계획생성 모듈을 갖는 멀티에이전트 기반구조의 확장방법

이 광 로[†] · 박 상 규[†] · 장 명 욱[†] · 민 병 의[†] · 최 중 민[††]

요 약

에이전트는 자율성, 사회성, 반응성, 지속성을 갖는 독립된 프로그램으로 지식과 추론 능력을 바탕으로 사용자의 작업을 대신해 준다. 여러 영역들을 포함하는 복잡한 문제를 효과적으로 해결하기 위해서 멀티에이전트 기반구조에 대한 연구가 활발히 진행되어 왔다. 그러나 이런 기반구조에서도 사용자의 질의는 상당히 애매하고 그에 대한 문제 해결에 대한 절차가 바로 생성되지 못하는 문제점이 있다. 이를 위해 멀티에이전트 기반구조에 계획 생성모듈을 추가시켜 좀더 지능을 갖춘 멀티에이전트의 개발이 요구된다.

본 논문에서는 OAA(Open Agent Architecture)를 이용한 에이전트 시스템이 사용자의 의도 파악과 작업수행을 위한 절차를 생성하고, 분산되어 독립적으로 흩어져 사용되고 있는 지식처리 시스템을 통합하여 상호의 지식을 공유하면서 서로 협동 가능하도록 OAA를 이용한 에이전트 시스템에 계획생성 모듈 추가방법을 제안한다. 또한 이 방법의 유용성을 검증하기 위해 여행일정 에이전트 시스템에 적용하였다. 이러한 결과로 OAA를 이용한 에이전트 시스템을 사용하는 사용자는 컴퓨터 네트워크 상에서 제공되는 서비스의 제공과 사용에 있어서 좀더 편리한 인터페이스 환경을 제공 받을 수 있게 되었다. 또한 현재 독립적으로 흩어져 사용되고 있는 지식처리 시스템인 전문가 시스템이나 계획기를 통합하여 상호의 지식을 공유하면서 서로 협동으로 일을 처리할 수 있는 환경을 제공한다.

A Method of Extending a Multiagent Framework with a Plan Generation Module

Gowang-Lo Lee[†] · Myong-Wuk Jang[†] · Sang-Kyu Park[†] · Byung-Eui Min[†] · Joongmin Choi[††]

ABSTRACT

An agent is a software element that, by making use of knowledge and inference, performs tasks on behalf of the user. In general, an agent has the properties of autonomy, social ability, reactivity, and durability. Many researches on agents are more and more aiming at the multiagent systems since it is not sufficient to let a single agent do the whole things, especially in a real world where tasks require many diverse activities. However, the multiagent frameworks still have some limitations in the processing of user queries that are often ambiguous and goal-oriented. Also, a series of procedures or plans could not be generated from a single query directly.

In order to give more intelligence to the multiagent framework, we propose a method of extending the frame-

† 정 회 원: 한국전자통신연구원
†† 정 회 원: 한양대학교 전자계산학과
논문접수: 1997년 3월 5일, 심사완료: 1997년 8월 26일

work with a plan generation module. The open agent architecture (OAA), which is a multiagent framework that we developed, is integrated with UCPOP, which is a AI planner. A travel schedule management agent(TSMA) system is implemented to explore the effects of the method. The extended system enables the user to only specify goal-oriented queries, and the plans and procedures to satisfy these goals are generated automatically. Also, this system provides a cooperative and knowledge-sharing environment that integrates several knowledge-based systems and planning systems that are distributed and used independently.

1. 서 론

에이전트는 자율성(autonomy), 사회성(social ability), 반응성(reactivity), 지속성(durability)을 갖는 독립된 프로그램으로 지식과 추론 능력을 바탕으로 사용자의 작업을 대신해 준다[1][2][3]. 또한 이러한 에이전트 다수가 상호협력하면서 여러 영역들을 포함하는 복잡한 문제를 효과적으로 해결하기 위한 멀티에이전트 기반구조에 대한 연구가 활발히 진행되고 있다. 한국전자통신연구원에서는 블랙보드와 플러그 앤 플레이(plug and play) 개념을 바탕으로 여러 에이전트 간의 이질성과 분산성 문제의 해결에 초점을 맞춘 멀티에이전트 기반구조 OAA(Open Agent Architecture)[4][5][6]를 개발했다. 그러나 이런 기반구조를 바탕으로 개발된 에이전트 시스템에서도 사용자의 질의는 상당히 애매하고 그에 대한 문제 해결에 대한 절차가 바로 생성되지 못하는 문제점이 있다. 예를 들어 사용자가 "다음주에 미국 산호세로 출장가고 싶다"라고 요구하였을 때 실제로 사용자가 바라는 것은 해외출장에 필요한 제반 사항들, 즉 출장 신청서 작성, 비행기 예약, 숙소 예약 등의 일련의 절차에 대한 정보를 제공받는 것이다. 따라서 이러한 문제를 해결하기 위해 OAA에 계획생성 모듈을 추가시켜 좀더 지능을 갖춘 멀티에이전트 시스템의 개발이 요구된다.

