지식레벨을 이용한 다중 에이전트 협동 정보시스템*

Multi-Agent Based Cooperative Information System using Knowledge Level

강 성 희** 박 승 수**
(Sung-hee Kang) (Seung-soo Park)

요 약 분산 협동 정보시스템은 분산, 이질적인 정보 환경에서 다양한 지식원과 문제 해결 능력 등을 가지면서 이들이 서로 통합되어 사용되는 시스템을 말한다. 이러한 시스템에서는 구성 정보를 서로 활용하고 제어할 수 있는 조정기능이 필요하고 이러한 조정기능의 역할에 따라 전체 시스템의 양태가 다양하게 결정된다.

본 논문에서는 다중 에이전트 패러다임을 적용하여 분산 협동 정보시스템의 한 가지 모델을 제시하고 테스트 시스템을 구현해보았다. 제안 시스템은 분산된 구성원을 에이전트화하고 이들 간의 협동을 원활하게 할 수 있도록 조정 에이전트를 사용하고 있다. 본 시스템에서 제안된 조정 에이전트는 지식의 세밀도(granularity) 레벨을 이용하여 에이전트 활성화 순서를 조정한다. 즉, 처음에는 단순하면서도 빠른 속도의 지식 형태를 활용하고 그 결과에 따라 좀 더 복잡한 지식 베이스를 활용하도록 하는 것이다.

제안 구조는 동일한 주제에 대하여 여러 레벨의 지식원이 존재할 때 이들을 적절히 순서화함으로써 마치 인지과정에서 의식의 초점(focusing)이 자연스럽게 바뀌는 것과 같은 효과를 얻을 수 있을 것이다.

Abstract Distributed cooperative information system is the one that has various knowledge sources as well as problem solving capabilities to get information in a distributed and heterogeneous data environment. In a distributed cooperative information system, a control mechanism to facilitate the available information is very important, and usually the role of the control mechanism determines the behavior of the total system.

In this research, we proposed a model of the distributed cooperative information system which is based on the multi-agent paradigm. We also implemented a test system to show its feasibility. The proposed system makes the knowledge sources into agents and a special agent called 'facilitator' controls the cooperation between the knowledge agents. The facilitator uses the knowledge granularity level to determine the sequence of the activation of the agents. In other words, the knowledge source with simple but fast processing mechanism activates first while more sophisticated but slow knowledge sources are activated late. In an environment in which we have several knowledge sources for the same topic, the proposed system will simulate the focusing mechanism of human cognitive process.

Keywords multi-agent, facilitator, knowledge level

1. 서론(Introduction)

네트워크 기술의 발달로 인하여 기존의 독립적으로 수행되던 정보시스템의 형태가 분산 컴퓨팅 인프라 구조에 의해 서로 연결된 분산 협동 정보시스템 형태

* 본 연구는 과학재단 특정기초(98-0102-0101-3)의 지원에 의해 수행되었음.
** 이화여자대학교 컴퓨터학과
Dept. of Computer Science and Engineering in Ewha Womans University

로 바뀌고 있다. 이러한 정보시스템 형태는 분산 데이터베이스 시스템(Distributed DB)이나 분산 인공지능(DAI) 등과 같은 분야에서 활발하게 연구되고 있으며, 미래의 정보 환경에서는 그 비중이 더욱 높아질 것으로 기대되고 있다.

미래의 정보 환경은 이질적이며 다양한 형태의 지식원이 산재되어 있고 의사결정 또는 문제 해결 능력 등도 분산되어 있을 뿐만 아니라 이들 개별적인 요소들이 서로 결합되어 좀 더 복잡한 문제 해결에 사용되어질 수도 있을 것이다. 다양한 특성을 갖는 분산 데이터 환경에서는 문제의 특성에 따라 적합한 지식원과 적당한 문제 해결 방식을 취하는 분산된 개별 요소 시스템들이 존재하게 되고 이들이 개별적으로 또는 서로 협동함으로써 주어진 문제를 유연하고 지능적으로 해결할 수 있는 구조가 필요하다. 최근 이러한 목적으로 다중 에이전트(multi-agent) 패러다임이 많이 제안되고 있다[5][11][15].

다중 에이전트 패러다임은 여러 개의 소프트웨어 에이전트들이 각자 독립적이며 자발적인 작업을 하면서 서로 협동하여 공동의 목표를 달성하는 것이다. 이러한 구조하에서는 개별 에이전트들이 효과적으로 협동하여 더욱 지능적인 양태를 보일 수 있도록 이들 구성 에이전트를 제어하는 기능이 요구된다. 이러한 조정 기능은 조정자(facilitator)라는 특별한 에이전트에 의해 통합 또는 조정되어진다. 특히 구성 에이전트들의 협동(cooperation)의 정도에 따라 조정자의 역할이 달라지게 되며 지능적이고 적응력 있는 조정자는 좀 더 복잡한 문제 해결을 가능하게 한다. 즉 기존의 인공지능 시스템에서 지능을 담당하던 추론 기능이 종래의 단일 시스템 중심의 추론 기계(inference engine) 형태에서 조정자의 양태(behavior)에 따른 형태로 전환되는 것이다.

본 논문은 이질적이며 분산된 정보 환경에 적당한 분산 협동 정보시스템의 한 가지 모델을 다중 에이전트 방식을 이용하여 제시하고 이를 간단한 테스트 시스템으로 구현해 보았다. 제안 시스템에서는 분산된 지식원을 독립적인 에이전트로 만들고 각 구성 에이전트는 자신의 도메인 지식에 적당한 방법으로 문제를 해결하며 다른 에이전트들과 서로 통신할 수 있도록 하였다. 본 시스템에서 제안된 조정 에이전트는 지식의 세밀도(granularity) 레벨을 이용하여 에이전트 활성화 순서를 조정한다. 지식의 세밀도 레벨이란 그 지식원이 어떤 대상에 대하여 갖는 지식의 질이 얼마나 자세하고 정확한가를 말하는 것이다. 제안 구조는 동일한 주제에 대하여 여러 레벨의 지식원이 존재할 때 이들을 적절히 순서화함으로써 마치 인지과정에서 의식의 초점(focusing)이 자연스럽게 바뀌는 것과 같은 효과를 얻을 수 있을 것이다. 즉, 처음에는 단순하면서도 빠른 속도의 지식 형태를 활용하고 그 결과의 만족여하에 따라 좀 더 복잡한 지식 베이스를 활용하도록 하는 것이다.

본 논문에서는 이러한 지식레벨의 응용 가능성을 보이기 위하여 실제 동물 도메인에 대한 몇 가지 서로 다른 레벨의 지식원을 효과적으로 활용하는 조정자를 설계하고 이를 구현해 보았다. 이들 서로 다른 레벨의 지식은 기술적으로는 단순한 데이터베이스(database), 단순한 지식베이스(knowledge base), 그리고 좀 더 복잡한 형태의 퍼지 기능을 갖는 지식베이스로 이루어지지만 인지학적 측면에서 본다면 표면지식(shallow knowledge), 중간지식, 심층지식(deep knowledge) 등의 형태로 구분된다고 가정할 수도 있을 것이다.

본 논문의 구성은 2장에서 다중 에이전트와 조정자의 역할을 중심으로 기존의 연구를 소개하고 3장에서는 제안 시스템의 설계에 대하여 그리고 4장에서는 구현에 대하여 기술한 뒤 결론을 내린다.

