다중 추론기반 상황인식 서버의 구현

정장섭*, 기병욱**, 방대욱***
계명대학교 컴퓨터공학과
e-mail:hdca64@hanafos.com*, kiquddnr@gmail.com**, dubang@kmu.ac.kr***

An Implementation of Context Aware Server Based on Multi-Reasoning

Jang-Seop Jeong, Byung-Wook Ki Dae-Wook Bang
Dept of Computer Engineering, Keimyung University

요 약

최근 웹상에 존재하는 데이터와 정보들을 사람뿐만 아니라 기계의 컴퓨터 프로그램이 이해할 수 있도록 시멘틱 추론을 기반으로 표현하는 방법은 날로 발전하고 있다. 그렇지만 상황인식 시스템을 구성하는 요소들의 모호성과 복잡성 해결을 위해 다양한 연구가 진행되고 있음에도 불구하고, 기존의 방법들은 주로 규칙기반 추론방법만 적용하여 구현함으로써 정보의 신뢰성과 정확성 확보에 있어 불합리한 요소와 추론의 한계가 많은 것이 현실이다. 따라서 본 논문에서는 단일 추론의 한계를 보완하기 위하여 온톨로지 추론, 규칙 추론 또는 확률 추론을 상황에 따라 선택할 수 있는 다중 추론기반 상황인식 서버를 제안하고, OWL 온톨로지 및 자바기술, JADE 이동 에이전트 그리고 Protégé-OWL API로 구현하는 과정과 성능분석 결과를 설명한다.

1. 서론

최근 노트북, PDA, 스마트 폰과 같은 모바일 장치가 출현되고 영향력을 가짐으로써 유비쿼터스 컴퓨팅이 점점 대중화 되어가고 있다. 유비쿼터스 컴퓨팅은 새로운 기술이 아니고 기존 네트워크 기술을 사용해서 사용자가 언제 어디서 어떤 방법을 사용하든지 정보를 받고 사용할 수 있게 한다. 유비쿼터스 컴퓨팅의 목표는 컴퓨터가 주위 환경 변화를 알고 변화에 따라 사용자가 필요한 것을 결정하여 컴퓨터가 자연스럽게 환경에 적응하는 것이다.

유비쿼터스 컴퓨팅의 한 분야로 상황인식 시스템이 있다. 상황인식 시스템이란 외부 사용자의 간섭 없이 현재 상황에 스스로 자신의 오퍼레이션을 적응시킬 수 있고, 주변 환경의 상황을 고려함으로써 실용성과 유효성을 증가시키는 것을 목적으로 하고 있다[1].

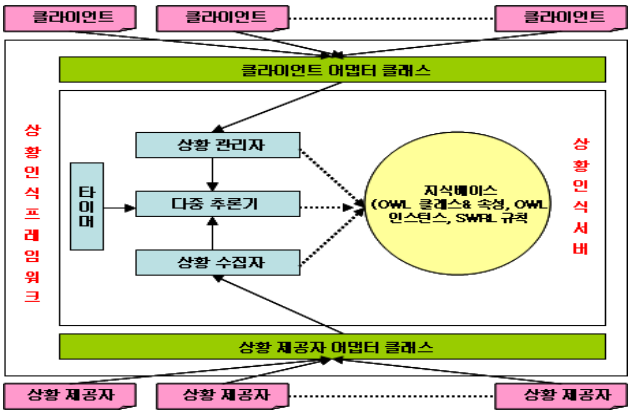
기존의 상황인식 서버구현 사례들은 주로 규칙기반 추론 방법에만 중점을 두고 있다 본 논문에서는 단일 추론으로 인한 추론 한계를 보완하기 위하여 온톨로지 추론, 규칙 추론 또는 확률 추론을 상황에 따라 선택할 수 있는 다중 추론기반 상황인식 서버를 제안하고, OWL 온톨로지 및 자바기술, JADE 이동 에이전트 그리고 Protégé-OWL API로 구현하는 과정과 성능분석 결과를 설명한다.