본 논문에서는 OAA를 이용한 에이전트 시스템이 사용자의 의도 파악과 작업수행을 위한 절차를 생성하고, 분산되어 독립적으로 흩어져 사용되고 있는 지식처리 시스템을 통합하여 상호의 지식을 공유하면서 서로 협동 가능하도록 OAA를 이용한 에이전트 시스템에 계획생성 모듈 추가방법을 제안한다. 또한 이 방법의 유효성을 검증하기 위하여 여행일정 에이전트 시스템을 설계구현한 결과를 소개한다. 본 논문의 구성은 1장에서 서론을 기술하고, 2장에서는 OAA 기반 에이전트 시스템을 살펴보고, 3장에서 OAA 기반 에이전트 시스템 내에 계획생성 모듈 추가방법을 기술하고, 4장에서 여행일정관리 에이전트 시스템을 기술하고, 5장에서 여행일정관리 에이전트 시스템의 구현을 기술하고, 6장에서 결론을 기술한다.

2. OAA 기반 에이전트 시스템

OAA는 하나의 조정 에이전트와 둘 이상의 응용 에이전트, 그리고 에이전트 통신언어 (ICL: Interagent Communication Languag)로 이루어진 분산환경과 플러그 앤 플레이를 바탕으로 한 동적인 구조를 갖는다. 에이전트는 그 역할에 따라 분류가 되는데 조정 에이전트는 응용 에이전트들 간의 교류를 위한 장소 및 조정기능을 제공한다. 이와 대조적으로 응용 에이전트는 특정한 도메인의 작업을 전문적으로 처리하는 역할을 맡는다. OAA에서는 에이전트의 효율적인 제어를 위하여 조정 에이전트를 통해서만 응용 에이전트들이 다른 에이전트들과 교류할 수 있다. 응용 에이전트들은 자신들의 능력을 조정 에이전트에게 미리 선언해 놓음으로써, 자신이 잘 처리할 수 있는 작업의 수행을 요구 받을 수 있도록 한다. ICL은 에이전트들 상호간에 정보교류를 위한 메세지 형태의 통신언어로 content 층, protocol 층, wrapper 층으로 구성된 3개의 층으로 이루어진다. Content 층은 에이전트 간에 교류되는 메시지 내용을 나타내고, Protocol 층은 에이전트들 간의 약속된 규칙을 의미한다. Wrapper 층은 에이전트가 그 내용을 보고 그 메시지가 ICL로 해석되어 질 수 있는 메시지인지 아닌지 판단하며, 어느 에이전트가 보낸 것인가를 알아낼 수 있도록 한다. 에이전트들은 이 ICL을 이용하여 다른 에이전트에게 작업을 요구하거나 수행결과를 돌려주며 자신이 제공할 수 있는 서비스를 ICL로 표현한다.

OAA에서는 클라이언트 개념에 해당하는 응용 에이전트들이 조정 에이전트가 갖고있는 블랙보드에

해결할 작업을 기록하는 식으로 진행된다. 조정 에이전트나 응용 에이전트들은 모두 독립적인 하나의 프로세스이다. 에이전트들은 미리 정의된 통신 프로토콜에 의해 메세지를 주고 받을 수 있다. 따라서 각 에이전트는 네트워크가 연결된 컴퓨터라면 어디에나 놓여질 수 있고, 네트워크를 매체로 원거리에 있는 에이전트와 상호 협력하면서 작업을 처리할 수 있다.

OAA를 이용한 에이전트 시스템[3][4][5][6][7]의 특성은 사용자의 작업지시를 위탁받아서 에이전트 간의 교류를 통하여 분산적으로 실행해 주며, 사용자가 원하는 에이전트를 만들어 OAA를 이용한 에이전트 시스템에 연결만 하면 기존에 이미 만들어진 다른 기능 에이전트와 협력하며 사용자의 요구를 수행해 준다. 또한 OAA를 이용한 에이전트 시스템은 작업지시를 쉽게 하기 위해 멀티모달 입력이 가능하도록 해준다. 따라서 사용자는 원하는 작업을 음성과 펜을 사용하여 자연어로 지시하면, 에이전트 간에 상호 협력하여 원하는 작업을 수행해 준다.

그러나 현재의 OAA를 이용한 에이전트 시스템은 다음과 같은 문제해결의 한계를 가지고 있다[3][4][5][6][7]. 첫째, 사용자의 질의가 상당히 애매한 경우 그에 대한 문제 해결에 대한 절차가 바로 생성되지 못하는 한계점이 있다. 이것은 OAA를 이용한 에이전트 시스템이 계획수립자를 갖지않고 사용자의 작업지시를 자연어 처리하여 단순히 ICL로 맵핑하는데 있다. 둘째, 사용자는 작업을 지시할 때 완전한 정보를 포함한 작업을 한번에 지시해야만 처리가 가능하다. 즉 에이전트와 사용자간의 interaction이 없고 사용자의 일방적인 지시에 의해 처리된다. 세째, 작업 스케쥴링이나 에이전트 관리에 단순 정보만을 사용하기 때문에 에이전트의 상태나 동적인 작업순서를 고려한 효율적인 처리가 곤란해 효율성에 한계가 있다. 이러한 한계들을 극복하기 위해 OAA를 이용한 에이전트 시스템에 계획생성 모듈을 추가시켜 좀더 지능적인 작업을 할 수 있는 멀티에이전트 시스템 개발이 요구된다.