2. 다중 에이전트와 조정자의 역할

2.1 다중 에이전트의 구성

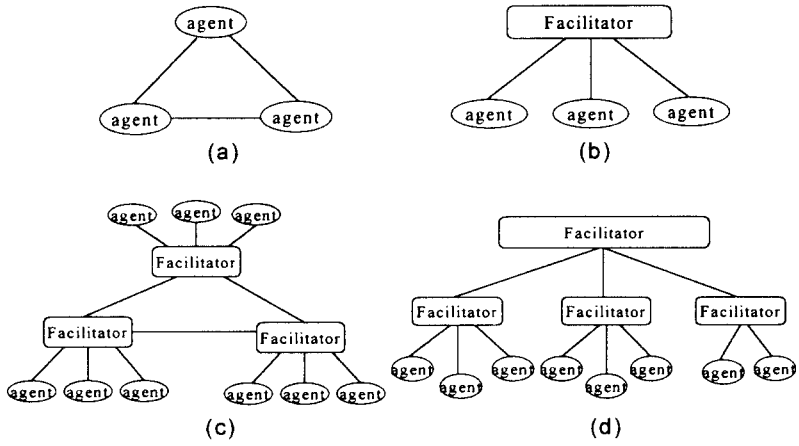
지능형 에이전트의 개발은 현재 여러 형태로 진행되고 있으나 다중 에이전트의 구성이나 조정 방법 등에 대해서는 아직 뚜렷한 해법이 나오지 못하고 있는 실정이다. 여기에서는 구성 에이전트간의 협동 구조 관점에서 다중 에이전트 시스템과 조정자의 역할을 조사, 분석하였다.

다중 에이전트를 구조적인 측면에서 분류하자면 크게 조정자가 존재하지 않는 경우〈그림 1(a)〉와 존재하는 경우로 나눌 수 있다. 조정자가 존재하지 않는 경우는 각각의 구성 에이전트들이 똑같은 자격으로 시스템 내의 다른 에이전트들에 대한 정보 및 협동 작업을 위한 정보를 가지고 있어야 하며 공통의 지정된 프로토콜을 사용하여 다른 에이전트들과 직접 통신하여야 한다. 이 구조에서는 각 에이전트들이 독립적으로 협동을 위한 제어를 수행하게 된다. 조정자를 가지는 가장 일반적인 형태는 단일 조정자가 다수의 에이전트들을 통합하는 구조〈그림 1(b)〉[12]이다. 이 구조는 지식 자원을 통한 정보처리를 담당하는 하부

의 에이전트들과 이들 에이전트들을 관리하고 작업을 분배하며 결과를 통보해주는 조정자, 그리고 사용자와의 상호작용을 담당하는 에이전트의 세 계층으로 구성된다. 이 구조는 조정자를 갖지 않는 구조에 비하여 복잡한 시스템 구현이 가능한 반면 메시지 교환의 양은 증가한다. 또 다른 형태로는 여러 개의 조정자가 대등하게 연결되는 단순형(그림 1(c))[3][13]과 베이스 질의어를 이용한 방법, 해당 문제의 알고리즘을 이용하는 방법, 규칙기반(rule-based)[4], 퍼지기반(fuzzy-based)[14] 등이 있다.

2.2 다중 에이전트의 협동구조

다중 에이전트 시스템은 여러 개의 단순한 구성 에이전트들이 서로 협동하여 문제를 해결하는 협동 구



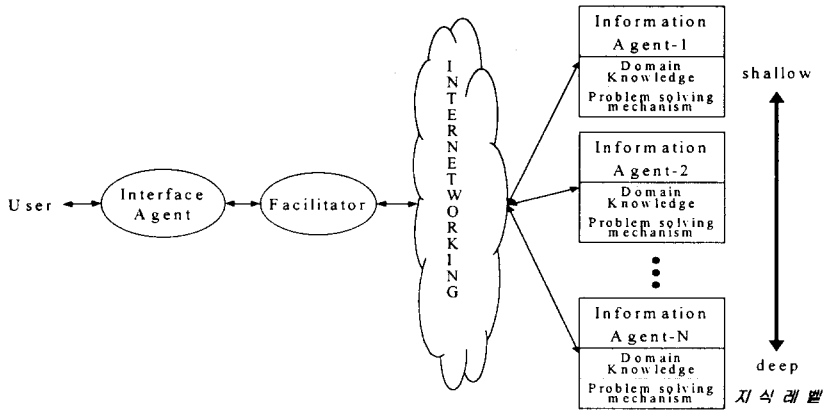
〈그림 1〉 다중 에이전트 구조

계층적으로 연결되어지는 계층형(그림 1(d))[6] 등이 있고 연결 방법에 따라 문제 해결을 위해 조정자가 결합되어지는 방법도 달라진다.

다중 에이전트를 구성하는 각 구성 에이전트는 기본적으로 자신의 영역(domain) 지식과 그 지식을 바탕으로 문제를 해결할 수 있는 처리기능을 가지며 이러한 정보를 자신의 능력(capability)에 맞게 모델링하게 된다. 구성 에이전트가 갖는 영역 지식은 어떤 지식을 다루고 있고 어떤 지식 표현 방법을 사용했는지에 따라 분류될 수 있다. 지식의 영역은 각 에이전트의 능력에 직접적으로 반영될 수 있는데 어떠한 영역 정보를 가지고 있느냐에 따라 에이전트가 해결할 수 있는 능력이 결정될 수 있다. 지식 형태는 파일, 데이터베이스[13], 지식베이스[9] 그리고 WWW 정보 등 다양한 타입[8]으로 표현되어질 수 있다. 각 구성 에이전트들이 문제 해결을 위해 적용되는 처리기법은 지식 표현 방법에 의존적이며 따라서 구성 에이전트가 갖는 지식 표현 방법에 따라 다른 문제 해결을 위한 방법이 적용될 수 있다. 예를 들면 단순히 데이터

조를 가지며, 이 때 문제를 해결하기 위해서 구성 에이전트들이 어떠한 방법으로 통합되어지는지에 따라 분류되어질 수 있다. 다중 에이전트에서 구성 에이전트를 통합하는 방법은 구성 에이전트들의 능력에 의존하며 일반적으로 각 에이전트가 문제 해결을 위한 부분 작업(partial task)을 수행하고 이들이 전체 문제를 해결하기 위해서 어떠한 형태로 결합하여 문제 해결을 시도한다. 또한 문제 해결을 위해 선택된 에이전트들이 한 팀으로 팀웍을 수행하거나 계층적으로 수행되어질 수 있는데 계층적인 경우는 하위 계층에서 수행되는 결과를 상위 계층에서 결합하는 방식의 통합 방법[13]을 사용한다. 수행 방법 면에서 관련 에이전트들은 병렬(parallel) 또는 순차(sequential)적인 방법[10]으로 수행될 수 있다. 병렬식 방법은 주어진 문제 해결에 필요한 에이전트들이 동시에 작업을 수행하는 반면 순차적 통합 방법은 해당 에이전트들이 순서적으로 작업을 수행하도록 제어하는 방식이다.