본 논문의 구성은 다음과 같다. 2절은 다중 추론기반 상황인식 서버 구조와 세부 모듈들의 기능을 기술하고, 3절에서는 상황인식 서버의 구현에 대해 기술한다. 4절은 구현사례 및 실험분석에 대해서, 5절에서는 결론 및 고찰에 관하여 기술한다.

2. 다중 추론기반 상황인식 서버의 구조

2.1 기반구조

다중 추론기반 상황인식 서버의 기반구조는 (그림 1)과 같이 클라이언트 어댑터, 상황인식 서버, 상황 제공자 어댑터로 구성된 상황인식 프레임워크와 클라이언트, 그리고 상황 제공자로 구성된다.



(그림 1) 다중 추론기반 상황인식 기반구조

상황인식 프레임워크는 클라이언트를 대신하여 상황정보를 수집, 관리, 그리고 전달하는 기능을 수행한다.

상황 제공자는 여러 센서들로 상황 데이터를 수집한다. 센서로는 물리 센서, 가상 센서, 논리 센서 등이 있다. 상황 제공자는 상황인식 프레임워크의 상황 제공자 어댑터

를 사용하여 상황서버와 통신한다.

클라이언트는 사용자가 발생한 이벤트를 처리하고 그 결과를 보여준다. 클라이언트는 상황인식 프레임워크의 클라이언트 어댑터를 사용하여 상황인식 서버와 통신한다.

2.2 상황인식 서버

상황인식 서버는 (그림 1)과 같이 내부에 OWL 클래스 및 속성, OWL 인스턴스, SWRL 규칙, 베이지안 네트워크로 구성된 지식베이스를 두고, 상황 관리자와 상황 수집자에 의해 상황 제공자와 클라이언트에 서비스를 제공한다. 상황인식 서버는 상황에 따라 선택한 추론기로 하위 상황정보를 추론하여 상위 상황정보를 생성하는 다중 추론기를 가동한다. 다중 추론기는 상황 수집자가 수집한 원시 상황정보를 저장하고 상위 상황정보의 생성이 필요한 경우, 상황관리자가 클라이언트로부터 요청받은 쿼리를 추론해야 하는 경우, 타이머로부터 주기적인 추론을 요청받는 경우 등의 상황에서 실행되어 추론을 한다.

(1) 상황인식 서버의 기능

상황 관리자는 외부로부터 검색 요청을 받아 지식베이스를 검색하고 그 결과를 외부로 전송해 준다. 검색요구는 SPARQL로 표현된 쿼리이거나 지식베이스의 OWL Individual을 직접 요구한 표현이다. 상황 관리자는 검색 요청을 처리하기 전에 상위 상황정보를 추론하는 다중 추론기를 가동할 수 있다.

상황 수집자는 상황 제공자가 전달하는 원시 상황정보를 지식베이스에 저장한다. 이후에 상황 수집자는 상위 상황정보를 추론하는 다중 추론기를 가동할 수 있다.

(2) 다중 추론기의 기능

① 다중 추론기 정의

다중 추론기는 온톨로지 추론기와 규칙기반 추론기, 베이지안 네트워크 추론기로 구성된다. 온톨로지 추론은 OWL 온톨로지로 표현된 상황정보에 암시적으로 내포하고 있는 규칙을 추론하고[2], 규칙기반 추론은 SWRL[3]로 표현된 규칙을 추론하며, 베이지안 네트워크 추론[4]은 확률에 의한 추론을 가능하게 한다.

② 추론기 구동시점

다중 추론기는 구동시점에 호출되어 적절한 추론기를 선택하고 추론기에 최적의 데이터를 제공하며 추론결과를 지식에 반영한다. 추론 시점은 크게 네 가지로 나눌 수 있다.

첫 번째로 상황 관리자가 클라이언트의 쿼리 요청을 처리하면서 추론이 필요한 경우이다. 주로 추론을 필요로 하는 구문 즉 SQWRL로 작성된 쿼리 처리를 하는 경우이다.

