지시스템은 동적인 상황정보에 대한 생성, 조작, 공유 등이 일관성 있게 이루어져야 한다. 이러한 상황정보의 조작을 위한 수많은 상황인지 모델이 제시되고 연구되어 왔다.
본 논문에서는 유비쿼터스 환경을 위한 온톨로지 기반 상황인지 시스템을 제시한다. 상황정보에 대한 생성, 컨텍스트 추론, 지식의 공유을 위해 온톨로지 표준언어인 OWL을 사용한 컨텍스트 온톨로지를 생성한다. 디바이스의 상황정보 생성을 위해 SWRL 규칙언어를 사용하고 생성된 디바이스 상황에 고장진단 및 수리서비스를 제공하기 위해 규칙추론기반 언어인 Jess를 사용하고 OWL기반의 컨텍스트 온톨로지와의 연계를 위해 Jess Tab API를 사용한다.

1. 서 론

무선 네트워크와 모바일 디바이스의 출현은 오늘날 컴퓨팅 방향을 디바이스와 서비스의 끊임없는 협동을 통해 사용자의 일을 처리해주는 유비쿼터스 컴퓨팅이라는 새로운 분야로 움직이게 했다. 유비쿼터스 컴퓨팅 기술의 출현은 사용자 측면에서 서비스를 수행하는 객체로서 디바이스와 어플리케이션을 분리함으로써 “언제나, 어디서나” 지속적으로 사용자에게 서비스를 제공한다[1]. 이러한 것을 통해 사용자는 그들의 일에만 관심을 가질 수 있게 되었고 어플리케이션은 동적환경에 대해 동작을 할 수 있어야만 한다. 유비쿼터스 환경에서 디바이스와 서비스와 에이전트는 그들의 상황을 인지하고 있어야만 하고 자동적으로 변화하는 상황에 대해 적응을 하여야 한다. 이를 상황인지라고 한다[2]. 이는 상황에 따라 사람, 장소, 물리적 컴퓨팅 디바이스 각각이 처해있는 환경을 특징화하기 위해 사용된다[3]. 이렇듯 상황인지 컴퓨팅의 본질은 상황정보를 수집하고 조직하고 처리하고 배포하는 것이다. 현재 상황정보를 수집하고 표현하고 사용하는 수많은 방법들이 개발되었고 많은 프로젝트에서 적용하여 프로젝트를 착수하였다[4].
상황인지 컴퓨팅에서는 상황정보를 접근하고 관리하고 표현하는 모델이 필요하다[2]. 이 상황인지 모델은 어플리케이션기반 접근방법, 모델기반 접근방법, 온톨로지기반 접근방법의 3가지가 있다. 본 논문에서는 온톨로지기반 접근방법을 사용한다.
온톨로지는 관심 있는 도메인에 대한 지식을 표현하기 위해 사용한다. 온톨로지는 도메인의 개념과 개념사이의 관계를 표현한다[8]. 최근 표준 온톨로지언어로서 W3C(World Wide Web Consortium)의 표준 온톨로지 언어 OWL(Ontology Web Language)을 사용한다[8]. OWL은 표준 규칙언어로서 SWRL(Semantic Web Rule Language)을 사용한다. SWRL은 OWL의 개념에 의해 표현된 규칙을 사용할 수 있는 기능을 제공한다[7].
유비쿼터스 컴퓨팅환경에서 특정한 시간주기로 센서에 의해 감지된 수많은 데이터를 컨텍스트 온톨로지에 인스턴스화함으로써 감지하다 지식을 확장시키고 SWRL 추론규칙을 통해 컨텍스트 정보를 생성하고 규칙 추론엔진인 Jess(Java Expert System Shell)을 이용 디바이스에 대한 고장진단, 수리를 하여 정보, 지식으로 확대하고자 하는 것이다.
본 논문에서는 온톨로지기반 상황인지 시스템에 대한 연구를 하고자 한다. 이 시스템은 디바이스에 대한 상황정보를 각각의 디바이스에 부착된 센서를 통해 감지한다. 이 때 감지된 데이터는 컨텍스트 온톨로지에 인스턴스화하고 SWRL기반 추론을 통해 상황정보를 생성한다. 생성된 정보는 규칙기반 추론엔진을 통해 각 디바이스의 현재 상태를 진단하고 진단된 정보를 사용자에게 알리거나 적합한 수리서비스를 제공한다.
센서데이터의 효과적인 관리 및 정보, 지식으로 확대, OWL기반의 컨텍스트 온톨로지 구축, SWRL을 통한 센서데이터 추상화, 디바이스에 대한 진단 및 수리에 적용할 추론규칙 설계의 세부적인 연구에 대해 논할 것이고 이를 검증하기 위한 시스템을 구축할 것이다.
본 논문의 구성은 다음과 같이 2장에서는 본 논문에서 제시한 시스템의 시나리오에 대해 설명하고자 한다. 각 세부적인 시나리오를 통해 최종 목표하는 시스템이 무엇인지 설명하고자 한다. 이는 실제 6장의 시스템 테스트 부분에서 시뮬레이터를 통해 검증할 것이다. 3장에서는 온톨로지기반 지능형 상황인지 에이전트 시스템의 구조를 살펴볼 것이다. 전체적인 시스템 구조에 대해 설명하고 세부적인 각 시스템의 기능에 대해 설명하