이러한 구성 에이전트들의 통합 방법에 따라 조정



〈그림 2〉 제안 시스템의 구조

자의 협동을 위한 제어 수준이 결정될 수 있다. 조정자는 기본적으로 구성 에이전트들의 수행을 조정하고 이들 간의 통신을 제어하는 역할을 수행하며 공유메모리(shared memory), 번역기(translator), 브로커, 매치메이커, 통합기(integrator), 중재자(mediator) 등의 역할을 수행하기도 한다. 또한 계획 생성(planning), 스케줄링(scheduling), 학습(learning), 충돌 해결(conflict resolution) 등의 기능도 가질 수 있다. 간단한 조정자는 단순한 공유 메모리로서의 기능만을 수행하기도 하고, 복잡한 시스템에서는 위에서 열거한 여러 가지 기능들을 복합적으로 포함함으로써 좀 더 지능적으로 작용하기도 한다[3][6][13]. 다중 에이전트 시스템에서 조정자의 역할은 구성 에이전트들의 능력과 그들이 통합되어지는 방법, 구조화되는 방법 등에 따라 요구되는 기능들의 양상이 달라지게 된다. 예를 들면, 각 구성 에이전트가 다른 능력을 가지고 문제 해결에 부분 작업으로서 역할을 수행하고 이들 간에 시간적 제한(time-constraints)을 가진다면 순차적인 통합이 가능해야 한다. 이러한 경우에 조정 에이전트는 구성 수행단위들에 대해 수행의 순서를 결정해주는 계획과정과 생성된 계획에 따라 에이전트간의 수행을 조절하는 기능 등이 필요하게 된다.

3. 지식레벨을 이용한 다중 에이전트 협동 정보시스템의 설계

3.1 지식레벨

인간의 사고를 분류함에 있어서 기억방식의 경우 장기 기억(long-term memory)과 단기 기억(short-term memory)으로 나누어 분석하는 경우가 많다. 인공지능에서도 이러한 기억 분류에 따라 메모리의 계층을 줄 수 있다. 단지 시간적인 관점을 떠나 지식의 경우에는 표면지식(shallow knowledge)과 심층지식(deep knowledge)으로 나누기도 한다. 표면지식은 단순히 조작적인 최종 결과만으로 표현되는 지식 형태로서 특정 영역의 문제에 대하여 경험에 의해 생성된 휴리스틱한 지식이다. 즉, 배경 속의 깊은 의미나 이유를 염두에 두지 않고 외적으로 나타난 결과에 따라 반응하는 지식 형태이다. 반면 심층지식은 표면지식의 상대적인 개념으로서 기본적인 이론, 원리, 공리, 사실 등을 망라한다. 따라서 이것은 표면지식의 타당성에 관한 내부적 원인 및 이유를 설명할 수 있는 지식을 말한다.

인간의 경우 표면지식과 심층지식의 연속적인 변환 과정을 통하여 사고가 이루어진다. 많은 경우에는 표면지식만으로 문제가 해결되지만 어떤 자극에 의하여 좀더 심층적인 지식이 불려나오고 때로는 오랜 시간 동안 깊은 사색에 빠질 수도 있는 것이다.

표면지식과 심층지식을 구현 관점에서 명확히 구분할 수 있는 방법은 현재로선 없다. 본 연구에서는 세밀도(granularity)에 따른 지식레벨이라는 다소 모호한 개념을 도입하여 이에 대한 한 가지 방법을 제시하여 보았다. 지식의 세밀도란 어떤 영역 지식에 대하여 얼마나 자세히 이를 기술할 수 있는가의 척도이다. 따라서 똑같은 지식처리기의 경우 더 많은 양의 데이터를 갖거나 한 레코드(혹은 사실)에 대하여 더 많은 애트리뷰트를 갖는 것으로 세밀도가 결정될 수

있다. 또한 지식처리기의 복잡도에 따라 그것이 얼마만큼 강력한가에 따라 세밀도가 정해질 수 있다. 일반적으로 세밀도가 낮을수록 속도는 빨라지는 반면 지식의 정밀성은 떨어지기 마련이다. 이러한 관점에서 보면 단순한 데이터베이스는 세밀도가 매우 낮고 수많은 규칙과 지식조정기능을 갖춘 지식베이스는 그보다 세밀도가 높다고 말할 수 있다. 본 연구에서 개별 에이전트들이 갖는 지식원에 대한 세밀도의 결정은 각 에이전트가 제공하는 자신에 대한 정보 - 지식표현 방법, 지식 형태, 애트리뷰트의 수, 사용 규칙의 수, 지식조정기능 포함 여부, 지식처리 메카니즘의 복잡성 등 - 에 의해 결정되어진다.

3.2 제안시스템의 기본구조

본 연구에서는 다양한 주제에 대한 여러 지식레벨의 지식원과 문제 해결 능력을 가지는 분산 개별 에이전트들이 서로 협동하여 정보를 처리하는 다중 에이전트 기반 분산 협동 정보처리 방법을 제시하였다. 제안 구조의 기본 모형은 〈그림 2〉와 같다.

전체 프레임워크는 각기 다양한 수준의 영역 지식과 이들 영역지식에 대해 독립적으로 특성화된 형태의 문제 해결 처리 기능을 가지고 있는 분산 정보 에이전트(information agent)들, 이들 구성 에이전트들 간의 수행을 조절하고 통제 또는 통합하는 조정 에이전트(facilitator), 그리고 사용자와의 인터페이스를 담당하는 사용자 인터페이스 에이전트(interface agent)로서 구성되는 단일 조정자를 갖는 계층적인 다중 에이전트 구조이다.

구성 에이전트들이 가지는 정보의 형태는 여러 가지 기준에 의한 특성을 가질 수 있다. 다루고 있는 주제, 사용된 지식 표현, 제공되는 정보의 형태(멀티미디어 정보, 웹 정보) 그리고 동일한 주제라고 할지라도 어떠한 관점에서의 파악한 정보인지 등에 따라 다양한 특성을 가질 수 있다. 본 프레임워크에서 정보 에이전트가 다루는 대상 지식원은 동일 주제에 대해 여러 지식레벨을 갖는 다양한 형태의 데이터들에 초점을 맞추고 있다. 이들은 데이터베이스, 지식베이스(shallow, deep KB)의 형태로 표현될 수 있고 멀티미디어 정보나 웹 정보 등을 포함할 수도 있다. 각 정보 에이전트는 다른 지식레벨의 다양한 형태의 지식원을 처리하기 위한 적절한 문제 해결 기능을 갖는데, 사용되는 방법은 영역 지식원의 형태에 따라 데이터베이스를 이용한 SQL 처리나 규칙 또는 퍼지 기반 등과 같은 다양한 추론 방법 그리고 여러 가지 기능이 연계된 복잡한 형태의 지식 관리 방법이 있을 수 있다.

제안 구조에서 각 구성 에이전트들 간의 조정기능은 조정 에이전트에 의해 수행되는데 조정 에이전트는 각 에이전트가 가지는 지식원의 지식레벨 정도에 따라 순차적인 접근 방법으로 문제를 해결할 수 있도록 구성 에이전트들을 제어한다.

