韓國精密工學會 '97年度 秋季學術大會論文集 pp.697~701

네트워크를 이용한 에이전트 기반 설계 환경 구축
A Construction Agent-based Design Environment using a Network

안상준 (연세대 공대 대학원), 이수홍(연세대 공대)
Sang-jun An(Graduate School, Yonsei Univ.), Soo-hong Lee(Yonsei Univ.)

ABSTRACT

The network environment includes a number of autonomous agents which are widely distributed, have different platforms, and change very dynamically. The information system operated on this environment must solve some basic problems; restrictive client-server models, heterogeneous systems, and intelligent agents. We are using KQML for knowledge management. Java language provides solutions against a strong server dependency and a heterogeneous troubles. We introduce KQML as an agent communication language and JATLite as a java agent template. These increase an efficiency of communication on network and enable us to use resources distributed in world wide web. Also, we describe a new agent system architecture and implement it through an example scenario.

Key Word : KQML(지식 질의/처리 언어), JATLite, Network(네트워크), Concurrent Engineering(동시공학), Agent Technology(에이전트 기술), Distributed System(분산시스템)

1. 서 론

20세기 후반부터 컴퓨터가 급속하게 발달하기 시작하면서 공학 전반에 걸쳐 컴퓨터의 응용이 점차 증가하고 있다. 기계 공학 분야에서도 이러한 추세는 계속되어 CAD, CAM, CAPP 등으로 이어지고 동시공학(CE, Concurrent Engineering)이라는 개념을 이끌어냈다.

이러한 컴퓨터의 진보와 더불어 1990년대에 하나의 쟁점으로 등장한 것이 바로 인터넷이다. 전세계에 걸친 고속 통신망을 이용하여 인터넷은 지금까지 해결하지 못했던 공동 협력 분산 작업을 거의 동시적으로 해결할 수 있는 기반을 마련하였다. 또한 인터넷과 연계된 이러한 접근 방법은 급속하게 변화하는 제품 개발의 추세와 생산비용의 절감, 가용한 서비스의 활용, 신속한 제품 설계라는 취지와 맞물려 국가와 기업, 학계간에 하나의 큰 과제로 부상하고 있다.[1, 2, 3]

네트워크 환경은 수많은 자율적인 에이전트들을 포함하는데 이러한 에이전트들은 통신망을 통하여 널리 분산되어 있고, 서로 다른 플랫폼을 가지고 있으며, 매우 동적으로 변화한다. 이런 환경에서 작동하는 정보 시스템은 기본적인 몇 가지 문제들을 처리해야만 한다.[4]

우선 인터넷 상에서 널리 쓰여지고 있는 아키텍처인 클라이언트-서버 모델은 너무 제한적이다. 현재 인터넷 정보 서비스들은 사용자가 원하는 대로 새롭고 판단력 있는 요소를 가져주기 어렵다. 대부분의 에이전트들은 그들이 한쪽에 의존한다기 보다 서로 상호작용하기를 원한다.

또한 에이전트들의 이질적인 문제를 처리해야만 한다. 이것은 에이전트들이 서로 다른 플랫폼들, 서로 다른 데이터 포맷들, 서로 다른 정보 서비스들의 기능들, 그리고 그 서비스들에 사용되어지는 서로 다른 표준 시스템들을 가지고 있다는 것을 의미한다. 이러한 문제로 한쪽의 정보를 다른 쪽에서 사용할 수 없다면 에이전트 기술은 그 효용 가치가 상당히 떨어질 것이다.

마지막으로 에이전트들은 지능적이어야 하며 인간과 효과적으로 상호작용을 할 수 있어야 한다. 우리

가 에이전트들을 지능적이라고 이야기할 때, 에이전트들은 다음과 같은 능력을 갖추어야 한다.

- 창의적인 행동
- 대화와 타협을 통한 공동작업
- 표현적인 통신 언어들을 사용한 상호통신
- 분산되어 있는 로컬 자원들의 응용
- 인간과의 효과적인 상호작용

따라서 위와 같은 문제들을 해결하기 위해서는 적절한 에이전트 통신 언어를 선택하고, 에이전트들 간의 새로운 아키텍쳐를 모델링하여 이를 구현하는 것이 필요하다.