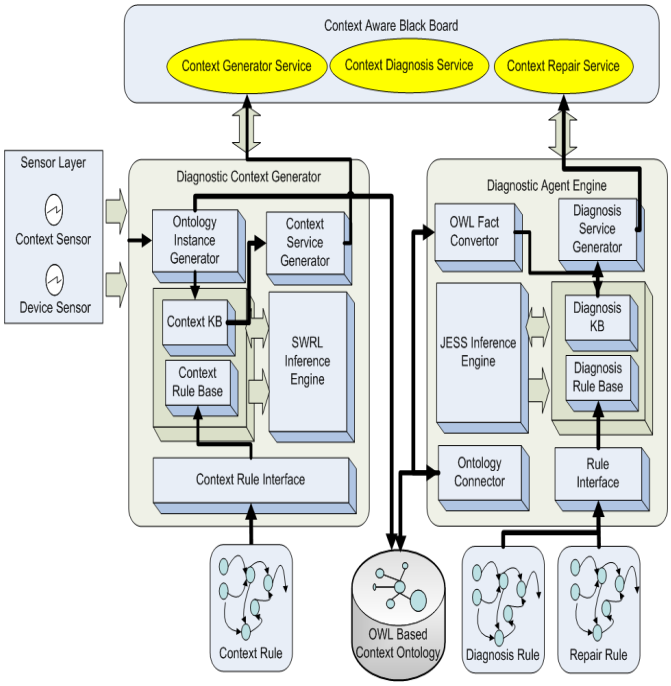
고자 한다. 4장에서는 온톨로지 표준언어인 OWL기반 컨텍스트 온톨로지의 구조에 대해 설명하고자 한다. 해당 온톨로지의 표준언어인 OWL기반 컨텍스트 온톨로지 구조에 대해 설명하고자 한다. 해당 온톨로지의 클래스 구조를 설명함으로써 각 센서들의 정보가 온톨로지에 어떻게 저장이 되는지 설명하고자 한다. 5장에서는 SWRL을 이용하여 센서로부터 수집된 데이터가 어떻게 추상화가 되는지에 대한 상세한 설명을 할 것이고 온톨로지 추론엔진의 진단 및 수리에 대한 추론규칙에 대해 설명하고자 한다. 추상화된 정보를 이용하여 디바이스에 대한 진단과 수리에 대한 추론이 어떻게 이루어지는지에 대한 상세한 설명을 하고자 한다. 6장에서는 3장의 시나리오를 바탕으로 실제 시뮬레이터의 테스트를 진행할 것이고 테스트의 결과에 대한 설명을 하고자 한다. 마지막으로 7장에서는 본 논문의 결과와 차후 구현된 시스템의 확대방향에 대해 논할 것이다.

2. 온톨로지기반 지능형 상황인지 에이전트 시나리오

"2007년 7월10일 오전10시 온도센서와 습도 센서에서 실내온도가 31도이고 실내습도가 50%라고 감지되었다. 에어컨 디바이스 센서는 현재 에어컨이 동작되지 않는다고 감지되었고 이는 현재 여름이고 온도가 22도 이상이면 에어컨이 동작해야 된다는 진단규칙에 따라 에어컨에 문제가 발생하였다는 사실을 추론한다. 에어컨 문제라는 진단결과에 따라 에어컨의 세부 센서들의 감지하다 체크해 본 결과 에어컨 필터 교체시기가 90일 이상이 되었다는 감지하다 확인한다. 에어컨 필터 교체 시기는 80일이고 반드시 교체를 해야 된다는 진단규칙에 따라 현재 에어컨의 필터 교체주기가 지났다는 결과가 진단되었고 이 진단정보는 에어컨 서비스센터에 전송됨으로써 자동으로 수리서비스를 받게 한다." 위의 시나리오를 보면 시스템은 시간, 온도, 습도등의 감지하다 각 디바이스 현재상태의 감지하다 통해 에어컨의 세부 디바이스에 대한 진단을 내린다.
시나리오를 하나 더 살펴보면 "2007년 6월 20일 밤 8시 30분 온도센서와 습도센서에서 실내온도가 15도이고 실내습도가 36%라는 사실과 조명 디바이스 센서에서 현재 조명이 동작하지 않는다는 사실이 감지된다. 여름이고 시간이 20시 30분 이후이면 조명이 자동으로 진단해야 한다는 진단규칙에 따라 현재 조명문제가 발생하였다는 사실을 추론한다. 조명에 문제가 발생하였다는 진단결과로부터 조명의 세부센서를 체크하고 체크결과 조명의 휴즈센서에 문제가 있다는 사실을 확인한다. 조명의 휴즈에 대한 진단정보는 조명서비스센터에 자동으로 전송된다."
위의 두 시나리오를 통해 사용자의 개입 없이 센서들의 센싱데이터와 진단규칙 및 수리규칙을 통해 디바이스의 고장을 진단하고 수리하는 것이 이 시스템의 목표이다.