3. 계획생성 모듈 추가방법

OAA를 이용한 에이전트 시스템의 지식처리를 위해 계획생성 모듈 추가는 계획수립 지식의 특성에 따라 3개의 레벨로 나누었다. 즉, 사람과 에이전트간의 interaction으로 도메인에 종속적이지 않고 응용 에이전트의 구성과 관계없이 항상 상주하는 고정형 지식, 에이전트와 에이전트간의 interaction으로 도메인에 종속적이며 응용 에이전트의 구성과 관계없이 항상 상주하는 종고형 지식, 에이전트와 응용 소프트웨어간의 interaction으로 도메인에 종속적이며 응용 에이전트의 구성에 따라 상주하는 유동형 지식 등으로 분류한다. 또한 각각의 계획수립을 순차적으로 상위레벨 계획수립, 중위레벨 계획수립, 하위레벨 계획수립으로 정의한다. 상위레벨 계획수립은 사용자인터페이스 에이전트의 지능화로서 사용자의 의도파악, 멀티모달 처리, 결과가공 출력 등 사용자와 에이전트간의 정보전달을 위한 계획수립이다. 중위레벨 계획수립은 조정 에이전트의 지능화로서 작업의 관리, 에이전트 관리, 작업 수행전략 수립, 작업분배 및 스케쥴링, 결과 통합 등 에이전트간의 효율적인 정보전달과 작업관리를 위한 계획수립이다. 하위레벨 계획수립은 응용 에이전트의 지능화로서 일정관리 에이전트의 경우 일정관리에 관련된 지능화, 기차예약 에이전트의 경우 기차예약에 관련된 지능화 등 특정영역에서 고유한 일을 처리하기 위해 필요한 계획수립이다.

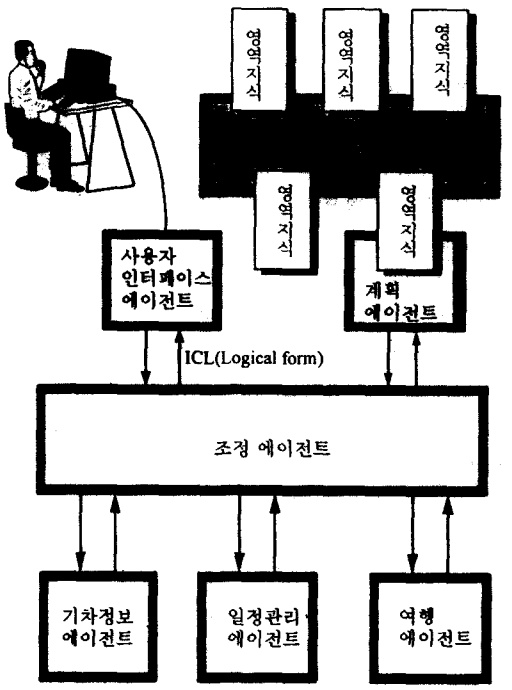
OAA를 이용한 에이전트 시스템의 지식처리 능력을 부여하기 위해서는 궁극적으로 상위레벨 계획수립, 중위레벨 계획수립, 하위레벨 계획수립이 필요하지만 본 연구에서는 먼저1단계로 하위레벨 계획수립을 여행일정관리 에이전트 시스템에 적용하여 구현했다. 개발한 여행일정관리 에이전트 시스템의 업무는 다음과 같이 제한했다. 우선 국내 여행만을 취급하며, 교통편은 육상 운송수단만을 고려했다. 또한 사용자와 간단한 질의를 통하여 필요한 정보를 추출하고, 이렇게 추출한 정보를 바탕으로 여행에 필요한 계획을 수립했다.

4. 여행일정관리 에이전트 시스템

4.1 여행일정관리 에이전트 시스템의 구조

여행일정관리 에이전트 시스템의 구조는 그림 1과 같이 크게 조정에이전트, 응용에이전트, 계획수립자 등 3개의 부분으로 나눈다. 여행 에이전트는 일정관리나 기차 에이전트와 마찬가지로 하나의 응용 에이전트로써 조정 에이전트에 연결되어 작업을 수행한다.