두 번째는 상황 제공자가 센서로부터 수집한 원시 상황정보를 지식베이스에 저장하고 지식베이스의 상위 상황정보 갱신이 필요한 경우이다.

세 번째는 타이머 설정으로 주기적인 추론을 설정한 경우이다. 이 경우는 시간 흐름에 따른 상황변화를 반영할 수 있게 한다.

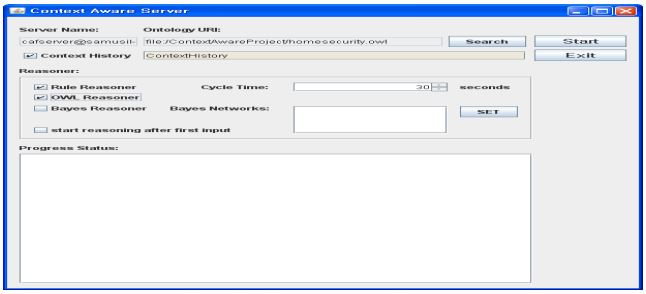
마지막으로 상황인식 서버가 최초 구동되면서 주어진 OWL 온톨로지로 지식베이스를 구축하고 초기 상위 상황정보를 추론하는 경우이다.

③ 추론기 선택

다중 추론기는 구동시점에 주어진 상황을 적절히 표현하는 상위 상황정보를 추론하기 위해 온톨로지 추론기, 규칙 추론기 또는 베이지안 추론기 중에서 하나 이상을 선택한다.

상황 관리자가 클라이언트의 쿼리 요청을 처리하는 경우 중에서 쿼리가 확률을 포함할 경우만 자동으로 베이지안 추론기를 선택한다.

그 외에는 온톨로지 추론기와 규칙 추론기를 선택하는데, 두 가지 모두 선택할 수도 있고 한 가지만 선택할 수도 있다. 이 선택은 주어진 OWL 온톨로지 정의에 따르므로 수동으로 지정한다. 즉 (그림 2)와 같이 상황인식 서버의 UI 화면에서 사용자가 설정한다.



(그림 2) 상황인식 서버의 추론기 선택 UI 화면

2.3 어댑터 클래스

(1) 상황제공자 어댑터 클래스

상황 제공자 어댑터 클래스는 상황 제공자와 상황인식 서버를 연결하는 클래스이다. 상황 제공자 어댑터 클래스는 상황 제공자가 감지된 센서 정보를 상황인식 서버에 전달하는 API가 메소드로 표현된 클래스이다. 일반적으로 API는 보통 인스턴스를 생성하여 사용하는 클래스들로 구성하지만 본 연구는 상속 가능한 어댑터 클래스로 구성하였다. 따라서 상황 제공자 클래스는 상황 제공자 어댑터 클래스를 상속받아 구현한다.

(2) 클라이언트 응용 어댑터 클래스

클라이언트 어댑터 클래스는 클라이언트와 상황인식 서버를 연결하는 클래스이다. 클라이언트 어댑터 클래스는 클라이언트가 상황인식 서버에 쿼리, 검색, 이벤트 등록 등을 전달하고, 결과 또는 이벤트 발생을 되돌려 받아 클라이언트에게 전달하는 API가 메소드로 표현된 어댑터 클

래스이다. 따라서 클라이언트는 클라이언트 어댑터 클래스를 상속받아 구현한다.

3. 상황인식 서버의 구현

제안한 다중 추론기반 상황인식 서버를 자바환경에서 구현하기 위해 상황정보 스키마와 기본 상황정보를 OWL 온톨로지로 구성하고 OWL과 함께 상황규칙을 SWRL로 표현하였으며, 규칙기반 추론엔진, Jade 이동 에이전트 기술, Jess 규칙 추론엔진, 베이지안 네크워크, Protégé-OWL API을 사용하였다.