3. 온톨로지기반 지능형 상황인지 에이전트 구조

유비쿼터스 환경에서 온톨로지기반 지능형 상황인지 에이전트 시스템은 센서로부터 데이터를 수집하고 컨텍스트 정보를 생성하는 진단형 컨텍스트 생성기(Diagnostic Context Generator)와 생성된 컨텍스트 정보를 통해 디바이스의 고장에 대한 진단과 진단된 결과에 대한 수리서비스를 제공하는 진단 에이전트 엔진(Diagnostic Agent Engine)으로 구성된다.
[그림3-1]에서 보는 바와 같이 컨텍스트 생성기는 온톨로지 인스턴스 생성기와 SWRL 컨텍스트 규칙엔진으로 구성된다. 센서로부터 감지된 데이터는 온톨로지 인스턴스 생성기를 통해 OWL기반의 온톨로지의 인스턴스로 저장된다. 온톨로지의 인스턴스로 저장된 데이터는 SWRL 컨텍스트 추론엔진을 통해 추상화된 컨텍스트 정보가 생성된다.
진단 에이전트 엔진은 컨텍스트 생성기를 통해 생성된 컨텍스트 정보를 온톨로지 연결 API(Ontology Connect API)를 통해 컨텍스트 온톨로지의 추론엔진의 FACT로 가지고 온다. 진단 컨텍스트 엔진의 규칙기반에 등록된 진단, 수리 추론규칙을 통해 진단과 수리에 대한 추론을 진행한다. 진단결과는 컨텍스트 온톨로지 API를 통해 다시 컨텍스트 온톨로지의 인스턴스로 저장이 된다. 이때 컨텍스트 온톨로지에는 센서로부터 감지된 원천데이터, 추상화된 컨텍스트 정보, 진단된 지식, 수리된 지식에 대한 인스턴스를 가지고 있다. 이는 향후 똑같은 상황에서 센싱데이터가 인스턴스화 되면 진단, 수리에 대한 추론을 진행하지 않고 바로 해당되는 추상화된 컨텍스트와 진단지식, 수리지식에 대한 서비스의 결과를 도출할 수 있다. 이는 시스템의 프로세스 향상에 도움이 될 수 있다.



[그림 3-1] 온톨로지기반 지능형상황인지 에이전트구조

3-1. 컨텍스트 생성기(Diagnostic Context Generator)

컨텍스트 생성기는 센서로부터 센싱된 데이터를 바탕으로 컨텍스트 정보를 생성하는 모듈이다. 이 때 감지된 데이터와 생성된 컨텍스트 정보는 OWL기반의 컨텍스트 온톨로지에 인스턴스화 된다. 컨텍스트 생성기는 온톨로지 인스턴스 생성기와 SWRL 컨텍스트 추론엔진으로 구성된다[7]. 온톨로지 인스턴스 생성기는 감지된 데이터의 유효성 및 중복을 검사하게 된다. 유효성 검사를 마친 센싱데이터는 컨텍스트 온톨로지의 인스턴스로 저장하게 된다. 온톨로지 인스턴스 생성기는 부가적으로 센서의 센싱 주기 및 시점을 셋팅하여 주기적으로 센서가 센싱을 하도록 기능을 제공한다.
SWRL 컨텍스트 추론엔진은 컨텍스트 온톨로지에 인스턴스화된 센서데이터를 바탕으로 실제 추상화된 컨텍스트 정보를 생성하는 모듈이다. 예를 들어 센서로 "현재 날짜가 2007년 7월이고 실내온도가 28도이다." 라는 센서데이터를 바탕으로 "현재의 계절은 여름이고 실내온도가 22도 이상이므로 기온이 덥다." 하여 "에어컨이 동작할 필요가 있다." 라는 추상회된 컨텍스트 정보를 생성한다. 이는 향후 진단 및 수리에 대한 추론규칙의 FACT로서 사용된다. 현재 SWRL 컨텍스트 추론엔진을 바탕으로 센서데이터를 실제 진단 및 추론에 사용할 수 있는

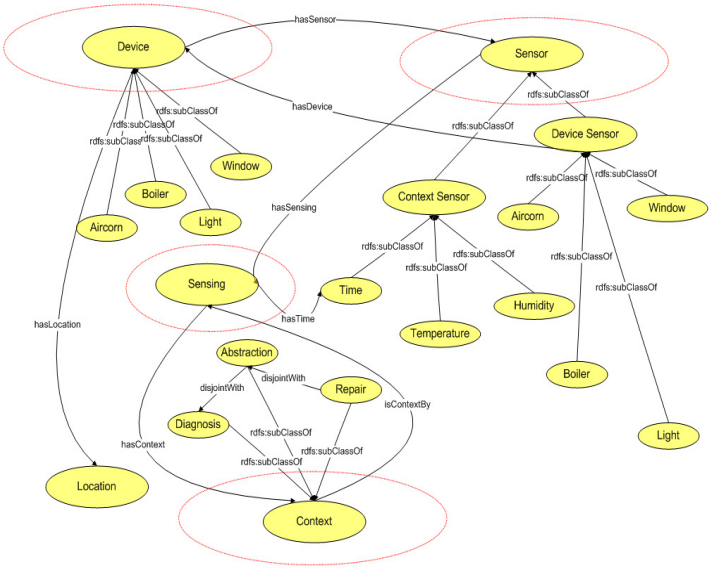
컨텍스트 정보 형식으로 진행한다.