따라서 여행 에이전트가 다른 응용 에이전트로부터 정보를 필요로 할 때(예를 들어 기차 예약을 위해 기차 정보 에이전트로부터 하이텔이나 천리안 등에서 티켓 정보를 가져오거나 사용자의 일정을 확인하기 위해 일정관리 에이전트의 도움을 얻을 때), 조정 에이전트를 조정자로 하여 에이전트간 통신을 한다. 그러나 여행 에이전트가 하는 일 중 계획수립에 관한 부분은 따로 분리하여 계획 에이전트로 구현하였는데 이 계획 에이전트는 UCPOP[8]이라 불리는 기본적인 계획자(planner)를 포함한다. 이렇게 함으로써 얻을 수 있는 이점은 여러 응용 프로그램이 하나의 계획자를 공유할 수 있다는 것이다. 기본적인 계획자를 여행 에이전트가 사용하기 위해서는 여행 에이전트에 고유한 영역 지식을 표현할 필요가 있는데, 이는 여행 에이전트와 기본적 계획자가 모두 접근할 수 있는 데이터로서 여행 에이전트의 행동을 표현하는 operator schema와 여행 에이전트의 업무에 필요한 제반 영역 지식이 여기에 포함된다. 추후에 다른 응용 에이전트가 기본적 계획자를 사용하려면 이와 같은 공유 지식베이스를 추가하기만 하면 된다.



(그림 1) 여행일정관리 에이전트 시스템의 구조
(Fig. 1) An architecture of TSMA(Travel-Schedule Management Agent) system

여행일정관리 에이전트 시스템의 동작원리는 조정 에이전트가 수행되면 각각의 서비스 제공을 위한 응용 에이전트는 조정 에이전트에게 연결을 시도한다. 이때 응용 에이전트는 자신이 제공할 서비스 리스트를 조정 에이전트에게 등록한다. 또한 지식처리 지원을 위한 영역지식이 있으면 계획 에이전트에게 등록할 것을 조정 에이전트에게 요구한다. 이렇게 해서 계획수립자의 지식베이스를 동적으로 구축한다. 사용자는 사용자 인터페이스 에이전트를 통해 작업을 요구한다. 사용자 인터페이스 에이전트는 요구한 작업을 ICL로 변환한 후 조정 에이전트에게 전달한다. 전달된 작업은 조정 에이전트에 의해 해당 에이전트에 전달되어 작업을 수행한다. 이때 다른 에이전트의 도움이 필요할 경우는 조정 에이전트를 매체로 상호 협력하면서 작업을 수행한다. 수행된 작업결과는 작업수행 요구의 역순으로 사용자에게 전달된다.

4.2 처리과정

여행일정관리 에이전트 시스템은 주어진 작업을 여러 에이전트가 서로 협동하여 복잡한 작업을 수행한다. 여행일정관리 에이전트 시스템의 처리과정을 단계별로 기술하면 다음과 같다. 단, 아래의 예는 사용자가 자연어처리 에이전트의 도움 없이 직접 에이전트에게 작업을 지시한 경우이다.

(1) 우선 사용자(LEE)는 사용자 인터페이스 에이전트를 통해 자신이 여행하고 싶은 도착시간 등에 대한 정보를 입력하게 된다. 예를 들면 "97년 1월 23일 11:30까지 ETRI에 도착하고 싶다"를 수행하려면 사용자 인터페이스 에이전트의 query dialog box에 post_query(at('LEE', 'ETRI', '1130', '970123', R)를 입력하게 된다.

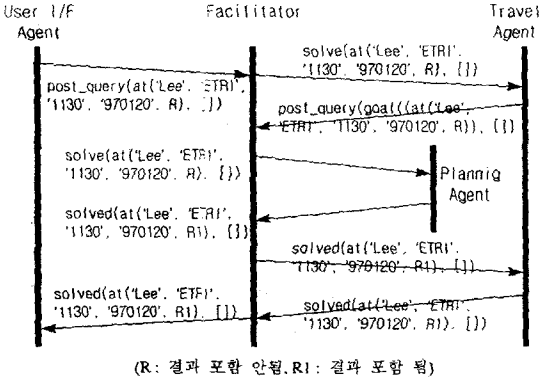
(2) 사용자인터페이스 에이전트를 통해 입력된 사용자의 query는 조정 에이전트로 전달되고 조정 에이전트는 등록된 capability database를 이용하여 사용자의 query를 해결할 수 있는 에이전트를 결정하게 된다. 위의 예에서 조정 에이전트는 'at'이라는 goal을 해결할 수 있는 에이전트가 여행 에이전트라는 것을 결정하여 여행 에이전트에게 위의

query와 함께 solve라는 메세지를 보낸다.

(3) 요구사항을 전달받은 여행 에이전트는 요구사항의 내용이 자신이 바로 실행할 수 있는 것인지를 자신의 데이터 베이스를 참조하여 판단하고, 그렇지 않을 경우 계획자를 불러 계획을 생성하도록 한다. 이를 위해 사용자의 원래 query에 'goal'이라는 predicate를 첨가시켜 계획 에이전트가 이를 처리하도록 한다. 즉, 여행 에이전트는 post_query (goal (at('LEE, ETRI', '1130', '970123', R)), [ ])의 메세지를 조정 에이전트를 통해 계획 에이전트에게 전달한다.