2. 에이전트 통신 언어로서의 KQML

지능적인 에이전트들 사이의 통신을 위해서는 공통적인 구문과 의미, 실제 사용 방법들을 공유해야 한다. 정보와 질의를 표현하는 언어로는 SQL, KIF, LOOM 등과 같은 것들이 있지만, 어느 하나 보편적으로 받아들여지는 것이 없고 표준화되기에는 이미 각각의 위치에서 확고한 자리 매김을 하고 있어 어려운 상황에 있다.

우리는 에이전트 통신 언어로써 KQML (Knowledge Query and Manipulation Language)을 선택하였다. KQML이란, 다양한 지적 시스템들 사이의 지식과 정보의 교환을 위해 설계된 통신 언어이다. KQML은 대규모의 지식베이스를 공유할 수 있고 재사용이 가능한 구축기술과 방법을 개발시키는 데 목적을 두고 있다. KQML은 일종의 메시지 포맷이지만, 동시에 에이전트간의 실시간 지식 공유를 지원한다. KQML은 지식을 공유하기 위해 지적인 시스템들과 상호작용 하는 응용 프로그램을 위한 언어로써 사용될 수 있다.[5]

KQML은 확장이 가능한 수행어(performative)들에 초점을 두고 있다. 이들을 통하여 에이전트가 다른 에이전트에게 KQML 메시지를 보냈을 때 수신 에이전트가 어떤 동작을 수행하도록 메시지가 의도되었다는 점에서 수행어라고 불린다. 또는 송신 에이전트의 지식을 다른 수신 에이전트들에게 알린다는 점에서 수행어을 정의하기도 한다. 이 수행어는 에이전트들 사이의 상호작용에 있어서 타협과 같은 높은 수준의 통신 모델을 개발하기 위한 기반을 제공한다.

KQML은 세 가지 계층으로 나뉘어 진다. 통신 계층에는 송신자, 수신자, 그들의 주소 등 통신과 관련된 여러 가지 요소들을 나열한다. 내용 계층에는 실제적인 메시지가 들어 있다. 이 부분에는 KQML, KIF, SQL, ASCII 등 표현 언어라면 어떤 것도 가능하다. 메시지 계층에는 수행어를 사용하여 메시지의 성질을 정의한다.

KQML은 메시지 요소들의 리스트를 '()'로 둘러싸고 있으며, 그 안의 리스트의 첫 번째는 수행어이고, 나머지는 키워드와 그 값의 쌍으로 이루어진다. 대부분 처음 구현되었던 것들이 Common Lisp로 이루어졌기 때문에 현재 구문은 이것과 유사하다.

KQML은 보통 KIF를 사용하여 각각의 특정 지식에 대해서 서술적으로 기술한다. KQML은 또한 객체 지향의 데이터를 전송하고, 보다 넓은 영역의 정보를 축적하는데 사용되어 진다. 일반적으로 KQML은 질문, 선언, 신뢰, 요구, 획득, 묘사, 제공 등과 같은 정보에 대한 상태를 의사 교환하는데 사용되어 진다. 이러한 에이전트 통신 언어로써 KQML의 유용성은 최근 발표된 몇몇의 논문에서 찾아볼 수 있다.[1, 4]

3. 에이전트 기반 설계 환경

우리는 자바 언어로 에이전트들을 개발함으로써 에이전트의 개념화를 이루었다. 자바 언어의 선택은 에이전트 기술의 문제점 중의 하나인 이질적인 문제를 해결하기 위해 자바가 플랫폼에 무관하다는 특성을 이용하고자 함이다. 또한 넷스케이프의 네비게이터나 마이크로소프트의 IE와 같은 웹브라우저를 이용하여 에이전트 애플릿을 통한 설계를 가능하게 하기 위해서이다. 기본적으로 TCP/IP 프로토콜을 이용하고, 에이전트 통신 언어로 KQML을 사용하여 인터넷에 분산되어 있는 다른 에이전트들과 비동기적으로 통신한다.

