커뮤니티 컴퓨팅을 위한 미들웨어 아키텍처

지경환[a], 양정진[b]

[a]가톨릭대학교 컴퓨터공학과
경기도 부천시 원미구 역곡2동 산43-1
Tel: +82-2-2164-4678, E-mail: sshine106@catholic.ac.kr

[b]가톨릭대학교 컴퓨터공학과
경기도 부천시 원미구 역곡2동 산43-1
Tel: +82-2-2164-4377, E-mail: jungjin@catholic.ac.kr

Abstract

커뮤니티 컴퓨팅은 유비쿼터스 컴퓨팅 환경을 기반으로 다양한 요구와 이를 수용하기 위한 서비스 혹은 개체를 나열하는 것 에서부터 이들 간 일시적으로 협력하고 주어진 상황을 효과적으로 인지할 뿐만 아니라 다수의 장치가 협력 관계를 추구하는 컴퓨팅 모델이다. 본 논문은 커뮤니티 모델과 미들웨어의 아키텍처를 확립하기 위한 요구사항과 구조에 대하여 논한다.

Keywords:

커뮤니티 컴퓨팅, 유비쿼터스, 멀티 에이전트 시스템

1. 서론

본 논문은 커뮤니티 컴퓨팅 환경을 이루는 개체들이 능동적으로 상호간 협력하여 목적에 맞는 다양한 서비스를 제공하는 환경을 위한 미들웨어 의 요구사항과 아키텍처를 제시한다. 궁극적인 목표는 단순하게 서비스를 나열하는 것에서부터 일시적으로 협력하고, 주어진 상황을 효과적으로 인지할 뿐만 아니라 다수의 장치가 협력 관계를 추구하는 성공적인 커뮤니티 컴퓨팅의 모델을 확립하는 것이다.

2. 관련연구

2.1 Ontology Repository

온톨로지(Ontology)[1]는 철학의 한 갈래로 존재의 본질을 연구하는 형이상학이다. 인공지능 영역에서의 Ontology는 개체와 각 개체간의 관계성을 기술하는 언어(Language)로써 인지능력을 위한 기본 구조가 될 수 있다. Ontology는 주어, 술어, 목적어 구문의 형태(Triple)로 자원을 기술하는 기반구조인 RDF(Resource Description Framework)[2]를 바탕으로 추론능력과 강화된 표현력을 제공해 줄 수 있는 DAML+OIL(DARPA Agent Markup Language + Ontology Inference Layer)으로 Description Logic을 기술하게 된다. 이 모든 언어는 XML(eXtensible Markup Language)을 기반으로 구법(Schema)을 가지게 되며 기계가 읽을 수 있는(Machine Readable) 특성을 가지게 된다.

Ontology Repository는 Ontology에 정의된 개념과 관계성을 질의, 회수, 갱신, 보관하고 버전관리를 위한 통합 기능을 제공하는 저장소이다.

2.2 JADE

JADE (Java Agent Development Framework)[3] 는 FIPA가 지정한 지능형 에이전트의 표준을 따르는 멀티 에이전트 시스템과 응용 프로그램의 개발을 하는 소프트웨어 개발 프레임워크이다.

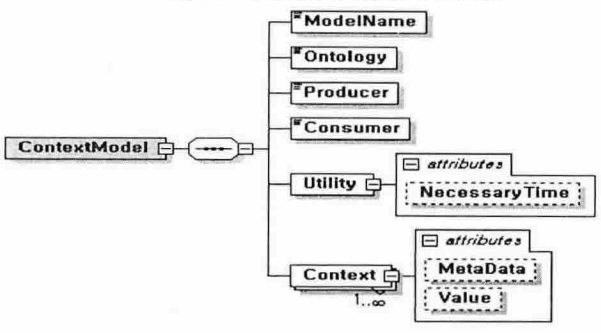
JADE는 FIPA-Compliant 에이전트 플랫폼과 자바 에이전트 개발을 위한 패키지이다. JADE는 모든 것이 자바로 구현되어 있고, 에이전트 구현을 위해 자바를 사용하여야 한다. 또한 MAS(Multi Agent System) 개발을 도와주는 다양한 디버깅 툴을 제공하고 ACL(Agent Communication Language)이 플랫폼 경계를 교차할 때 메시지는 자동적으로 FIPA 규정에 적합한 구문으로 변환되고, 코드화되며, 알맞은 프로토콜로 전달된다.