(4) 계획 에이전트는 전달받은 요구사항을 만족시킬 수 있는 계획을 수립하기 위해, 여행 에이전트와 공유하고 있는 도메인 지식, 그리고 다른 응용 에이전트와 공유하고 있는 도메인 지식을 활용한다. 이 과정에서 수립된 계획에는 다른 응용 에이전트가 정보를 제공한 결과를 본 후 어떤 행동을 취할 것인지를 결정하는 단계가 포함될 수도 있는데, 이를 표현하기 위해 조건부 계획을 세워 여행 에이전트에게 준다.

(5) 여행 에이전트는 계획자에 의해 수립된 계획을 받아 실행한다. 여행 에이전트에게 고유한 행동만을 포함하는 계획인 경우 단순히 주어진 계획을 순서대로 실행한 결과를 조정 에이전트에게 돌려 주면 되지만, 다른 에이전트에게 보낼 질의를 처리해 줄 것을 의뢰하고 그 답이 올 때까지 기다렸다가 그 답의 내용에 따라 적절한 행동을 취한다.



(R: 결과 포함 안됨, R1: 결과 포함 됨)

(그림 2) 작업처리를 위한 내부처리과정

(Fig. 2) Flowchart of internal processing among agents

이러한 처리과정을 그림으로 나타내면 그림 2와 같이 나타낼 수 있다.

4.3 여행일정관리 에이전트 시스템의 특징

여행일정관리 에이전트 시스템의 특징은 OAA에서 제공하는 특징[4][6] 이외에 다음과 같은 특징을 갖고 있다.

계획수립을 할 때 동적인 외부 환경과 불완전한 정보 하에서도 계획수립이 가능하도록 조건부 계획(conditional plan)과 정보결합(information binding)을 이용한다. 즉 여러 에이전트에 의해 공유되는 기본적 계획자는 정보획득을 포함하는 계획을 수립할 수 있고, 정보획득의 결과에 따라 그 행동을 달리 선택할 수 있도록 조건부 계획을 세울 수 있게 했다. 따라서 계획수립 도중에 필요한 정보가 있으면 계획수립을 잠시 중단하고 필요한 정보를 에이전트 시스템을 통해 요구하고 얻는다. 정보가 얻어지면 계획수립을 계속 진행하여 계획을 수립한다.

시스템의 확장을 용이하게 하기 위해서 영역 지식을 가능한 한 선언적으로 표현하고 지식 내에 변수를 사용했다. 이렇게 함으로써 추후에 에이전트의 능력을 확장하여 다른 운송수단을 포함할 경우 에이전트의 프로그램 모듈을 수정하지 않고 영역지식 만을 추가, 수정함으로써 시스템을 용이하게 확장할 수 있다. 또한 지식 내에 변수를 사용함으로써 시간이나 상황에 따라 변하는 정보를 따로 구축하거나 외부에서 정보를 얻을 수 있다.

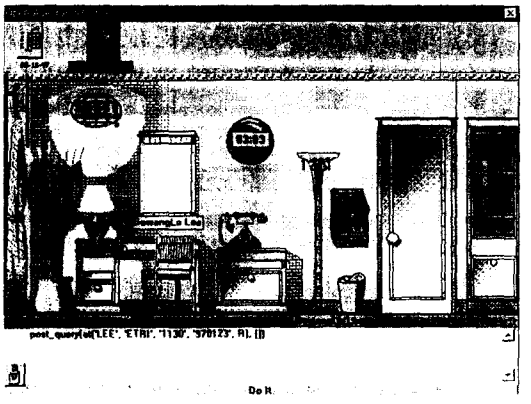
영역 지식과 계획수립 엔진을 분리하여 계획생성 모듈을 여러 에이전트에서 공유할 수 있다. 따라서 에이전트 시스템을 구성하는 응용 에이전트에 따라 런타임(run time) 중에도 동적인 지식베이스를 구축할 수 있다. 특히, 이 구조는 이동 에이전트에게 계획수립을 지원할 경우 문제가 되는 계획수립 엔진의 구축과 이동의 문제를 해결할 수 있다. 즉 이동 에이전트는 자신이 필요로 하는 지식만을 들고 다니다가 필요시에 에이전트 시스템에 있는 계획자에게 등록한 후 계획수립을 하면된다.

여행 에이전트가 사용자로부터 입력으로 받아들이는 명령에는 시간이나 장소 등을 지칭하는 표현이 포함되어 있게 마련이다. 우리의 여행 에이전트는 이들 시간/장소에 관한 표현에 가능한 한 제한을 두지 않

는다. 예를 들어 날짜를 표현할 때, 1997년 1월 23일은 "97.1.23", "970123", "97/1/23", "1월 23일", "1월 넷째 월요일"등, 어떤 방법으로 표현되어도 처리가 가능하다.