우리는 스탠포드 대학의 CDR에서 개발한 JATLite 자바 API를 이용하여 에이전트들과 기존의 에이전트들과의 인터페이스를 개발하고 있다.[7] JATLite는 에이전트 개발을 위한 기본적인 자바 클래스 라이브러리를 제공하고, 메시지 통신을 위한 기본적인 패널을 제공한다. 이 패널을 이용하여 사용자가 다른 에이전트들에게 메시지를 보내거나 다른 에이전트들로부터의 메시지를 읽을 수 있게 된다.

현재 JATLite는 TCP/IP와 KQML을 이용하여 구현을 하고 있지만, 또 다른 에이전트들의 통신을 위해서 통신 언어와 관련된 추상 계층과 기본 통신 계층을 제공한다. 또한, 이 두 계층을 바탕으로 하여 실제적으로 메시지를 표현하고 해석하고 처리하는

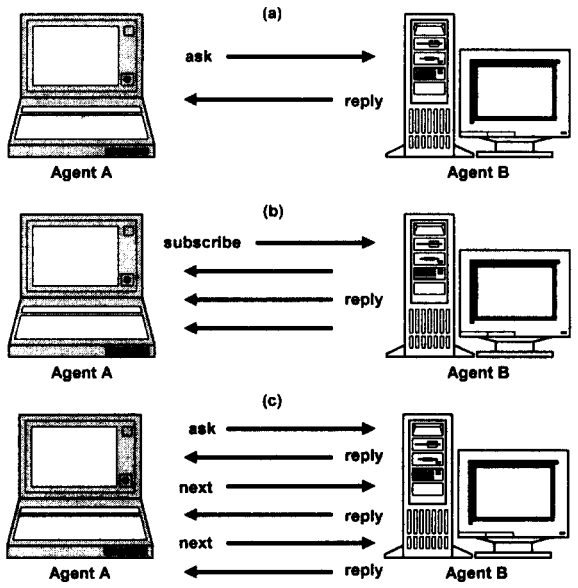

Fig. 1. 두 에이전트 사이의 메시지 교환 형태

KQML 계층과 라우터 계층, 프로토콜 계층을 제공한다.

4. 에이전트 시스템 아키텍처

두 에이전트 사이의 메시지 교환은 기본적으로 세 가지 형태로 이루어질 수 있다. 가장 단순한 형태는 묻고 대답하는 형태이다.(Fig. 1.(a) 참조) 두 번째 형태는 한 에이전트가 다른 에이전트에게 알고 싶은 내용에 대해 예약을 하는 것이다. 그러면 예약 신청을 받은 에이전트는 그 내용에 대한 지식이 생길 때마다 신청한 에이전트에게 알려주며, 신청시에 축적된 지식이 있으면 신청과 동시에 전달시킨다.(Fig. 1.(b) 참조) 마지막 형태는 첫 번째 형태의 확장이다. 한 에이전트가 다른 에이전트에게 질문을 던졌을 때, 질문을 받은 에이전트는 해결에 필요한 내용을 요청하고 그 내용을 받아 처리하다가 다시 필요한 내용을 요청하고 다시 받아서 처리한다. 이런 형태를 계속해서 반복한 후 해결한 내용을 되돌린다.(Fig. 1.(c) 참조)

우리는 이러한 기본 메시지 교환을 확장하기 위해서 퍼실러테이터(Facilitator)라는 에이전트를 사용한다. 퍼실러테이터는 기본적으로 라우터 역할을 하며, 다른 에이전트들의 이름과 주소, 역할 등을 기억하여 메시지의 전달과 중재의 기능과 ANS(Agent Name Service) 기능을 수행한다. 모든 메시지는 퍼