3.1 Jade로 서버 구조 구현

상황인식 서버는 어댑터 클래스를 통해 클라이언트 또는 상황 제공자로부터 요청이 있으면 작업을 수행하는 요구 구동형 구조로 구현되어야 한다.

Jade 기술의 에이전트는 스스로 수행할 수 있으며 메시지를 기다리며 대기를 하다가 메시지가 도착하면 메시지에 따라 태스크를 수행할 수 있는 기능이 있다. 따라서 상황인식 서버를 에이전트로 구현하고 내부 상황 관리자, 다중 추론기, 상황 수집자들을 태스크로 코딩하면 비교적 수월하게 구현할 수 있다.

3.2 Protégé-OWL API 사용

상황인식 서버는 지식베이스를 OWL 온톨로지의 내부 표현으로 관리한다. 그러므로 상황인식 서버를 구성하는 코드가 지식베이스를 접근하려면 이를 지원하는 API가 필요하다. Protégé-OWL API는 현재 표준은 아니지만 OWL 온톨로지 조작을 완벽하게 지원하며, SWRL 규칙과 관련 정보를 Jess 엔진에 보내고 다시 추론된 결과를 받아 지식베이스에 저장하는 과정과 OWL 온톨로지 정보의 변화를 감지하는 이벤트 발생시키므로 OWL 온톨로지 기반 지식베이스 관리에 아주 유용하다.

3.3 외부 규칙 추론기 사용

지식베이스에 있는 SWRL 규칙으로 상황정보를 추론하려면 현재는 외부 추론기를 사용하여야 한다. 외부 추론기로 Jess 엔진을 사용하려면 OWL Individual은 Jess 지식으로 표현되어야 한다. 특히 각 Individual이 소속된 Class와 Individual이 소유한 Property들이 표현되어야 한다. 또한 각 Individual의 same-as와 different-from 정보도 포함되어야 한다. 그리고 상황이력에 있는 동적 상황정보의 Individual들 중 추론에 사용되는 것도 포함되어야 한다.

Jess 템플릿 기능은 OWL Class 계층을 표현하는 메카니즘을 제공한다. Jess 템플릿 계층은 계층에 속하는 Individual의 이름을 보관하기 위해 Jess 슬롯을 사용하여 Class 계층을 모델화하는데 사용될 수 있다. 예를들면, owl:Thing class를 표현하기 위해 Jess 템플릿을 다음과 같이 정의한다.

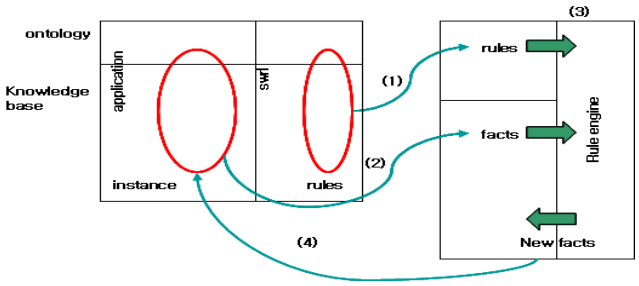
OWL 지식베이스에 있는 SWRL 규칙을 Jess로 표현하는 것은 간단하다. 그럼 다음 SWRL 규칙을 살펴보자

Computer(?x) ^ usedDuration(?x,?y) ^
swrlb:greatThanOrEqual(?y,1825)
-> useStatus(?x,"expire")

이 규칙은 위에서 정리된 Individual의 표현을 사용하여 Jess에 다음과 같이 표현될 수 있다.

(defrule aRule (Computer (name ?x)))
(usedDuration ?x ?y)(swrlb:greaterThanOrEqual(?y, 1825)) => (assert useStatus(?x, "expired"))

(그림 3)은 상황인식 서버가 Jess 규칙 엔진에 SWRL 규칙을 Jess 규칙으로 변환하여 전달하고 추론 결과를 상황인식 서버로 전달해주는 과정을 설명하고 있다.