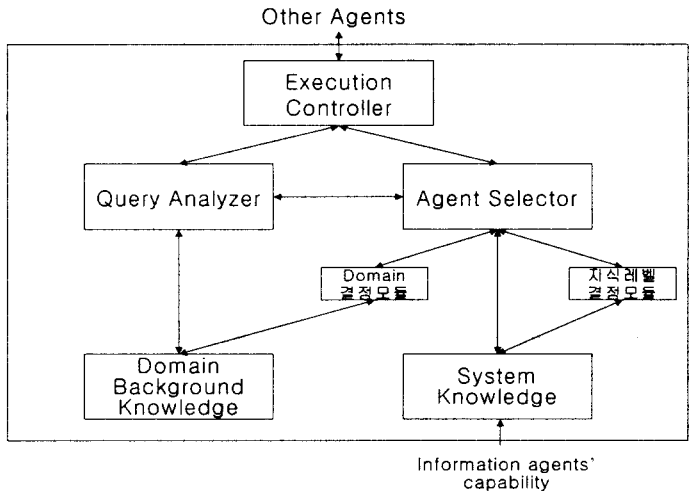
3.3 조정 에이전트

3.3.1 조정 에이전트의 구조

일반적으로 조정 에이전트가 협동 처리를 수행하기 위해서는 기본적으로 다음과 같은 작업이 필요하다. 우선 주어진 문제를 하나의 에이전트에 의해 수행 가능한 형태의 부분 문제로 분해하는 과정이다. 또한 조정 에이전트는 각 에이전트에 대한 능력 정보를 가지고 주어진 문제를 해결할 수 있는 적당한 에이전트를 선택하는 과정을 수행하게 되며 선택된 에이전트에 대한 수행 제어를 통해 문제를 해결한다.

제안 프레임워크에서 각 정보 에이전트는 도메인 지식을 포함한 자신의 정보(agent capability)를 제공하고 조정 에이전트는 이들 정보를 분산 정보 에이전트간 협동 수행을 제어하는데 사용한다. 사용자 인터페이스로부터의 질의(query)는 분석되고 도메인에 대한 배경 지식(Domain Background Knowledge, DBK)을 이용하여 문제 해결을 위해 요구되는 필수적인 정보를 추출하게 된다. 분석된 정보를 바탕으로 적당한 에이전트를 선택하고 선택된 에이전트를 활성화시킨다. 정보 에이전트는 주어진 질의를 자신의 스키마 정보를 바탕으로 에이전트 도메인 지식과 지식처리 메카니즘에 적당한 형태로 변환하여 질의에 대한 처리를 한다.

〈그림 3〉은 본 논문에서 제안하는 조정 에이전트의 구조를 나타낸다. 질의 분석기(Query Analyzer)는 인터페이스 에이전트로부터의 도메인 배경 지식(DBK)을 이용하여 질의를 분석한다. 에이전트 선택기(Agent Selector)는 질의에 대해서 DBK로부터 도메인(domain)을 결정하고 각 에이전트에 대한 정보를 포함한 시스템 지식(system knowledge)을 통해 지식레벨을 결정하여 질의를 수행할 적당한 에이전트를 선택한다. 시스템의 전체 수행과정에서 에이전트의 선택은 지식레벨이 낮은 것부터 순차적인 방법으로 제어되며 이러한 수행과정 및 다른 에이전트들과의 협동을 제어하는 역할은 수행 제어기(Execution



〈그림 3〉 조정 에이전트의 구조

Controller)가 수행한다.

3.3.2 조정 에이전트의 지식 및 역할

조정 에이전트가 분산 정보 에이전트의 협동 제어를 위해서 사용하는 주요 지식은 다음과 같다〈표 1〉.

Knowledge
- Domain Background Knowledge,
  - Concept(Object) Knowledge
  - Property(Attribute) Knowledge
  - etc.
- System Knowledge
  - Agent Knowledge
  - Information Source Knowledge
  - etc.

〈표1〉 조정 에이전트의 지식

Facilitator's Knowledge					
Domain Background Knowledge			System Knowledge		
Concept (Object) Knowledge	Property (attribute) Knowledge	etc.	Agent Knowledge	Information source Knowledge	etc.

구성하는 각 에이전트의 시스템 정보(System Knowledge)는 다음과 같은 내용을 포함한다. 이러한 시스템 지식은 각 에이전트의 지식레벨을 결정하고 질의에 적당한 에이전트 선택에 이용된다.

System Knowledge
- domain(sub_domain):
- knowledge_level: low/med/high
- knowledge_representation: DB/Rule/...
- knowledge_type: shallow/deep/...
- problem_solving_mechanism: SQL/rule inferencing/...
- schema:
- number_of_attributes:
- number_of_rules:
- self_learned: 지식조정기능포함여부
- agent_name:
- agent_site:

예를 들면, 정보 에이전트에 대한 시스템 지식으로서 다음과 같은 정보를 조정 에이전트가 사용할 수 있다.

System Knowledge(1)
- domain: animal
- knowledge_level: low
- knowledge_representation: relational_DB
- knowledge_name: animal
- problem_solving_mechanism: SQL
- schema: (class:string, name:string,

color:string, weight:numeric)
number_of_attributes: 4
self_learned: NO
agent_name: Agent1

System_Knowledge(2)
domain: animal
knowledge_level: med
knowledge_representation: rule_based
knowledge_name: animal.clp
problem_solving_mechanism:
rule_inferencing
schema: (class:string, name:string,
color:string, weight:numeric,
property:string)
number_of_attributes: 5
number_of_rules:
self_learned: NO
agent_name: Agent2

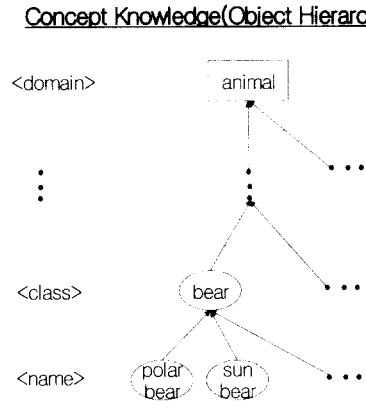
조정 에이전트가 시스템에 주어지는 질의를 분석하고 이에 적절한 정보 에이전트를 선택하기 위해서는 시스템 지식 이외에 도메인 배경 지식이 요구된다. 질의는 도메인 배경 지식을 통해 분석되므로 질의 형태와 도메인 배경 지식간에는 서로 밀접한 관련성이 있다.

본 제안 시스템에서 정보 검색 시 테스트되어지는 질의의 형태는 다음과 같다.

Query Type
Type 1. 〈name〉|〈class〉
Type 2. 〈property〉 of 〈name〉|〈class〉
Type 3. 〈property_adj〉 〈class〉

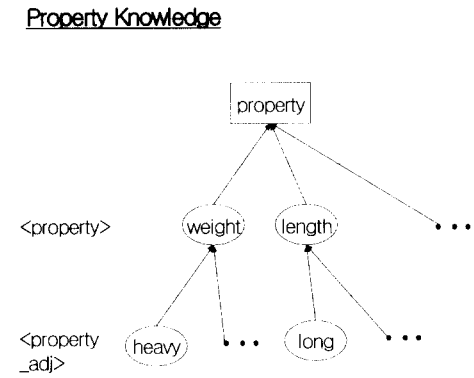
예를 들어 ***"heavy bear"***라는 질의는 기본적으로 *heavy + bear*로 파싱된 후 *heavy*는 〈property_adj〉에 *bear*는 〈class〉에 대응되는 정보로서 분석되어지는 Type 3에 해당되는 질의 형태를 나타낸다고 할 수 있다. 그리고 이런 질의는 다른 형태로 확장 가능하다.

주어진 질의에 대한 질의 분석을 통해서 해당 도메인 정보와 질의로부터 제공되는 기본 정보를 추출해 내기 위해서는 해당 도메인의 배경 지식이 필수적으로 요구된다. 도메인 배경 지식은 〈name〉, 〈class〉, 〈property〉, 〈property_adj〉 등에 대한 도메인 정보를 포함한다. 〈name〉, 〈class〉에 대한 정보는 도메인에 대한 개념 구조(concept hierarchy)를 나타내는 지식으로서 표현된다. 가장 상위레벨의 개념이 문제의 도메인을 나타낸다. *animal* 도메인에 대한 개념 지식은 〈그림 4〉와 같이 구성될 수 있다.