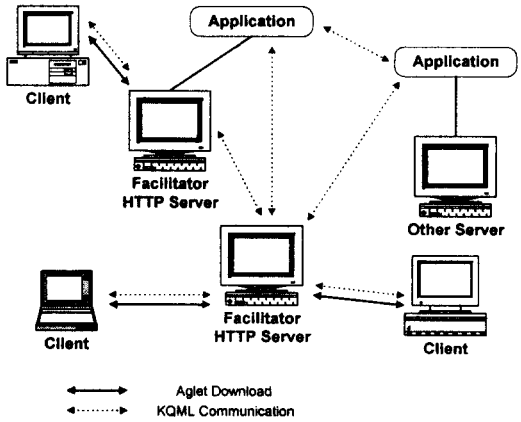

Fig. 2. 에이전트 시스템 아키텍처

실러테이터에게 전달되며, 수신 에이전트가 정해져 있으면 그 에이전트의 주소를 찾아 메시지를 전달한다. 만약 수신자가 정해지지 않은 메시지라면 해결할 수 있는 가장 적절한 에이전트를 찾아 메시지를 전달한다.

에이전트들이 이러한 기능을 수행할 수 있는 것은 에이전트마다 가상 지식 베이스(Virtual Knowledge-Base)를 갖고 있기 때문이다.(Fig. 3. 참조) 이 가상 지식 베이스는 에이전트가 실행될 때 자신의 데이터베이스로부터 시작하여, 실행 동안에 다른 에이전트들에 의한 메시지를 통하여 점점 성장해 간다.