(그림 3) Jess 규칙 추론기 사용

4. 실험 분석

본 절은 제안한 다중 추론기반 상황인식 서버를 자바환경에서 구현하고, 실험 클라이언트 애플리케이션의 실행을 통한 실험결과로 성능을 분석한다.

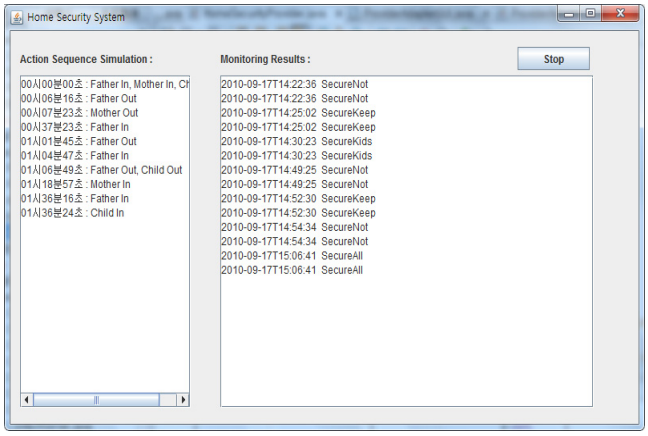
4.1 실험 애플리케이션

실험 애플리케이션은 상황인식 프레임워크를 기반으로 가족 구성원 중에서 현재 어떤 구성원이 집에 남아 있는지를 판단하여 적합한 안전보안 서비스(예, 가스 벨브 통제, 보일러 통제, 출입문 통제 등)를 실행해 준다. 서비스를 받는 대상은 가족 구성원으로 어른과 어린아이이다. 집에 있는 현재 구성원이 누구인가에 따라 각기 다른 서비스가 실행된다. 구성원에서의 어린아이는 가정 내의 장치에 대한 작동이 서투르거나 오작동을 할 가능성이 높은 10세 미만의 유아로 정의한다. 또한 외출은 2가지로 구분하여 단기외출과 장기외출로 정의한다. 단기외출은 잠시 가까운 곳(예, 집 근처의 문구점 등)을 방문하기 위한 외출로써 안전보안을 크게 신경 쓰지 않아도 되는 외출인 반면에, 장기외출은 안전보안에 만전을 기해야 하는 외출이다.

안전보안 서비스는 안전과 관련된 것들로 출입문의 잠금장치, 가스 제어기, 보일러 제어기이다. 가스 제어기와 보일러 제어기는 두 가지의 서비스를 제공한다. 가스 제어기는 잠금장치의 Open/Close와 조작을 제어하는

Lock/Unlock의 서비스를 제공한다. 보일러 제어기는 보일러의 On/Off와 조작을 제어하는 Lock/Unlock의 서비스를 제공한다.

이 애플리케이션을 위한 지식베이스는 OWL 온톨로지로 파일로 주어지는 클래스, 프로퍼티, SWRL 규칙, Individual로 정의되어 있으며, 가족 구성원의 외출 상황은 상황 제공자를 통해 실시간으로 서버에 입력된다. (그림 4)는 구현한 실험 애플리케이션의 UI 화면이다.



(그림 4) 사례 애플리케이션 실행화면

4.2 실험 과정 및 분석 결과

본 실험에서 사용한 상황 유형 및 유형별 서비스 시나리오는 <표 1>, <표 2>와 같다.