3. 커뮤니티 컴퓨팅을 위한 모델

Context는 커뮤니티 컴퓨팅 환경에서 가장 중요한 정보가 될 것이며 다양한 개체들은 많은 종류의 Context를 생성, 소비 할 것이다. 또한 다양한 개체들과 더불어 Community Model을 생성하게 될 주된 역할을 할 것이다.

3.1 Context Model

그림 1 – Context Model Schema



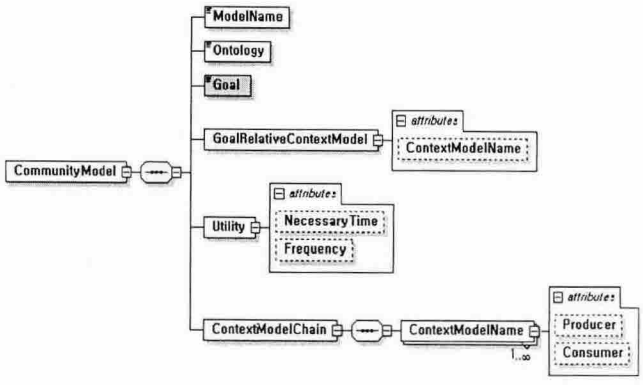
간단한 차폐형 시스템(closed system)에서 요구되는 Context의 구법(Schema)은 컴포넌트 내부에 번역 될 수 있지만 개방형 시스템(open system)에서는 다양한 Context의 구법을 주어진 상황에 따라 적응적이고 자동적으로 다룰 수 있어야 한다. 따라서 기계가 읽고 편집할 수 있으며 필요에 따라 학장 될 수 있는 선언적인 Context Model이 필요하며 이를 위해 데이터와 데이터의 의미를 기술하고 구법을 정의 할 수 있는 XML(eXtensible Markup Language)과 XML Schema[4]를 고려할 수 있다.

- Context Model은 해당 Model을 생성하는 개체와, 소비하는 개체를 기술 할 수 있다. (그림 1 참고)
- Context Model의 인스턴스는 생산자, 소비자의 값을 가지며 이 값은 Ontology Repository의 해당하는 Context Model 값과 비교되어 생산자, 소비자 리스트를 갱신 한다.
- 소비 시간은 유용성(Utility)으로서 생산, 소비된 시간적 차이를 Ontology Repository에 집계한다. (그림 1 참고)
- 특정 개체에서 사용되는 Context Model의 인스턴스는 동일한 의미의 메타데이터들을 가지므로 각 개체는 단지 해당 Markup 정보의 데이터를 회수하여 사용할 수 있다. (그림 1 참고)

3.2 Community Model

Community Model는 Agent를 기반으로 하나의 Community를 형성 하기 위하여 그림 2 와 같은 구법을 정의 한다.

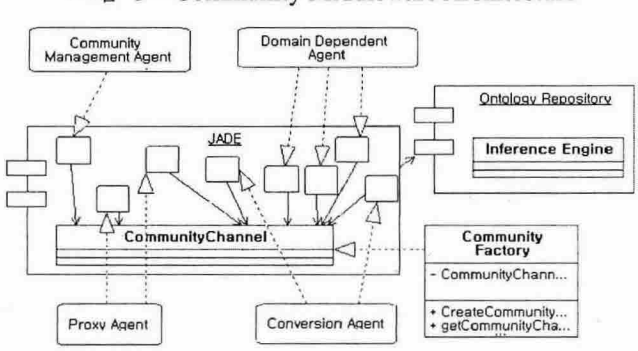
그림 2 – Community Model Schema



- Community Model은 하나의 목적(Goal)을 가지며 목적을 달성할 때의 Context를 표현하는 Context Model을 가진다. (그림 2 참고)
- Goal에 해당하는 Context Model을 생성하는 Agent와 이 Agent가 소비하는 Context Model의 체인을 가지게 된다. (그림 2 참고)
  - Context Model은 Ontology Repository에 해당 모델을 생성하는 Agent의 리스트를 가진다. 즉, 동일한 Community를 구성할 수 있는 Agent의 조합이 다를 수 있고 Agent의 활동 여부에 따라 대체 가능한 Community Model로 관리될 수 있다.
- Context Model을 생성하는 Agent가 소비하는 Context Model이 없다면 Community Model에 포함될 Agent의 선별을 완료 한다. (그림 2 참고)
- Community Model은 유용성(UTILITY)의 근거가 되는 유지 시간, 발현 주기를 가진다.