5. 여행일정관리 에이전트 시스템의 구현

여행일정관리 에이전트 시스템의 구현은 크게 OAA를 이용한 에이전트 부분, 계획수립자 그리고 LISP 엔진부분 3개의 부분으로 나눌 수 있다. OAA를 이용한 에이전트 부분은 C로 구현되었으며, 각 에이전트들은 독립적으로 구성되어 있기 때문에 네트워크에 물려있는 호스트라면 어디든지 탑재가 가능하다. 계획수립자는 불완전한 정보 하에서의 계획수립을 위한 시스템을 설계하기 위하여 백워드 계획수립(backward planning)이 가능한 기존의 계획자 중에서 University of Washington에서 개발된 UCPOP을 사용했다. LISP 엔진부분은 C로 구현된 OAA를 이용한 에이전트 부분에 LISP으로 구현된 UCPOP을 embed 시키기 위한 부분이다. 즉 UCPOP은 계획수립에 필요한 정보나 계획수립 결과를 OAA를 이용한 에이전트 부분에 LISP 형식에 준하여 전달한다. 따라서 이들간의 정보교환을 위해 UCPOP의 입력과 출력을 분석하여 OAA를 이용한 에이전트 부분에서 사용할 수 있도록 변환시켜주는 부분이 요구되며, 이것을 구현한 것이 LISP 엔진부분이다. 그림 3에 여행일정관리 에이전트 시스템의 초기화면을 나타낸다.



(그림 3) 여행일정관리 에이전트 시스템의 초기화면
(Fig. 3) Main screen of TSMA(Travel-Schedule Management Agent) system

5.1 여행계획을 위한 도메인 규칙정의

여행계획을 위한 도메인 지식과 operator schema는 다음과 같이 정의되며, 이 도메인 규칙은 다음과 같은 형태로 구성된다.

```
(define (domain DOMAIN-NAME)
  (:fact FACT-DEFINTION)
  (:operator OPERATOR-DEFINITION))
```

여기서 fact와 operator 정의는 여러 개가 올 수 있다. fact는 도메인지식을 나타내며 선언적 fact 외에도 변수를 포함한 일반적인 fact를 정의할 수 있다. 예를 들면, fact가 다음과 같은 경우 x는 변수가 아니어야 하고 x의 값이 ETRI나 SNU인 경우 그곳의 가장 가까운 기차역인 Daejon이나 Seoul을 y의 값으로 할당해주게 된다.

```
(:fact (nearest_train_station ?x ?y)
  (cond ((variable? ?x):no-match-attempted)
    (t (list (setb ?y (cadr (assoc ?x '(ETRI Daejon)
(SNU Seoul)))))))))
```

operator는 원하는 goal을 달성하기 위한 조건들을 기술해주며 각각의 조건들은 또다시 다른 oprator에 의해 계획을 달성하기 위한 규칙이 정의될 수 있다. 가령 예를 들어, operator가 다음과 같은 경우 oprator는 :effect 부분에서 선언된 'at'이라는 goal을 만족시키기 위해서는 :precondition에서 선언된 각각의 조건들이 모두 만족되어야 한다는 것(:and로 되어있으므로)을 나타낸다.

```
(:operator travel_by_train
:parameters(?subj ?src ?dest ?t_dept ?t_arr ?data ?class
?train_no)
:precondition
  (:and (train_station ?dest)
    (residence_city ?subj ?src)
    (default_train_class ?subj ?class)
    (best_train_conn ?src ?dest ?t_dept ?t_arr ?date ?
    class ?train_no)
    (located ?subj ?src ?t_dept ?date)
```

(has_ticket ?subj ?date ?train_no))
:effect(at ?subj ?dest ?t_arr ?date))

5.2 수행결과

서울에 살고있는 "LEE"라는 사용자가 "1월 23일 ETRI에 11:30까지 도착하고 싶다"라고 입력했을 때의 하위레벨계획의 처리과정과 결과를 나타낸다.

```
Initial:((RESIDENCE_CITY LEE SEOUL) (LOCATED
        LEE SEOUL 900 970123)
        (SEATS_LEFT 970123 999)) Given 0
Step 1:(PURCHASE-TRAIN-TICKET LEE 970123
        999 6500) Created 3
        0→(SEATS_LEFT 970123 999)
Step 2:(TRAVEL-BY-TRAIN LEE SEOUL DAEJON
        900 1100 970123 2 999)
        Created 2
          3→(HAS_TICKET LEE 970123 999)
          0→(LOCATED LEE SEOUL 900 970123)
          0→(RESIDENCE_CITY LEE SEOUL)
Step 3:(TRAVEL-BY-TAXI-FROM-TRAIN-STOP
        LEE DAEJON ETRI 1100 1130 30 5000
        970123) Created 1
          2→(AT LEE DAEJON 1100 970123)
Goal:(AT LEE ETRI 1130 970123)
          1→(AT LEE ETRI 1130 970123)
```

이 계획의 의미는 사용자 LEE가 ETRI에 97년 1월 23일 11시30분까지 가려면 다음과 같은 3가지 단계를 거쳐야 한다는 것이다. 1)먼저 97년 1월 23일 999번 기차의 차표를 6500원에 구입하고, 2)서울에서 대전으로 가는 9:00 출발 기차를 타고, 3)대전역에서 ETRI까지 택시를 탄다.