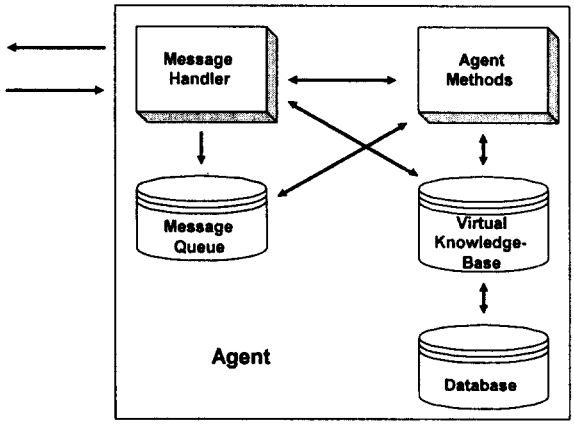

Fig. 3. 가상 지식 베이스를 가진 에이전트의 구조

이러한 지적인 에이전트들을 이용하여 다음과 같은 좀 더 고급적인 통신 모델을 구현할 수 있다. 예

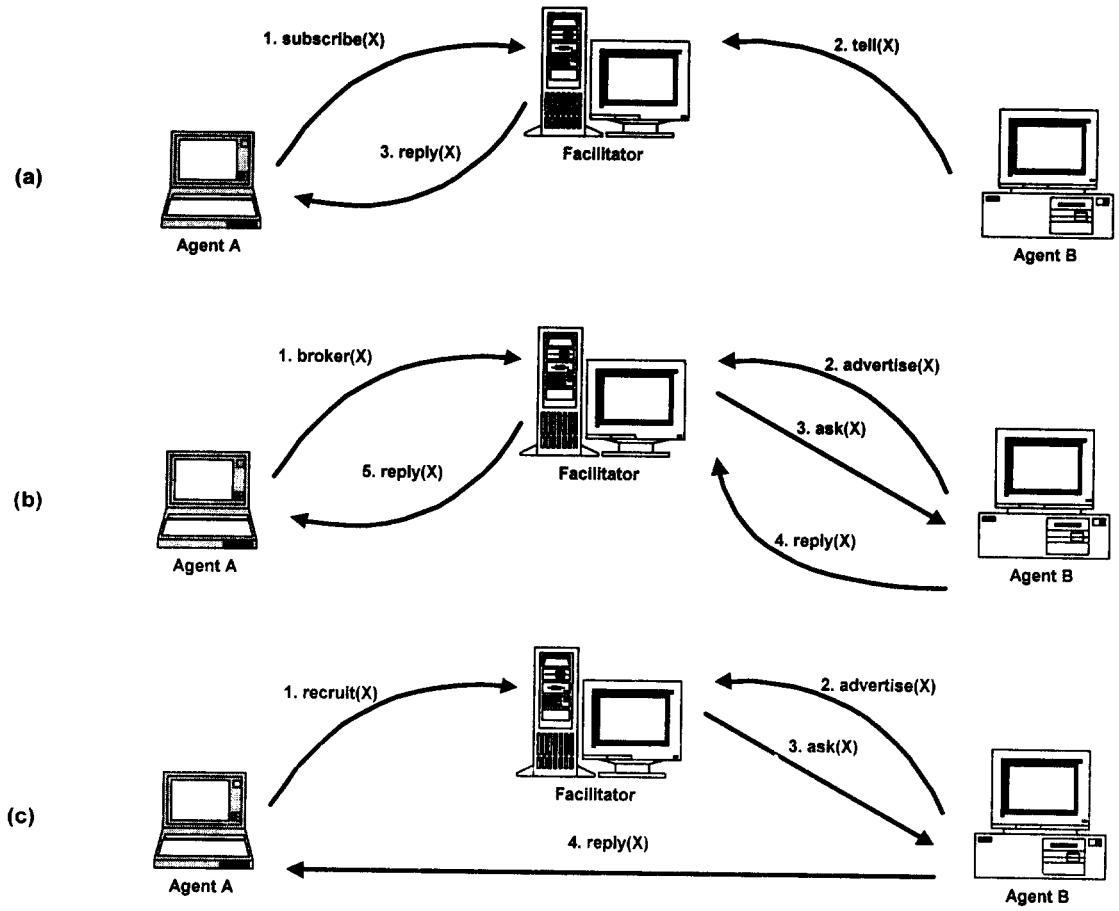

Fig. 4. 지능적인 에이전트들 사이의 통신 형태

를 들어, 에이전트 A가 퍼실러테이터에게 X에 대해 예약을 하고 에이전트 B가 퍼실러테이터에게 X에 대해 이야기를 하면 퍼실러테이터는 에이전트 A에게 X에 대한 내용을 전해준다.(Fig. 4.(a) 참조) 또는 에이전트 A가 퍼실러테이터에게 X에 대해 요청하고 에이전트 B가 퍼실러테이터에게 X에 대하여 알려줄 수 있다고 알려오면 퍼실러테이터는 에이전트 B에게 X에 대해 물어보고 결과를 얻어 에이전트 A에게 알려준다.(Fig. 4.(b) 참조) 이 경우에는 에이전트 B가 바로 에이전트 A에게 결과를 알려줄 수도 있다.(Fig. 4.(c) 참조)

5. 예제 시나리오를 통한 구현

우리는 위에서 설명한 아키텍처를 구현하기 위해 간단한 예제 시나리오를 구성하여 적용하였다. 이것은 특정 제품에 사용되는 모터를 결정하기 위한 과정이다. 에이전트들은 각각 서로 다른 역할을 맡고 있으며, 인간과 함께 작업을 수행하는 에이전트들과 컴퓨터 단독으로 모든 일을 처리하는 서비스 에이전트들로 구성된다. 에이전트들의 이질적인 문제 해결을 위해서 시나리오의 실행은 각각 Windows 95/NT, UNIX에서 자바 애플리케이션들과 애플릿들로 구현되었다.

- 설계 에이전트는 모터를 제외한 기본 설계를 하여 매니저 에이전트에게 전달한 후 매니저 에이전트와 함께 상의하여 기본 설계를 마친다.
- 매니저 에이전트는 해석 서비스 에이전트에게 기본 설계에 적당한 모터의 사양을 구해달라고 요청을 하고, 해석 서비스 에이전트는 계산된 값들을 알려준다.
- 매니저 에이전트는 부품정보 서비스 에이전트에게 모터 사양에 적당한 모터들을 찾아달라고 요청을 하고, 부품정보 서비스 에이전트는 자신

의 데이터베이스를 검색하여 해당되는 모터들과 가격, 용량 등 그에 따른 몇 가지 데이터들을 보내준다.