<표 1> 상황 유형별 상황 및 서비스

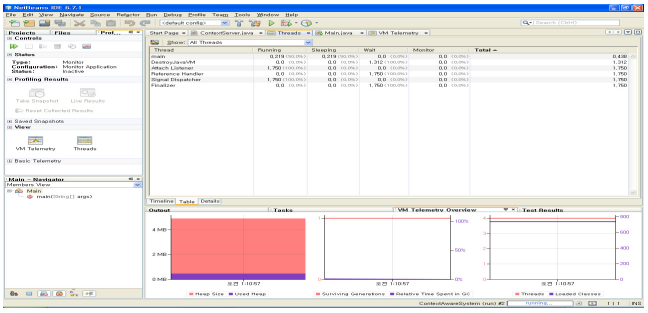
상황 유형	집에 남은 가족 구성원	서비스
SecureNot	어른, 아이	시나리오1
SecureKeep	어른단기외출, 아이	시나리오2
Secure All	어른 장기외출, 아이장기 외출	시나리오3
Secure Kids	어른 단기·장기외출, 아이	시나리오4

<표 2> 시나리오별 제공 안전보안 서비스

안전보안 장치	시나리오1	시나리오2	시나리오3	시나리오4
출입문 잠금장치	Unlock	Lock	Lock	Lock
가스밸브 제어기	Open	이전상태 유지	Close	Close
가스조작 제어기	Unlock	이전상태 유지	Lock	Lock
난방작동 제어기	계절에따라	계절에 따라	계절에 따라	계절에 따라
난방조작 제어기	Unlock	이전상태 유지	Lock	Lock

실험에 사용된 구성원 외출시간 시퀀스는 (그림 4)의 왼쪽 리스트에 주어진 시퀀스이다. 그리고 집에 남아 있는 구성원에 따른 상황유형은 외출 시점에 행해진 추론의 결과로 도출된 (그림 4)의 오른쪽 리스트이다.

실험 분석을 위해 상황인식 서버가 동작 중인 상태에서 실험 애플리케이션을 실행하고 넷빈즈 IDE의 프로파일링을 통해 구현된 코드의 메소드 실행시간을 추적하였다. (그림 5)는 프로파일링 결과를 분석 그래프로 나타내었다.



(그림 5) 서버 실행시간 그래프-1

결과 분석에 의하면 상황인식 서버에서 소요되는 시간의 대부분은 추론시간이었다. 그 이유는 본 연구가 규칙 추론기로 외부 Jess 엔진을 사용한 결과 데이터 변환에 많은 시간을 소요하였다. 그리고 온톨로지 추론기를 선택하면 소요시간이 많이 증가하는데, 역시 외부 엔진 사용이 그 원인이었다.

5. 결론

본 논문에서는 단일 추론의 한계를 보완하기 위하여 온톨로지 추론, 규칙 추론 또는 확률 추론을 상황에 따라 선택할 수 있는 다중 추론기반 상황인식 서버를 제안하여 문제점을 해결하고자 하였다. 실험에 의하면 현재 상황인식 서버의 핵심인 추론기의 구현이 외부 엔진에 의존한 결과 서버의 성능에 문제가 있었다. 향후 현재는 지원되지 않으나 외부 엔진이 아닌 자체 엔진을 지원하는 OWL API로 구현하여 실용성을 높여야 할 것이다.

참고문헌

[1] M. Baldauf. S. Dustdar and F. Rosenberg, "A survey on context-aware systems", *International Journal of Ad Hoc and Ubiquitous Computing,* pp. 263-264, 2007.

[2] Wang X.H., Gu T., Zhang D.Q., Pung H.K., "Ontology Based Context Modeling and Reasoning using OWL", *Workshop on Context Modeling and Reasoning*(CoMoRea 2004), Orlando, Florida USA, March 2004.

[3] 방대욱, "상황인식 애플리케이션을 위한 상황정보 구성 및 추론에 관한 연구", *Bulletin of the Institute for Industrial Science,* ISSN 1225-4649, Vol. 32-1, 2009.

[4] Gu T., Pung K., Zhang D. Q., "Bayesian Approach for Dealing with Uncertain Contexts", *Pervasive 2004,* Vienna, Austria, April 2004.

[5] W3C, RDF Vocabulary Description Language 1.0:RDF Schema, Working Dranft, 23, January 2003.

[6] Gruber, T., "A Translation Approach to Portable Ontologies", Knowledge Acquisition, Vol. 5, No. 2, pp.199-220, 1993.