3-2. 진단 에이전트 엔진(Diagnostic Agent Engine)

진단 에이전트 엔진은 컨텍스트 생성기를 통해 생성된 컨텍스트 정보를 바탕으로 추론엔진의 진단규칙 및 수리규칙을 통해 디바이스의 진단 및 수리에 대한 추론을 진행한다. 본 시스템에서는 자바 기반 추론언어인 JESS(Java Expert System Shell)을 이용하여 추론엔진을 구성하였고 이러한 Jess와 온톨로지와의 연동을 위해 Jess Tab을 사용하였다[6].
컨텍스트 정보는 JESS의 FACT로서 사용되고 진단규칙 및 수리규칙을 사용하여 추론을 진행한다. 예를 들어 컨텍스트 생성기를 통해 생성된 컨텍스트 정보가 "현재 에어컨을 동작할 상황이다." 라는 정보와 센서로부터 감지한 데이터인 "현재 에어컨이 동작하지 않는다." 라는 FACT를 이용하여 "현재 에어컨 문제가 발생하였다." 라는 진단을 내리게 된다. " 이 진단결과를 바탕으로 각 디바이스의 세부센서로부터 감지된 데이터를 체크한다. 예를 들어 "에어컨의 필터교체 시기가 100일 되었다." 라는 센서의 감지하다 "에어컨의 필터교체 주기는 80일이다." 라는 진단규칙으로부터 "에어컨에 문제가 발생하였고 필터교체주기가 지났다." 라는 세부 진단을 내리게 된다. 또한 에어컨의 필터교체주기 초과에 대한 진단결과로부터 "에어컨 부품서비스 센터에 필터 교환에 대한 정보를 전송한다." 라는 수리에 대한 서비스를 자동으로 전송한다.
이상의 진단과 수리에 대한 새로운 지식은 컨텍스트 온톨로지의 인스턴스로 저장된다.

3-3. 진단형 컨텍스트 온톨로지(Diagnostic Context Ontology)

온톨로지는 관심 있는 도메인에 대한 지식을 표현하기 위해 사용된다. 온톨로지는 도메인상의 개념 및 개념들 사이의 관계를 표현한다[5]. 본 시스템에서는 OWL기반의 온톨로지를 사용한다. 디바이스의 정보와 센서정보를 백그라운드 지식으로 활용하고 센서로부터 감지된 데이터와 SWRL 컨텍스트 추론엔진을 통해 생성된 컨텍스트 정보와 온톨로지기반 추론엔진을 통해 진단지식과 수리지식을 저장하기 위해 컨텍스트 온톨로지를 설계 및 구현한다[6][7].
[그림 3-2]은 진단형 컨텍스트 온톨로지의 구조이다. 그림에서 컨텍스트 온톨로는 디바이스의 백그라운 정보로서 활용되는 Device 클래스와 온톨로지 컨텍스트 정보와 Device에 대한 센서를 표현하는 Sensor 클래스와 감지하다 데이터가 저장되는 Sensing 클래스와 생성된 컨텍스트와 추론된 지식을 저장하는 Context 클래스로 나눌 수 있다. 각각의 클래스들은 해당 정보의 계층구조를 표현하기 위한 서브클래스들과 프로퍼티들로 연결이 되어있다.



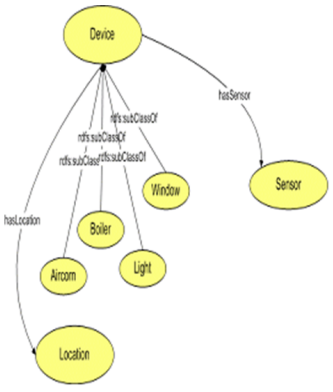
[그림3-2] OWL기반 컨텍스트 온톨로지 구조도

4. 진단형 컨텍스트 온톨로지

온톨로지기반 지능형 상황인지 에이전트 시스템은 감지하다 결과와 생성된 컨텍스트, 진단지식, 수리지식을 컨텍스트의 온톨로지 인스턴스화하여 활용한다. 본 시스템은 감지된 데이터를 온톨로지로 통해 확장하고 컨텍스트의 생성, 진단. 수리에 대한 추론을 진행하게 해준다. 본 시스템은 온톨로지 표현언어로서 OWL을 사용한다. OWL은 온톨로지 표현언어로서 W3C(World Wide Web Consortium)의 표준 온톨로지 언어이다. 개념과 개념간의 관계를 통해 관심 있는 도메인에 대한 지식표현을 위해 사용한다[8]. 상세한 컨텍스트 온톨로지의 내역을 살펴보면 다음과 같다.

4-1. Device 클래스

Device 클래스는 실제 디바이스의 정보가 저장되는 클래스이다. 디바이스의 서브클래스로서 에어컨, 보일러, 조명, 창문클래스를 가진다. 디바이스는 해당되는 디바이스센서를 가진다. 또한 디바이스는 해당되는 위치를 가지고 이는 Location클래스를 통해 가지고 올 수 있다. [그림 4-1]은 Device 클래스의 상세한 구조와 실제 OWL로 표현된 그림이다.