- 매니저 에이전트는 가장 적절하다고 판단되는 A라는 모터를 선택하여 설계 에이전트에게 알려주고, 부품정보 서비스 에이전트에게 모터 A를 체크인 하도록 한다.
- 설계 에이전트는 모터를 포함하여 마무리 설계를 마친 후, 결과를 제조 에이전트에게 보낸다. 제조 에이전트는 매니저 에이전트에게 제조 가능하다는 메시지를 보내고 제조 준비에 들어간다.
- 이때, 모터 A의 가격이 크게 오르게 되고, 정보 관리 에이전트가 부품정보 서비스의 데이터베이스를 수정한다. 부품정보 서비스 에이전트는 모터 A의 바뀐 정보를 매니저 에이전트에게 알려준다.
- 매니저 에이전트는 제품에 사용될 모터를 B라는 모터로 선택하고, 이 결과를 설계 에이전트에게 보내고, 설계 에이전트는 재설계를 하여 제조 에이전트에게 보낸다. 이때, 제조 에이전트는 모터 B를 사용하는 경우 제조가 불가능하다는 메시지를 매니저 에이전트에게 보낸다.
- 매니저 에이전트는 가격이 높더라도 모터 A를 사용할 것인지 아니면 기본 설계부터 다시 시작할 것인지를 고민하게 된다.
- 이때, 정보관리 에이전트가 새로운 모터 C를 데이터베이스에 추가시키고, 부품정보 서비스 에이전트는 모터 C가 모터 A와 유사하고 가격이 싸므로 매니저 에이전트에게 이 정보를 알려준다.
- 매니저 에이전트는 설계 에이전트에게 모터 C로 설계할 것을 요청하고, 설계 에이전트는 결과를 제조 에이전트로 보낸다. 제조 에이전트는 제조 가능하다는 메시지를 매니저 에이전트에게 보낸 다음 승인 후 제조를 시작한다.

6. 결 론

하나의 언어가 에이전트 통신 언어로 사용되어지기 위해서는 많은 필요 조건들을 가지며, 이것들 중에 일부분은 상호 배타적인 관계에 있다. KQML언어는 비록 전부는 아니지만 이러한 필요 조건들 가운데 많은 것들을 처리할 수 있는 새로운 통신 언어이다. 명백한 것은 KQML이 모든 분야에 적절한 통신언어로써 표준이라는 것이 아니라, 이것이 지능적인 소프트웨어 에이전트 아키텍처 면에서 넓은 범위에 걸쳐 유용하다는 것이다.

자바 언어는 에이전트 기술상의 문제점이었던 서버 의존도와 이종성의 문제를 해결해준다. 이것은 네트워크 상에서 통신의 효율성을 증가시키며 전 세계에 걸쳐 분포되는 재원들을 효과적으로 사용할 수 있도록 해준다.

우리는 시나리오를 통하여 그 동안 에이전트 기술상의 문제점들을 해결하고, 앞으로 에이전트 기술의 개발 방향을 찾을 수 있었다. 우리가 개발하는 에이전트 기반 환경에 여러 가지 동시 공학적 방법론의 적용과 KQML의 지식 처리 능력을 추가하여 에이전트 기술은 지능적으로 분산 공동 협력을 통한 설계 및 해석, 평가를 가능하게 할 것이다.

참고 문헌

1. M. Cutkosky, M. Genesereth, et al., "PACT : An Experiment in Integrating Concurrent Engineering Systems", IEEE Computer Special Issue on Computer Supported Concurrent Engineering, January 1993.
2. Charles Petrie, Heecheol Jeon and Mark R. Cukosky, "Combining Constraint Propagation and Backtracking for Distributed Engineering"
3. H. Robert Frost and Mark R. Cutkosky, "Design for Manufacturability via Agent Interaction", ASME Design for Manufacturing Conf., Irvine, CA, Aug. 18-22. 1996.
4. James Mayfield, Yannis Labrou, Tim Finin, "Evaluation of KQML as an Agent Communication Language", Proceedings of the 1995 Workshop on Agent Theories, Architectures, and Languages, November 7, 1995.
5. Tim Finin, Don McKay, Rich Fritzson, "An Overview of KQML: A Knowledge Query and Manipulation Language", Technical Report, Computer Science Department, University of Maryland, March 2, 1992.
6. M.R.Genesereth, R.E.Fikes, et al., "Knowledge Interchange Format, Version 3.0 Reference Manual",Technical Report Logic-92-1, Computer Science Department, Stanford University, 1992.
7. "JATLite", http://java.stanford.edu/