5.3 연구성과 및 보완점

본 논문에서는 OAA를 이용한 에이전트 시스템이 사용자의 의도 파악과 작업수행을 위한 절차를 생성하고, 분산되어 독립적으로 흩어져 사용되고 있는 지식처리 시스템을 통합하여 상호의 지식을 공유하면서 서로 협동 가능한 에이전트 시스템의 확장을 위해 먼저 OAA를 분석하고 OAA를 이용한 에이전트 시스템에 계획 생성 모듈을 추가하는 방법을 제안했다. 또한 OAA를 이용한 에이전트 시스템에서의 지식을 세단계로 분류하고 1단계로 하위레벨의 유용성을 검증하기 위해 여행일정 에이전트 시스템에 적용하였다. 이러한 연구의 성과는 다음과 같다.

첫째, 동적환경변화에 대한 계획수립이 가능하며, 지식베이스의 사용에 있어서 투명성을 제공한다. 즉, 현재 대부분의 계획 수립자는 처리 과정에 있어서 필요한 정보가 지식베이스에 모두 구축되어 있어야 한다. 그러나 실질적인 계획수립 문제에는 이러한 정보가 외부에 있는 경우가 많다. 본 방법은 계획수립 도중 필요한 정보를 지식베이스에 구축된 정보뿐만 아니라 지식베이스 외부에 있는 정보에 대해 에이전트를 통해서 얻고, 얻어진 정보를 바탕으로 계획수립을 한다. 이때 필요한 정보취득 방법은 단순하게 조정 에이전트에게 요청하면 조정 에이전트가 정보를 제공할 수 있는 에이전트에게 요청해서 결과를 돌려준다.

둘째, 분산된 지식 시스템에 대한 재사용성과 지식 시스템에 대한 통합/확장 방법에 있어서 매우 효율적이다. 즉, 현재 독립적으로 만들어진 응용 프로그램이나 지식처리 시스템을 개별 에이전트화 하여 멀티 에이전트 기반구조에 연결함으로써 필요한 서비스나 지식을 새로 만들지 않고도 쉽게 사용할 수 있다. 여기서 에이전트화에 대한 생성을 간단히 소개하면 다음과 같다. 에이전트 기본기능에 대해서는 기존에 만들어진 에이전트 라이브러리를 그대로 연결하여 사용하면 되고, 필요한 기능에 대하여 do_event라는 함수를 정의해서 응용 프로그램의 기능을 연결한다. 계획수립자에 대한 에이전트화는 계획수립에 필요한 지식을 구축하고, 계획수립자와 에이전트간의 정보교환을 위한 서비스 기능들(solvable list)을 만들어주면 된다. 이러한 개별 에이전트화에 대한 상세한 내용은 참고문헌[9]에 기술되어 있다.

현재 본 연구의 보완점으로는 다음과 같다. 구현한 시스템의 지원 기능과 적용한 응용 영역을 아주 단순화 시킴으로써 해결 능력이나 일반성이 제한적으로 보인다. 따라서 일반성을 고려한 응용 영역에 대한 지식베이스의 확장이나 다른 지식베이스 시스템의 통합이 요구된다. 그리고 지식베이스의 동적 확장면에 있어서 지식간의 일관성, 모순성을 처리하는 결합(integrity) 문제를 고려해야 하나 현재로서는 인위적

인 수작업으로 지식간의 일관성, 모순성을 처리하고 있다. 그러나 이러한 문제는 매우 어려운 문제로서 또다른 연구 대상으로 추후의 연구과제로 하고 있다.

6. 결 론

멀티에이전트 구조인 OAA를 이용한 에이전트 시스템의 기능확장에 따라 OAA를 이용한 에이전트 시스템을 사용하는 사용자는 컴퓨터 네트워크 상에서 제공되는 서비스의 제공과 사용에 있어서 좀더 편리한 인터페이스 환경을 제공 받을 수 있게 되었다. 또한 현재 독립적으로 흩어져 사용되고 있는 지식처리 시스템인 전문가 시스템이나 계획기를 일반 응용 소프트웨어와 통합하여 사용할 수 있는 환경과 지식처리 시스템간의 상호지식을 공유하면서 서로 협동으로 일을 처리할 수 있는 환경을 제공한다. 특히 각 에이전트들이 독립적으로 구성되어 있기 때문에 네트워크에 물려있는 호스트라면 어디든지 탑재가 가능하다. 따라서 사용자 인터페이스 에이전트를 PDA나 notebook PC와 같은 휴대성이 편리한 디바이스에 탑재하여 어디서든지 지식처리 시스템을 사용할 수 있고 기존에 구현된 시스템에 통합이 용이하다.