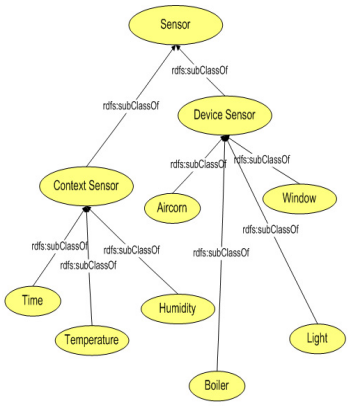



```
<defconcept name="Device"/>
<defconcept name="Aircorn"/>
<impliesc>
  <catom name="Aircorn"/>
  <catom name="Device"/>
</impliesc>
<defconcept name="Boiler"/>
<impliesc>
  <catom name="Boiler"/>
  <catom name="Device"/>
</impliesc>
<defconcept name="Light"/>
<impliesc>
  <catom name="Light"/>
  <catom name="Device"/>
</impliesc>
<defconcept name="Window"/>
<impliesc>
  <catom name="Window"/>
  <catom name="Device"/>
</impliesc>
<defconcept name="Location"/>
<defconcept name="Sensor"/>
```

[그림 4-1] Device 클래스의 구조와 OWL표현

4-2. Sensor 클래스

각 디바이스에 부착된 세부적인 센서정보가 저장되는 클래스이다. Sensor 클래스는 서버 클래스로서 Context Sensor와 Device Sensor로 구성된다. Context Sensor는 온도, 습도, 시간 등의 정보를 감지하는 센서로 구성되고 Device Sensor는 디바이스별 감지하다 저장하는 클래스이다. 센서는 특정시점의 감지하다 데이터를 가지고 있다. [그림 4-2]는 Sensor클래스와 OWL로 표현된 그림이다.

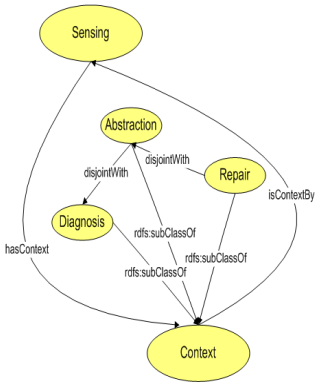



```
<defconcept name="Sensor"/>
<defconcept name="Context_Sensor"/>
<impliesc>
  <catom name="Context_Sensor"/>
  <catom name="Sensor"/>
</impliesc>
<defconcept name="Device_Sensor"/>
<impliesc>
  <catom name="Device_Sensor"/>
  <catom name="Sensor"/>
</impliesc>
<defconcept name="Time"/>
<impliesc>
  <catom name="Time"/>
  <catom name="Context_Sensor"/>
</impliesc>
<defconcept name="Temperature"/>
<impliesc>
  <catom name="Temperature"/>
  <catom name="Context_Sensor"/>
</impliesc>
<defconcept name="Humidity"/>
<impliesc>
  <catom name="Humidity"/>
  <catom name="Context_Sensor"/>
</impliesc>
```

[그림 4-2] Sensor 클래스의 구조와 OWL표현

4-3. Context 클래스

Context 클래스는 감지하다 데이터를 SWRL 추론엔진을 통해 생성된 컨텍스트 정보를 저장하는 Abstraction 클래스, 추론엔진을 통해 생성된 지식을 저장하는 Diagnosis 클래스, Repair 클래스를 서브 클래스로 가진다. Context 클래스는 특정한 시점의 센싱된 결과를 가지고 있고 SWRL Rule 엔진을 통해 생성된 컨텍스트 정보와 Jess 엔진을 통해 추론된 진단, 수리에 대한 결과를 가지고 있다. [그림 4-3]은 Context 클래스의 구조와 OWL로 표현된 그림이다.




```
<defconcept name="Sensing"/>
<defconcept name="Context"/>
<defconcept name="Abstraction"/>
<impliesc>
  <catom name="Abstraction"/>
  <catom name="Context"/>
</impliesc>
<disjoint>
  <catom name="Abstraction"/>
  <catom name="Diagnosis"/>
</disjoint>
<defrole name="hasContext"/>
<domain>
  <ratom name="hasContext"/>
  <catom name="Context"/>
</domain>
<range>
  <ratom name="hasContext"/>
  <or>
    <catom name="Context"/>
  </or>
</range>
<equalr>
  <ratom name="hasContext"/>
  <inverse>
    <ratom name="isContextBy"/>
  </inverse>
</equalr>
```

[그림 4-3] Context 클래스의 구조와 OWL표현

5. 온톨로지기반 지능형 상황인지 에이전트 구현

5-1. SWRL을 통한 컨텍스트 정보생성

컨텍스트 온톨로지는 Sensing 클래스에 저장된 인스턴스를 이용하여 SWRL 추론엔진을 통해 센싱된 데이터를 추상화한다. 이 추상화된 정보는 Abstraction Class에 인스턴스로 저장이 되고 규칙 추론엔진인 Jess 시스템의 새로운 Fact로 저장이 되어 이후 진단규칙 및 수리규칙의 추론에 사용이 된다.

```
hasDevice(?y, ?z) ∧ swrlb:equal(?z, "Bolier") ∧
abstractTemperature(?x, ?arg1) ∧ swrlb:equal(?arg1,
"cold ") ∧ dataBoolean(?x, ?arg1) ∧
 swrlb:notEqual(?arg1, true)
→ abstract(?x, "Bolier Problem")
```