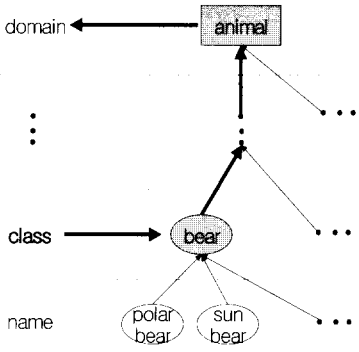
〈그림 4〉 도메인 배경 지식(개념 지식)

〈property〉, 〈property_adj〉에 대한 정보는 다음과 같은 형태〈그림 5〉로 구성되어질 수 있다. 일반적으로 개념이나 객체(concept or object)를 표현하기 위해서는 그 개념을 나타내는 기본적인 특성(property)을 갖게 되므로 개념 지식과 특성 지식은 서로 관계가 있다.

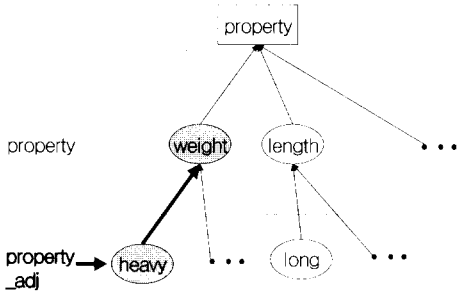


〈그림 5〉 도메인 배경 지식(특성 지식)

질의에 적당한 정보 에이전트의 선택은 질의가 요구하는 도메인 정보 추출과 정보 에이전트의 순차적인 제어를 결정하기 위한 각 정보원의 지식레벨을 통해서 이루어질 수 있다.



〈그림 6〉 개념 지식을 이용한 도메인 결정의 예



〈그림 7〉 특성 지식을 이용한 질의에 대한 기본 정보 추출의 예

질의가 요구하는 도메인 정보 추출은 DBK인 개념 구조(concept hierarchy)를 통해 얻어진다. 각 도메인에 대한 개념 구조를 가지고 있다면 하위레벨에서 매핑되는 정보로부터 가장 상위레벨의 개념(트리 구조상의 루트 노드)을 결정하는 것이 해당 정보에 대한 도메인을 결정하는 방법이 될 수 있다. 예를 들어, ***heavy bear***에서 *bear*는 도메인 개념 구조상의 〈class〉에 매핑되고 그 구조에서 가장 상위레벨의 개념이 도메인 정보가 된다. 이 경우는 *animal*이 질의에 대한 도메인으로 결정된다〈그림 6〉. 또한 질의를 처리하기 위해서는 질의에서 주어지는 기본적인 정보를 DBK를 이용하여 추출해야 한다. 예를 들면, ***heavy bear***에 대해 *heavy*는 〈property_adj〉를 나타내는 개념에 매핑되는데 바로 상위 레벨의 개념이 가장 근접할 수 있는 개념이므로 *heavy*는 〈property〉 *weight*의 특성을 나타내는 adjective임이 추출되어질 수 있다〈그림 7〉. 따라서 ***heavy bear***에 대한 질의로부터 domain은 *animal*이고, class는 *bear*, weight에 대한 특성정보가 *heavy*라는 좀 더 영역지식에 의존적인 형태의 기본 정보가 유도되어질 수 있다〈그림 8〉.

heavy bear
→ property_adj : *heavy*
class : *bear*
→ domain : *animal*
class : *bear*
weight : *heavy*

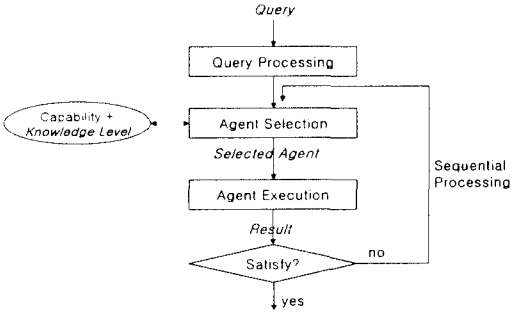
〈그림 8〉 (도메인 배경지식을 이용한) 질의 분석의 예

또한 적당한 에이전트의 선택을 결정하고 분산 정보 에이전트의 협동 제어에 사용되는 정보로서 본 논문에서는 지식레벨을 사용한다. 따라서 각 에이전트로부터 선전(advertise)되는 시스템 지식으로부터 지식의 세밀도를 결정하는 과정이 우선적으로 요구된다. 지식의 세밀도는 시스템 지식으로 제공되는 여러 지식원에 대한 정보 - knowledge_representation, knowledge_type, problem_solving_mechanism, number_of_attributes, number_of_rules, self_learned 등 - 에 따라 결정된다. 기본적으로 지식 표현 방법과 지식처리기가 복잡할수록, 한 레코드(혹은 사실)에 대하여 더 많은 애트리뷰트를 가지거나 규칙의 수가 많을수록, 학습기능을 통해 지식 조정이 가능할수록, 지식이 다루는 데이터 타입이 다양할수록 지식원의 세밀도가 높다고 할 수 있다. 이러한 관점에서 단순 데이터베이스는 세밀도가 매우 낮고 수많은 규칙과 지식조정기능을 갖춘 지식베이스는 그보다 세밀도가 높다고 할 수 있다.

3.3.3 조정 에이전트의 협동 작업

도메인 배경 지식 및 시스템 지식으로부터 질의에

대한 도메인과 구성 정보 에이전트의 지식레벨이 결정되면 이들 정보를 바탕으로 조정 에이전트는 질의 수행에 적당한 에이전트를 선택하게 된다. 질의로부터 결정된 도메인 정보와 각 에이전트의 시스템 정보를 통해 해당 도메인 정보를 가지고 있는 에이전트를 뽑아낸다. 도메인에 의해 필터링된 결과, 동일 주제를 갖는 여러 지식레벨의 지식원을 가지는 정보 에이전트들이 결정되어질 수 있다. 이들 구성 에이전트들은 동일 주제에 대한 문제 해결 능력을 가지고는 있으나 이들 각 지식의 깊이 정도가 다르므로 이들로부터 해결할 수 있는 문제 해결 수준이 다를 수 있다. 조정 에이전트는 각 에이전트의 능력 정보뿐만 아니라 지식레벨 정보를 고려하여 지식원의 세밀도 정도에 따라 순차적인 수행을 하도록 조정한다. 우선 가장 낮은 세밀도 정도를 가진 에이전트를 선택하여 수행하게 하고 사용자의 만족 여부에 따라 좀 더 깊은 레벨의 지식을 갖는 에이전트로 하여금 문제를 해결하도록 반복 처리하는 순차적인 방법의 수행을 조정 에이전트의 수행 제어기가 제어한다(그림 9). 앞에서 예를 든 Agent1과 Agent2는 둘 다 animal 도메인에 대한 정보를 포함하고 있다. 이 때 low한 knowledge_level을 갖는 Agent1을 선택하여 수행하고 knowledge_level에 따라 Agent2의 순으로 순차적인 반복 수행이 되도록 조정된다.