향후 연구내용은 OAA를 이용한 에이전트 시스템의 지식처리를 위한 확장과 이동 에이전트 지원을 위한 확장으로 다음과 같다. 첫째, 상위레벨 계획수립와 중위레벨 계획수립의 연구 둘째, 이동 에이전트를 위한 OAA의 기능확장 및 이동 에이전트용 계획수립 엔진을 개발 세째, 지식간의 일관성, 모순성을 처리하는 결합문제 해결 넷째, 확장된 기반구조 지원을 위한 에이전트 개발툴[9]의 확장 등이다.

참 고 문 헌

[1] Seiji Yamada, "Planning for an Agent," 일본인공지능학회지, vol.10. no.5, pp.677-682, September 1995.

[2] Michael R. Genesereth, Steven P. Ketchpel, "Software Agents," Communication of the ACM, Vol.37, No.7, pp.48~53, July 1994.

[3] 이광로, 장명욱, 박상규, 민병의, "멀티에이전트 기반 전자우편 시스템," 한국정보과학회 충청지부 추계학술발표 논문집, 제9권, 제1호, pp.65-69, 11월 1996년.

[4] P. Cohen, A. Cheyer, M. Wang, and S.C. Baeg, "An Open Agent Architecture," In O. Etzioni, editor, *Proceedings of the AAAI Spring Symposium Series on Software Agents*, pp.1-8, Stanford California, March 1994. American Association for Artificial Intelligence.

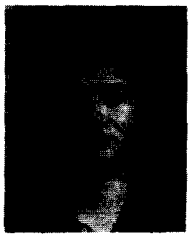
[5] Soon Cheol Baeg, Sang Kyu Park, Joong Min Choi, Myung Wuk Jang, and Young Hwan Lim, "Cooperation in Multi-agent Systems," Intelligent Computer Communications (ICC '95), Cluj-Napoca, Romania, pp.1-12, June 1995.

[6] 백순철, 최중민, 장명욱, 박상규, 임영환, "이형 분산 환경에서 에이전트들간의 이형성을 극복하기 위한 멀티에이전트 기반구조," 정보과학회논문지(C), 제2권, 제1호, pp.24-37, 3월 1996년.

[7] Joongmin Choi, Sang-Kyu Park, Soon-Cheol Baeg, Myeong-Wuk Jang, Gowang-Lo Lee, and Young-Hwan Lim, "Message-Based Agent Communications in a Tightly Coupled Multiagent System," Fourth Golden West International Conference on Intelligent Systems (GWICS-95), San Francisco, CA., pp.194-198, June 1995.

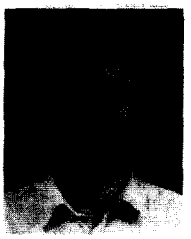
[8] Anthony Barrett, Dave Christianson, Chung Kwok, Keith Golden, Scott Penberthy, Ying Sun, and Daniel Weld, "UCPOP User's Manual(version 4.0)," Technical Report 93-09-06d, University of Washington, September 1995.

[9] 이광로, 박상규, 장명욱, 민병의, 황승구, "에이전트 시스템 개발도구에 관한 연구," 정보과학회논문지(C), 제2권, 제1호, pp.24-37, 3월 1997년.



이 광 로

1986년 일본 Fukuoka 공업대학교 전자기계과 졸업(학사)
1988년 일본 Ritsumeikan 대학교 전기공학과 졸업(석사)
1988년~현재 한국전자통신연구소 인공지능연구실 선임연구원
1994년~1995년 미국 SRI International(International Fellow)
관심분야: 에이전트 시스템, HCI, 패턴인식



박 상 규

1982년 서울대학교 컴퓨터공학과 졸업(학사)
1984년 한국과학기술원 전산학과 졸업(석사)
1984년~1987년 대림산업 전산실 근무
1989년~현재 한국과학기술원 전산학과 박사과정 수료
1987년~현재 한국전자통신연구소 인공지능연구실 선임연구원
관심분야: 에이전트 시스템, 정보검색, HCI

장 명 욱

1990년 고려대학교 전산학과 졸업(학사)
1992년 한국과학기술원 전산학과 졸업(석사)
1992년~현재 한국전자통신연구소 인공지능연구실 연구원
관심분야: 에이전트 시스템, 패턴인식

민 병 의

1982년 한양대학교 졸업(학사)
1984년 한국과학기술원 전기 및 전자공학과 졸업(석사)
1992년 한국과학기술원 전기 및 전자공학과 졸업(박사)
1984년~1987년 대림산업 기술연구소
1987년~현재 한국전자통신연구소 인공지능연구실 실장
관심분야: 멀티미디어 시스템, 에이전트 시스템

최 중 민

1984년 서울대학교 컴퓨터공학과 졸업(학사)
1986년 서울대학교 컴퓨터공학과 졸업(석사)
1993년 State University of New York at Buffalo, Computer Science 졸업(박사)
1993년 7월~1995년 한국전자통신연구소 인공지능연구실 선임연구원
1995년~현재 한양대학교 전자계산학과 조교수
관심분야: 인공지능, 에이전트 시스템, 지식표현 및 추론