[표 5-1] 보일러고장에 대한 컨텍스트 정보생성 규칙

[표 5-1]은 보일러에 대한 감지하다 통해 보일러 고장의 컨텍스트 정보를 생성하는 SWRL 추론규칙이다. “2007년 1월에 실내 온도는 5도 이하이다.”라는 감지하다 “현재 겨울이고 날씨가 춥다.”라는 컨텍스트 정보와 “보일러가 동작할 필요가 있다.” 라는 컨텍스트 정보를 생성한다. 실제 보일러 센서로부터 “현재 보일러가 동작하지 않는다.“ 라는 센서정보를 통해 보일러가 동작해야 될 컨텍스트 상황에 실제 보일러가 동작하지 않으므로 ”보일러 문제가 발생하였다.“ 라는 컨텍스트 정보를 생성할 수 있다.

5-2. 규칙 추론엔진

본 시스템에서는 규칙기반 추론엔진인 Jess(Java Expert System Shell)을 이용하여 추론엔진을 구성하였고 이러한 Jess와 온톨로지와의 연결을 위해 Jess Tab을 사용하였다[6]. SWRL 규칙엔진에 의해 생성된 컨텍스트 정보를 JessTab을 이용하여 온톨로지와 연계하고 규칙기반 추론엔진의 Fact로서 사용함으로써 추론엔진의 진단규칙, 수리규칙을 구동시키는 것이다. 이때 온톨로지의 인스턴스는 추론엔진에 Fact로서 자동으로 사용되고 이미 구축된 진단규칙과 수리규칙을 통해 추론을

진행하게 된다. [표 5-2]은 에어컨의 진단규칙의 예를 보여준다.

```
(defrule diagnosis::diagnosis_not_power_aircorn
    "에어컨 Problem이고 전원이 OFF이면  전원불량 "
    (declare (auto-focus TRUE))
    (diagnosis_fact (device  aircorn)
    (diagnosis_ident  aircorn) (value problem))
    (sensoring_input (device power) (sensor_ident
     powerOnOff) (value off))
     =>
    (assert (diagnosis_fact (device power)
    (diagnosis_ident powerOnOff) (value off)
    (desc "에어컨에  Power가 제공되지 않습니다.")))
 )
```

[표 5-2] 에어컨의 고장진단 규칙

센서에서 감지하다 데이터와 SWRL 규칙을 이용하여 생성된 컨텍스트 정보를 통해 디바이스에 대한 진단을 내리는 규칙이다. 위의 예에서 현재 디바이스의 감지하다 컨텍스트 정보를 바탕으로 에어컨 문제가 발생하였고 전원이 제공되지 않고 있다는 진단결과를 내리는 규칙이다.

5-3. 진단 및 수리규칙의 수행

진담모듈은 추상화 단계에서 SWRL을 이용하여 생성된 컨텍스트 정보와 센서로부터 감지하다 데이터를 이용하여 디바이스의 고장진단 서비스를 제공해주는 규칙이다. 예를 들어 "여름이고 실내온도가 22도 이상이라서 에어컨을 동작시켰음에도 불구하고 실내온도가 22도 아래도 내려가지 않는 경우 윈도우 디바이스의 문제가 있음을 알려준다.

```
(defrule diagnosis::diagnosis_open_window_aircorn
    "에어컨이 동작하지만 온도가 떨어지지 않고 창문이
      열려있음. "
    (declare (auto-focus TRUE))
    (abstract_fact (device  aircorn) (abstract_ident
     operation) (value need_operation))
    (sensoring_input (device aircorn) (sensor_ident
     operation) (value on))
    (sensoring_input (device window) (sensor_ident
     OpenClose) (value open))
    (sensoring_input (device  temperature)
    (sensor_ident  degree) (value ?t&:(>= (integer ?t)
     21) ))
     =>
    (assert (diagnosis_fact (device  window)
    (diagnosis_ident OpenClose) (value open) (desc "
     창문이 열려 있습니다.  ")))
)
```