(그림 9) 조정 에이전트의 협동 작업

4. 제안 시스템의 구현

전체 프레임워크를 골격으로 다양한 지식레벨의 지식원을 갖는 다중 에이전트 기반 분산 협동 정보시스템의 간단한 테스트 시스템을 개발해보았다. 실제 구현에 사용된 적용 영역은 동물 관련 정보를 기반으로 하는 정보 검색 시스템이다. 동물 영역에 대한 데이터베이스, 규칙기반의 표면 지식(shallow KB), 그리고 심층 지식(deep KB)을 지식원으로 갖는 여러 지식레벨의 정보원과 각 정보원에 적당한 문제 해결 방법 - 데이터베이스 관리시스템, 규칙 기반, 퍼지 기반, 지식 관리 시스템 등 - 을 적용하여 대상 지식으로부터 해당 문제를 처리하는 정보 에이전트들을 구성하고 이들의 협업과정을 조정 에이전트가 제어하도록 하였다. 조정 에이전트가 사용하는 통합 및 제어 방법은 앞서 언급한 지식의 세밀도 레벨에 따른 순차적인 방법을 사용하였다. 지식의 세밀도 레벨이 높은 정보가 심층 지식에 해당하게 된다. 따라서 새로운 질의가 들어올 때 지식의 세밀도 레벨이 낮은 것부터 순차적으로 수행되도록 제어하였다.

4.1 구현 시스템의 지식레벨

분산된 정보 에이전트들은 도메인 정보에 대한 여러 지식레벨을 갖는 다양한 형태의 데이터들을 지식원으로 가지며 이러한 지식원에 적당한 처리기법을 제공한다. 주어진 질의로부터 수행에 적절한 정보 에이전트가 선택되어지면 조정 에이전트에 의해서 추출된 기본 질의 정보는 선택된 에이전트에 적당한 형태로 변형되어 처리된다.

구현 시스템에서 사용되는 지식원은 지식레벨에 따라 데이터베이스, 표면 지식베이스(shallow KB) 그리고 심층 지식베이스(deep KB)의 형태를 가진다. 지식레벨의 의미는 주어진 문제를 어느 정도의 깊이까지 해결할 수 있는지에 대한 정도를 말한다. 예로써 "***heavy bear***"라는 질의어가 들어왔을 때 각 레벨의 지식원에서 어떤 형태의 문제해결이 시도되는가를 알아보기로 하자.

(1) 데이터베이스

동물 관련의 데이터베이스를 지식원으로 갖는 정보 에이전트는 주어진 질의를 대응되는 SQL문으로 변환하여 데이터베이스로부터 원하는 정보를 제공하게 된다. relational_DB 형식의 데이터를 지식원으로 갖는 정보 에이전트의 시스템 지식과 질의 분석 결과의 기본 정보로부터 다음과 같이 질의에 대응되는 SQL 문을 만들 수 있다.

System_Knowledge
    domain: domain_name
    knowledge_level: level_information
    knowledge_representation: relational_DB

knowledge_name: table_name
schema: (attri1: , attr2: , ... , attrn:)
SQL Query
SELECT *
FROM table_name
WHERE 질의 분석 결과의 기본정보

앞의 예제인 "***heavy bear***"인 경우, 조정 에이전트에 의한 질의 분석의 기본 정보로부터 다음의 SQL 질의가 만들어져 처리되게 된다.

```
SELECT *
FROM animal
WHERE class=bear and weight=heavy
```

그러나 제공하는 스키마 정보와 SQL 질의가 부합되지 않는다면 질의에 대한 처리가 불가능하게 된다. 이 예의 경우, 데이터베이스 형태로서 제공되는 정보가 *bear*에 대한 무게 정보만 가지고 있고 *heavy*라는 개념이 존재하지 않기 때문에 *heavy bear*에 대한 위의 질문은 결과를 제공할 수 없게 된다. 이것은 사실 정보를 포함하는 데이터베이스로부터 특성에 대한 개념 정보를 추론할 수 없기 때문에 발생한다고 할 수 있다. 따라서 이러한 경우 재질의가 필요하게 된다. 재질의 확장은 다음과 같이 특성 구조(property hierarchy)를 일반화(generalization)하는 방법을 사용한다.

1) 특성 구조에서 상위의 개념으로 확장한다.
예를 들면 *weight*의 상위 개념인 *property*로 확장함으로써 다음과 같은 SQL 질의를재생성한다.

```
SELECT *
FROM animal
WHERE class=bear and property=heavy
```

2) 1)과 같은 방법으로 확장하지 못할 경우, 그 항목(term)을 삭제함으로써 일반화시킨다. 예를 들면, *weight*의 가장 상위 개념까지 확장해도 해결이 되지 않는 경우에는 property에 대한 항목을 삭제함으로써 일반화시킨다.

```
SELECT *
FORM animal
WHERE class = bear
```

그러나 이러한 질의의 일반화는 사용자가 의도하는 질문에 대한 결과와는 거리가 먼 *bear*에 대한 일반 정보를 제공함으로써 사용자의 요구에 미치지 못하게 된다. 이러한 문제는 데이터베이스에는 사실 정보만을 포함하고 사실로부터 얻어지는 개념 정보에 대한 추론 기능이 부재하기 때문에 어느 정도의 무게를 갖는 *bear*를 *heavy*하다고 자동으로 추론해 낼 수 없다는 데 있다.

(2) 표면 지식베이스(shallow KB)

규칙 기반의 표면 지식베이스를 갖는 정보 에이전트의 경우에는 기본적으로 추출된 질의 정보와 매칭되는 사실(fact)을 찾고 부합되지 않는 경우에 질의 정보를 일반화한 사실을 찾는다. 예를 들어 "heavy bear"의 경우 〈class〉가 bear이고 〈weight〉가 heavy인 사실이 존재하지 않으면 〈class〉가 bear이고 〈property〉가 heavy가 되는 사실을 찾는다. 이 경우에는 heavy한 개념을 결정할 수 있는 CLIPS 규칙 〈그림 10〉을 이용함으로써 어느 정도의 weight를 가지면 heavy하다고 하는 정보를 추론을 통해 제공할 수 있다. 따라서 weight 정보로부터 heavy하다는 개념을 추론하고 이를 통해 heavy한 특성을 갖는 bear에 대한 정보를 찾아낼 수 있다. 그러나 이 경우는 heavy한 개념의 결정이 단순하게 정의될 수 있는 경우에만 사용이 가능하다는 문제점을 갖는다. 왜냐하면 대개의 경우 heavy라는 개념은 상대적이고 상황에 따라 달라질 수 있기 때문이다.