4. 커뮤니티 컴퓨팅 미들웨어 아키텍쳐

그림 3 – Community Middleware Architecture



4.1 Domain independent Agent

4.1.1 Community Management Agent

특정 목적(Goal)을 감지 하면 목적과 연관된 Context Model을 Ontology Repository 컴포넌트의 RDB(Goal, Context Model)로부터 회수 후 Context Model을 <Goal Relative Context Model>의 데이터 값으로 가지는 Community Model을 검색한다. 검색된 리스트 중 활동 가능한 Agent를 가지는 인스턴스를 바탕으로 그림 3 좌측 하단에 표현된 Community Channel을 생성하여 사용하고 Context Model의 순서에 따른 흐름을 관리하는 Agent이다.

4.1.2 Proxy Agent

Proxy Agent는 요구된 Context Model을 근거로 해당하는 수동적인 개체(Passive Entity)를 동작시키고 수동적인 개체가 생성한 데이터들을 바탕으로 Context Model과 인스턴스를 생성 하는 에이전트이다.

소비되고 생성되는 Context Model 메타데이터의 의미적 상호운용성(Semantic Interoperability)을 취하고 Context Model을 구성하는 데이터의 근원이 되는 수동적인 객체(Passive Object)의 API와 API들의 호출 순서를 위한 규칙(Rule)을 Ontology Repository로부터 회수 할 수 있어야 한다.

4.1.3 Conversion Agent

Agent가 소비하는 Context Model정보를 찾을 수 없을 때 기존의 Context Model을 바탕으로 필요한 Context Model을 생성하는 역할을 수행하며 이를 위해 여과(filter), 병합(merge), 합성(composite), 번역(translate), 산출(calculate)하는 규칙(Rule)을 바탕으로 목적 Context Model을 생성 한다.

Conversion Agent는 소비되고 생성되는 Context Model의 메타데이터의 의미적 상호운용성(Semantic Interoperability)과 각 경우에 필요한 변환 규칙(Conversion Rule)을 회수, 추론 하기 위해 Ontology Repository를 사용 할 수 있어야 한다.

4.2 Domain dependent Agent

주어진 목적을 가지는 다양한 Agent가 이에 해당되며 다른 Agent에 의해 생성된, 혹은 변환된 Context Model들을 바탕으로 설계된다. 이때 목적을 기준으로 Ontology Repository의 RDB(Goal, Context Model) 상에 해당 Context Model이 존재 하지 않을 때 새로운 설계가 필요하다.

4.3 Middleware Architecture

커뮤니티 컴퓨팅의 미들웨어 아키텍처는 그림 3과 같다. Domain (In)dependent Agent들이 JADE Platform에서 활성화 되며 Community Channel을 통한 Communication을 수행한다. Community Channel은 기존 Channel을 Community 단위로 분할하여 복잡도를 낮추어 줄 수 있다.

JADE Platform내의 Agent들은 필요에 따라 Ontology Repository에 접근하여 검색, 회수, 질의, 추론, 저장을 수행하며 적응적으로 동작한다.

5. 결론

본 논문에서는 커뮤니티 컴퓨팅을 위한 미들웨어 아키텍처를 제시하였다. 결과적으로 Context Model을 기반으로 하는 Agent의 설계를 이끌어 낼 수 있었고 Agent간의 상호작용을 통하여 능동적으로 목적을 달성하는 환경 조성을 위한 미들웨어를 제시하였다.

Context Model의 자동적이고 적응적인 변환은 제시된 미들웨어에 독립적이고 Ontology Repository에 의존적이다. 즉, 미들웨어와 Ontology Repository의 역할을 명백하게 분리 하였다. 따라서 Community Model이 생성, 소멸되고 관리 될 때의 적응력은 Ontology Repository와 연관된 추론 능력에 따라 좌우될 것이다.

6. 향후 계획

커뮤니티 컴퓨팅 미들웨어의 아키텍처를 구성하는 도중, Ontology Repository의 다양한 요구사항과 Context Model간 변환규칙의 필요성, 실험적 결과를 얻을 수 있는 구현, Agent설계를 위한 요구사항을 연구할 필요성이 있다. 궁극 적으로 커뮤니티 컴퓨팅 환경 구축을 위한 연구를 진행 할 것이다.

7. 참고문헌

[1] OWL Web Ontology Language Semantics and Abstract Syntax http://www.w3.org/TR/2004/REC-owl-semantics-20040210/

[2] RDF Primer, W3C Working Draft 23 January 2003, http://www.w3.org/TR/rdf-primer

[3] JADE (Java Agent DEvelopment Framework) (2005) http://jade.tilab.com/doc/index.html

[4] XML Schema W3C Recommendation, 28 October 2004, http://www.w3.org/TR/2004/REC-xmlschema-0-20041028/