[표 5-3] 에어컨의 진단규칙

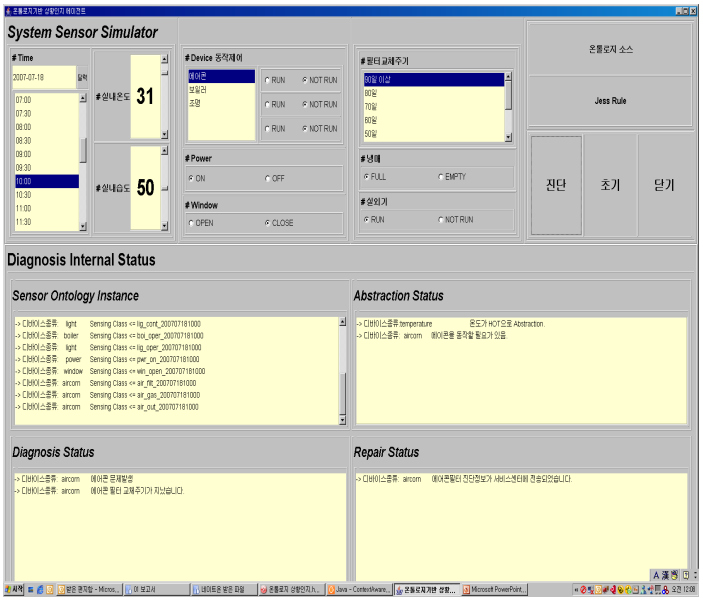
수리규칙은 진단규칙에서 진단한 디바이스 문제와 각 디바이스의 세부사항의 문제점을 진단하여 사용자에게 전달해주는 규칙이다. 예를 들어 에어컨이 동작하지 않아서 에어컨 문제점을 확인하니 에어컨 필터의 교체주기가 90일 이상 되어서 문제가 발생하였다면 에어컨의 필터를 교체해야 한다는 정보를 사용자에게 전달해주거나 또는 이 정보를 서비스센터로 직접 전달할 수 있다.

```
(defrule repair::repair_not_filter_aircorn
    "에어컨 필터 교체주기시 Repair  "
    (declare (auto-focus TRUE))
    (diagnosis_fact (device aircorn) (diagnosis_ident
     aircornFilter) (value need_exchange_filter))
     =>
      (assert (repair_fact (device  aircorn)
      (repair_ident Filter) (value need_exchange_filter)
      (desc "에어컨 필터 진단정보가 서비스센터에 전송
      되었습니다.  )))
)
```

[표 5-4] 에어컨에 대한 수리규칙

6. 시스템 테스트

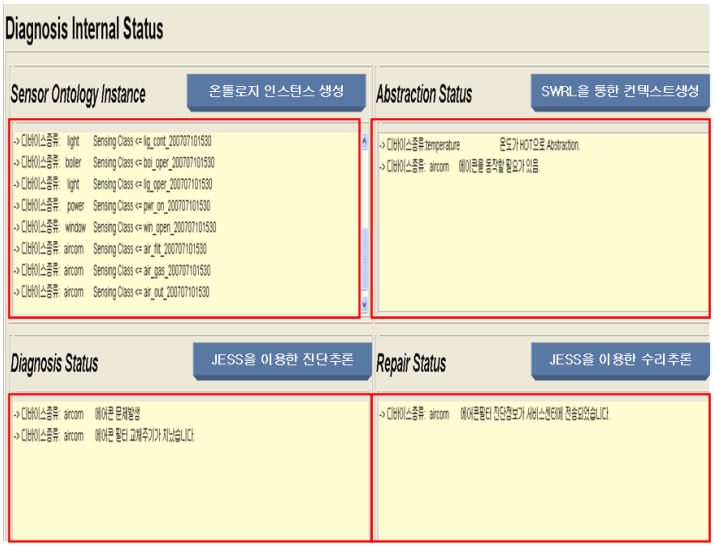
[그림6-1]은 2장의 시나리오에 대한 시뮬레이터를 수행한 화면이다. 시뮬레이터는 "시간이 2007년 7월18일 오전 10시이고 실내온도가 31도이고 실내습도가 50%이다." 라는 감지하다 감지하였다. 시간 센싱데이터 로부터 "현재 여름이다."라는 추상화된 컨텍스트 정보를 생성하였고 "여름이고 실내온도가 22도 이상이다." 라는 감지하다 "현재 온도가 높다." 라는 컨텍스트 정보를 생성할 수 있다.
지금까지의 생성된 컨텍스트 정보를 통해 "에어컨이 동작할 필요가 있다." 라는 최종 컨텍스트 정보를 생성할 수 있다. 지금까지의 컨텍스트 생성과정은 SWRL 규칙을 통해 이루어진다. 에어컨 디바이스의 감지하다 현재 에어컨이 동작하지 않는다는 사실을 알 수 있고 "에어컨이 동작할 필요가 있는데 실제 에어컨이 동작하지 않는다." 라는 진단규칙으로부터 "현재 에어컨에 문제가 발생하였다." 라는 진단결과를 도출할 수 있다.



[그림 6-1] 시나리오에 대한 시뮬레이터 메인화면

에어컨의 문제가 발생하였으므로 에어컨의 세부센서로부터 감지하다 데이터를 체크한다. 실제 "에어컨의 필터 교체주기는 80일 이하이어야 한다." 라는 진단규칙을 통해 현재 센서의 감지하다 에어컨 필터교체주기가 90일이라고 감지되었으므로 "에어컨 필터교체주기가 지났다." 라는 진단결과를 도출한다.

이에 대한 수리규칙을 통해 에어컨 필터 고장에 대한 진단결과는 “서비스센터에 진단결과가 전송되었습니다.”라는 수리결과를 시뮬레이터는 화면에 표시한다.
이처럼 시뮬레이터에서는 각 센서로부터 감지한 데이터로부터 현재 에어컨의 필터교체주기가 지났고 이 진단정보가 서비스센터에 자동으로 전송되었다는 결론을 추론해 낸다. 이는 유비쿼터스 환경에서 자동으로 각 디바이스의 결과를 진단함으로써 사용자의 개입 없이 지속적인 서비스를 제공해주는 것이다.