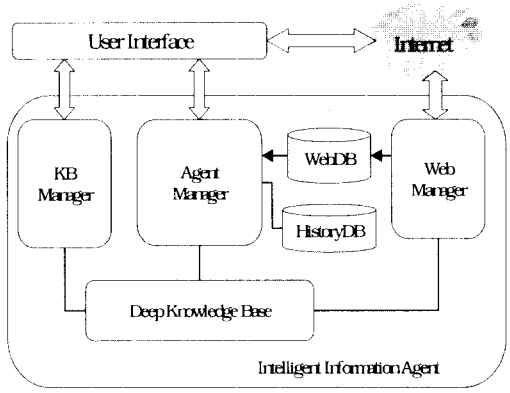
```
; bear template
(deftemplate animal
      (slot class)
      (slot name)
      (slot color)
      (slot weight)
      (slot property))

; rule-concept /heavy/
(defrule heavy-bear
      ?x <- (animal (class bear)
                    (name ?name)
                    (weight ?w))
      (test (> ?w 200))
=>
(modify ?x (property heavy))
```

〈그림 10〉 CLIPS에서 heavy 개념의 추론 규칙

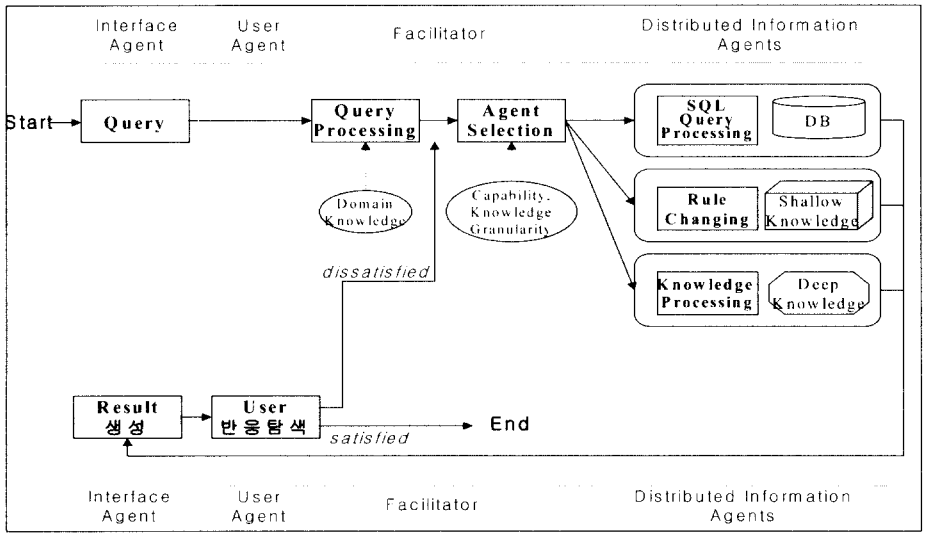
(3) 심층 지식베이스(deep KB)

본 연구에서는 심층 지식베이스로서 [1]에 기술된 동물에 대한 지식베이스를 예로 사용하였다. 이 시스템의 특징은 사용자의 피드백에 따라 퍼지 개념의 변화가 동적으로 이루어질 수 있다는 것이다. 이러한 심층 지식베이스를 지식원으로 갖는 정보 에이전트는 에이전트 관리자, 지식베이스 관리자, 웹 관리자로 구성되며〈그림 11〉, 여기서 사용되는 심층 지식베이스는 동물에 대한 전문적인 지식으로 동물에 대한 객체 정보, 속성 정보 및 지리 정보 그리고 단위 정보 등을 포함한다. 에이전트 관리자는 주어진 질의에 대해 처리 결과와 웹 문서를 제공하는 부분으로 상황에 따라 에이전트가 취해야 할 행동을 결정하는 역할을 한다. 에이전트가 취하는 행동으로는 질의의 추론이나 학습 등이 있을 수 있다. 학습 모듈은 사용자 에이전트로부터의 사용자 만족도에 대한 피드백으로서 영역 지식에 대한 학습을 수행하는데 학습 방법은 퍼지 원소 함수(fuzzy membership function)를 조정하여 퍼지 형용사의 범위를 조정하는 방법을 사용한다. 즉, 사용자의 반응에 대한 피드백 정보를 기반으로 퍼지 원소 함수의 범위를 재조정함으로 해서 형용사 연산자에 대한 개념의 적용을 학습하게 된다. 웹 관리자는 웹을 대상으로 적절한 웹 문서 제공을 위한 웹 DB의 구축과 유지 및 관리를 담당한다. 지식베이스 관리자는 지식베이스의 구축과 관리를 담당하고 있다. 이들은 서로 유기적인 관계를 가지고 작동하게 된다.

따라서 심층 지식베이스를 갖는 에이전트가 heavy bear라는 질의를 처리한 경우 사용자의 만족도에 따라 heavy한 개념을 학습함으로써 융통성 있는 정보 처리를 가능하게 하고 사용자의 의도에 적절한 결과를 제공할 수 있게 한다.



〈그림 11〉 deep KB를 갖는 정보 에이전트의 구조



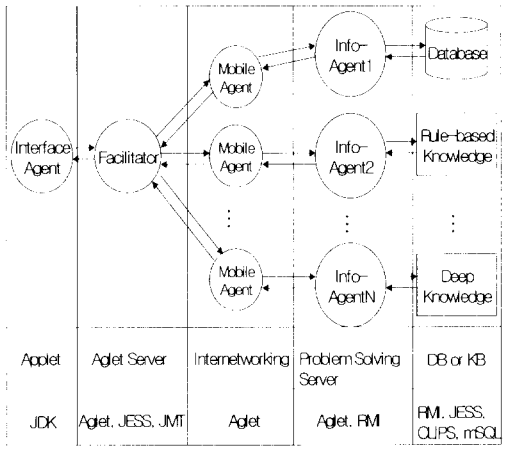
〈그림 12〉 프로토타입 시스템의 시나리오

4.2 전체 수행 과정

〈그림 12〉는 질의가 들어왔을 때, 관련된 에이전트들이 어떻게 문제를 해결하는지에 대한 과정을 단계적으로 보여준다. 구현 시스템에서의 지식 세밀도 레벨은 데이터베이스, 표면 지식베이스, 그리고 심층 지식베이스 순으로 가정했다. 따라서 새로운 질의가 들어올 때, 이 레벨 순서에 따라 순차적 수행을 가능하게 하고 사용자의 반응에 따라 반복적 제어를 수행한다.

4.3 구현 환경

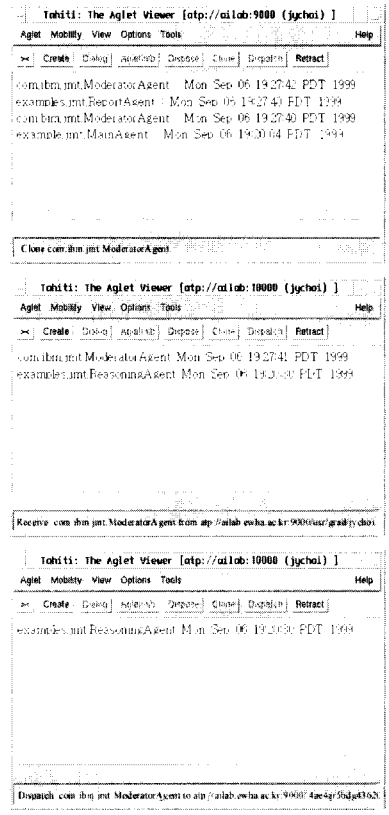
구현 시스템은 구성 에이전트 간 통신에 이동 에이전트(mobile agent) 패러다임을 적용하여 구현되었다. 이동 에이전트 전용 프로토콜을 지원하는 서버를 가진 호스트들 사이를 이동하는 이동 에이전트[2]를 이용하여 각각의 호스트에 상주하고 있는 정보 에이전트들로 하여금 대상 지식원에 대한 처리를 수행하게 하고 그 결과를 받아오는 역할을 수행하게 한다. 정보 에이전트는 각각의 호스트에 상주하는 에이전트로서, 이동 에이전트로부터 메시지를 전달받아 해당 지식을 기반으로 자신의 문제 해결 방식에 의해 처리된 결과를 다시 이동 에이전트에게 전해준다. 이때 질의에 적당한 정보 에이전트의 선택 및 이동 에이전트의 생성, 에이전트 간 통합 방법에 대한 결정 등 에이전트 간 수행 조정 및 제어는 조정 에이전트에 의해 처리된다. 시스템 개발 환경은 다음 〈그림 13〉과 같다.