[그림 6-2] 시나리오에 대한 시뮬레이터 수행결과

[그림 6-2]는 시뮬레이터에 대한 수행결과 화면이다. 시뮬레이터의 수행결과는 컨텍스트 온톨로지의 인스턴스 생성화면, SWRL을 이용한 컨텍스트 생성결과, JESS를 이용한 고장에 대한 진단추론 결과, 진단결과에 대한 수리추론 결과화면이다. 시뮬레이터에서 센서의 센싱값은 컨텍스트 온톨로지의 클래스의 인스턴스로 저장된다. SWRL을 통해 센싱된 결과는 컨텍스트로를 생성한다. [그림 6-2]의 예에서 “온도가 HOT이고 에어컨이 동작할 필요가 있다.”라는 컨텍스트 정보가 생성되었다. 이 컨텍스트 정보는 “에어컨에 문제가 발생했고 에어컨 필터교체주기가 지났다.” 라는 진단결과가 추론되었다.“ 진단결과에 대한 수리추론은 ”에어컨의 필터정보가 서비스센터에 전송되었다.“라는 수리추론결과가 생성되었다. 이는 JESS의 추론결과이며 이는 다시 컨텍스트 온톨로지의 인스턴스로 저장된다.

7. 결론 및 향후 연구

온톨로지기반 지능형 상황인지 에이전트는 시스템을 통해 유비쿼터스 환경에서 에어컨, 보일러, 조명등의 몇가지 디바이스에 대한 센싱데이터와 시간, 온도, 습도 등의 Context 센싱데이터를 통해 디바이스의 상태를 진단을 해보았다. 이 시스템은 센서정보를 온톨로지 인스턴스로 저장함으로써 백그라운드 지식으로 확대하고 SWRL Rule을 통해 컨텍스트정보를 생성하여 Jess Rule을 이용하여 디바이스의 고장에 대해 진단과 수리를 진행한다.
현재 이 시스템은 체크할 디바이스에 대한 세부상태를 더욱더 세밀하게 진단할 센서를 확보하는 것이 필요하고 이는 해당 디바이스에 대한 전문가 지식이 필요하고 할 수 있다. 그리고 동일 디바이스에 대해 여러대가 있을때의 상황에 대한 고찰이 필요하다.
컨텍스트 온톨로지를 각 디바이스의 서비스센터와 연결하여 진단된 결과를 자동으로 전송함으로써 사용자가 인식하지 못한 상황에서 지능적으로 디바이스에 대한 서비스를 받을 수 있는 시스템으로 기능을 확대할 필요도 있다.
이러한 전문가의 지식을 가미한 Rule의 확대. 재생산과 타 시스템의 연계를 통해 시스템을 Upgrade할 필요가 있다고 할 수 있다. 하지만 온톨로지기반 지능형 상황인지 에이전트 시스템은 기본적인 디바이스의 고장상태를 체크하여 자동으로 진단하고 기본적 진단결과에 대한 수리를 해 줄 수 있다는 측면에서 유비쿼터스 환경의 하나의 설례를 남긴 것이라고 자평하는 바이다.

참 고 문 헌

[1] Henricksen K, Indulska J, Rakotonirainy, "Infrastructure for Pervasive Computing : Challenges", Workshop on Pervasive Computing INFORMATIK 01, Viena, September 2001.
[2] Tao Gu, Xiao Hang Wang, Hung Keng Pung, Da Qing Zhang, "An Ontology-based Context Model in Intelligent Environments", Department of Computer Science, National University of Singapore, Singapore
[3] Dey, A. and Abowd, G., "Towards a Better Understanding of Context and Context-Awareness", Workshop on the what, who, where, when and how of context-awareness at CHI 2000, April 2000.
[4] M. Strimpakou, et al., "Context Modelling and Management in Ambient-aware Pervasive Environments", Int Workshop on Location and Context-Aware(LoCA 2005)
[5] S. M. Chantzara, I. Sygkouna, S. Vrotis, I. Roussaki, and M. Anagnostou, "Context Management for the Provision of Adaptive Service to Roaming Users", IEEE Wireless Comm, 11(2) 2004, pp40-47
[6] Henrik Eriksson : JessTab Manual, integration and Protege and Jess, Linkoping University, 2004
[7] Martin O'Connor, Holger Knublauch : Writing Rules for Semantic Web Using SWRL and Jess, Standford University School of Medicine 2004.
[8] Matthew Horrideg, Holger Knublauch, Alan Rector, Robert Stevens : A Practical Guide To Building OWL Ontologies Using The Proteget-OWL Pluging and CO-ODE Tools, The University of Manchester, 2004
[9] Maria A. Strimpakou, Ioanna G. Roussaki, Miltiades E. Anagnostou : A Context Ontology for Pervasive Provision, National Technical University of Athens, Greece, 2004
[10] Sajjad Ahmad Madani, Mihaela Ulieru : An Application of Industrial Agents to Concrete Bridge Monitoring, Instutute of Computer Technology, Canada Research Chair.