〈그림 13〉 프로토타입 시스템의 구현

4.4 구현 결과

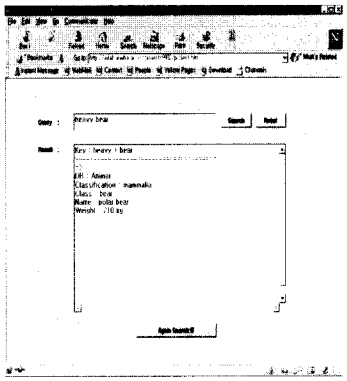
"***heavy bear***"라는 사용자의 질의로부터 조정 에이전트에 의해 관련된 정보 에이전트가 결정되고 선택된 정보 에이전트의 활성화를 위해서는 해당 호스트로 보내질 이동 에이전트가 생성되어야 한다. 다음 〈그림 14〉는 선택된 정보 에이전트를 수행시키기 위해서 이동될 이동 에이전트를 생성하고 이동, 디스패치(dispatch)되는 과정을 나타낸다.



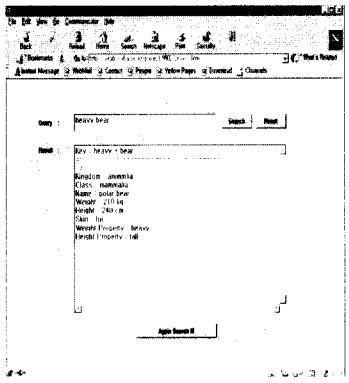
〈그림 14〉 이동 에이전트 생성, 이동 및 디스패치 과정

다음의 결과 화면〈그림 15〉는 다른 지식레벨의 지식원에 대해서 적당한 처리를 수행하는 분산된 개별 정보 에이전트가 "***heavy bear***"에 대한 질의 처리를 수행한 결과이다. 각 결과 화면은 데이터베이스, 표면 지식 그리고 심층 지식을 영역 지식으로 갖는 에이전트에 의한 결과를 나타낸다. 각 에이전트들은 자신의 정보로부터 *heavy bear*에 대한 관련 정보를 처리한다. 이 때 조정자는 각 지식원의 지식레벨에 따라 에

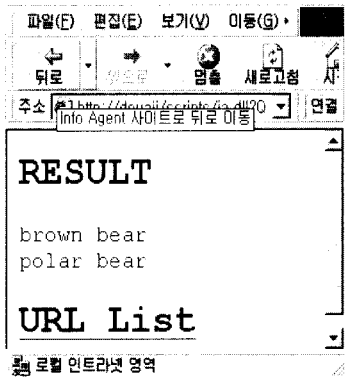
이전트의 수행이 순차적으로 진행되도록 제어한다. 각 에이전트의 결과는 이동 에이전트를 이용하여 조정 에이전트로 전달되고 인터페이스 에이전트를 통해 제시된다.



(a) DB로부터의 결과



(b) shallow-KB로부터의 결과



(c) deep-KB로부터의 결과

〈그림 15〉 분산 정보 에이전트의 수행결과

5. 결론(Conclusion)

인터네트 기술의 발달로 인하여 분산된 다양한 형태의 정보를 효과적으로 처리할 수 있는 방법들이 요구되고 있다. 본 논문에서는 이러한 문제를 다중 에이전트 패러다임을 이용하여 풀어보고자 하였다. 분산된 다양한 지식원을 에이전트화 하여 이들을 어떻게 조정하는 것이 적당한가에 대해서는 지식원의 상황에 따라 달라지기 때문에 방대한 연구를 요할 것이다. 본 연구에서는 한 가지 주제에 대하여 여러 개의 지식원이 존재하고 이들의 세밀도가 서로 다른 경우 이를 처리하는 방법에 대하여 살펴보았다.

한 가지 주제에 대하여 여러 개의 지식원이 제공되는 경우 이를 병렬 혹은 순차적으로 처리할 수 있는데 본 연구에서는 사용자의 요구에 대하여 간단한 지식처리로부터 시작하여 사용자의 만족도에 따라 순차적으로 더 복잡한 처리를 제공하는 방법을 제안하였다. 제안 시스템은 인지과정의 초점변환을 구현하는 한 가지 방법을 제시한다고도 볼 수 있다. 이 방법의 타당성을 보이기 위하여 세 가지 형태의 대표적인 지식원, 즉, 단순한 데이터베이스, 전문가시스템 형태의 지식 베이스, 그리고 퍼지 기능을 갖춘 대규모 지식베이스를 망라하는 형태의 다중 에이전트 기반 분산 정보 시스템에 대하여 테스트 시스템을 구현하여 보았다.

향후 본 연구를 바탕으로 조정자의 역할을 다양하게 확장할 수 있도록 에이전트 구조의 특성을 분석하여 구성 에이전트의 정보로부터 적절한 기능을 연관시킬 수 있는 형태의 프레임워크로 발전시킬 계획이다.

참고문헌(References)

[1] 이용현, 변영태, 구연건, "정보통신망에서 지능형 정보 에이전트와 특정 영역에서의 구현", 박사학위 논문, 홍익대학교 전자공학과, 1999.

[2] 최지영, 박승수, "이동 에이전트를 이용한 분산 전문가 시스템 개발환경의 설계 및 구현", 석사학위 논문, 이화여자대학교 전산학과, 1997.

[3] Bayardo, R., et al., "InfoSleuth: Agent-Based Semantic Integration of Information in Open and Dynamic Environments", Technical Report, MCC, 1997.

[4] Cengeloglu, Y., "DYNACLIPS (DYNAmic CLIPS) A Dynamic Knowledge Exchange

Tool For Intelligent Agents", The Third CLIPS Conference at NASA's Johnson Space Center, September, 1994.

[5] Cunnigham, P., Evans, R., "Software Agent : A review", http://www.cs.tcd.ie/Breda.Nangle/iag.html, 1997.

[6] Garcia-Molina, H., et al., "The TSIMMIS approach to mediation: data models and languages", Journal of Intelligent Information Systems, vol.8, pp117-132.

[7] Genesereth, M., Katchpel, S., "Software agents", Communications of the ACM, Vol. 37, No. 7, 1994.

[8] Jacobs, N., Shea, R., "the Role of Java in InfoSleuth: Agent-based Exploitation of Heterogeneous Information Resources", IntraNet96 Java Developers Conference, 1996.

[9] Mayfield, J., Finin, T., "The Cycic Friends Network: getting Cyc agents to reason together", http://www.cs.umbc.edu/www/lait/papers/

[10] Minami, K. and Suzuki, T., "JMT(Java-Based Moderator Templates) for Multi-Agent Planning", OOPSLA'97 Workshop, Java-Based Paradigms for Agent Facilities, 1997.

[11] Nwana, H., "Software Agents : An Overview", knowledge Engineering Review, Vol. 11, No 3, pp. 1~40, 1996.

[12] Rieken, D., "M : An Architecture of integrated Agents", Communications of the ACM, Vol. 37, No. 7, 1994.

[13] Seug, D., Sycara, K., "Cooperative Intelligent Software Agents", http://www.cs.cmu.edu/~softagents/

[14] Song, H., Franklin, S., "SUMPY: A Fuzzy Software Agent", Proceedings of the ISCA Conference on Intelligent Systems, June 1996.

[15] Wooldridge, M., Jennings, N., "Intelligent Agents : Theory and Practice", Knowledge Engineering Review, 1994.