HTTP 클라이언트/서버에 기반한 스케쥴링 비서 에이젼트 시스템의 설계 및 구현

박 창 현† · 정 호 열††

요 약

최근 인터넷 및 정보통신 기술이 급속하게 발달함에 따라서 네트워크를 기반으로 한 응용 소프트웨어 및 네트워크 관련 기술들이 많이 연구, 개발되고 있다. 이와 관련하여 과거 인공지능 분야에서 많이 연구되어 왔던 에이젼트 기술이 최근의 네트워크 기술을 기반으로 하여 다시 활발한 연구가 진행되고 있다. 본 논문에서는 일반 비서업무 중에서 상사의 스케쥴 관리를 대행할 수 있도록 하는 비서 에이젼트 시스템의 설계 모델을 제시하고 그 구성 기법을 기술한다. 본 논문에서 기술하는 비서 에이젼트 시스템은 데이터베이스 시스템과 지식기반 시스템을 포함하고 있으며, 이들은 각 비서 에이젼트와의 상호 협력하여 효율적인 스케쥴 자료 처리 능력 및 스케쥴 자료에 대한 판단 능력을 부여한다. 본 논문의 비서 에이젼트 시스템은 기존의 그룹 스케쥴링 또는 개인 스케쥴 관리 프로그램과는 달리 상사-비서-예약자 사이에서의 다양한 상호작용을 통하여 실제 비서와 유사한 동작에 의해서 운용되는 것을 보인다.

Design and Implementation of a Scheduling Secretary Agent System Based on HTTP Client/Server Mechanism

Chang-Hyeon Park† · Ho-Youl Jung††

ABSTRACT

Recently, according to the rapid development of internet and communication technologies, a lot of works based on network techniques have been developed. In relation to this trend, the agent systems that had been studied in the early AI have been being studied greatly in association with the network techniques. This paper presents a design model of a secretary agent system in which each secretary agent can manage the schedules of her/his superior, and the descriptions about the implementation of the secretary agent system. In the presented secretary agent system, a database system and a knowledge-based system are included and cooperated with each secretary agent to provide the ability of manipulating lots of schedule data and making decisions on them. This paper also shows that the presented secretary agent system can behavior like a real secretary through the various superior-secretary-meeting_requester interactions, which is different from the group scheduling programs or personal scheduling programs.

※ 본 논문은 한국과학재단의 박사후 해외연수 지원에 의한 것임.
† 종신회원 : 영남대학교 컴퓨터공학과 교수
†† 정 회 원 : 영남대학교 전자정보공학부 교수
논문접수 : 1998년 12월 18일, 심사완료 : 2000년 2월 15일

1. 서 론

에이전트에 대한 정의는 여러 학문 분야에서 다양하게 언급되고 있으나, 대체적으로 다른 대상을 위해 행동하는 객체로 정의하고 있다[12]. 컴퓨터 분야에서는 계산적인 객체(computational entity)로서의 에이전트를 언급하면서, 다른 객체를 위해서 행동하는 객체, 적절한 반응적(reactive) 행동과 순응적(proactive) 행동을 나타내며, 이동성(mobility) 및 협력성(cooperation)을 보이면서 행동하는 소프트웨어 시스템을 에이전트 시스템으로 언급하고 있다[4, 5, 6, 8, 12]. 에이전트 시스템의 종류는 대개 지능형 에이전트와 이동형 에이전트로 대별되는데, 지능형 에이전트는 특정의 작업을 위해서 지식을 이용하는 정적인(static) 대상을 의미한다[4, 12]. 이는 과거의 인공지능 연구에서 주류를 이루던 것으로, 그 연구의 대부분이 인간의 능력을 모방할 수 있는 지적인 객체를 생성하려는 노력으로서, 이는 에이전트 개념의 한 부분인 특정 객체의 할 일을 대행하는 객체를 형성하려는 노력과 상통한다. 반면에 이동형 에이전트는 네트워크를 통해 여러 시스템을 방문하면서 다양한 작업을 자치적(autonomous)으로 수행할 수 있는 동적인 시스템을 지칭하고 있다. 최근의 대부분의 에이전트 시스템에 대한 연구는 이와 같은 이동형 에이전트에 집중되고 있다[5, 6, 8, 10, 12].

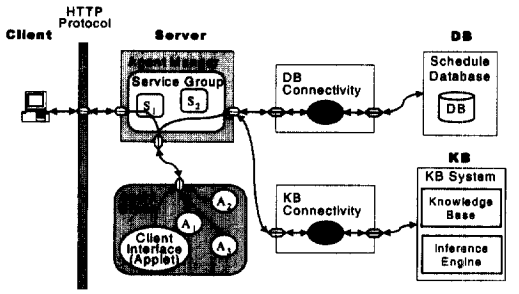
본 논문에서는 통상의 비서업무 중에서 상사의 스케쥴을 관리하는 업무를 대행하는 스케쥴링 비서 에이전트 시스템 모델을 설계하고 이의 구성 기법을 기술한다. 컴퓨터를 이용한 통상의 스케쥴 관리 프로그램들은 크게 두 가지로 나뉠 수 있는데, 하나는 캘린더 프로그램을 이용한 개인 스케쥴 관리 프로그램들이며, 다른 하나는 Lotus Organizer, MS Schedule+, Microsystems Software CaLANdar 등과 같은 그룹 스케쥴링 프로그램[3]들이다. 개인 스케쥴 관리 프로그램들은 단지 사용자의 스케쥴을 저장하고 지정된 시간에 사용자에게 확인시켜주는 것이 주요기능이며, 그룹 스케쥴링 프로그램들은 여러 사용자의 스케쥴 프로그램을 네트워크로 연결하여 다수 인원의 스케쥴을 일괄적으로 처리하여 특정의 시간대를 도출하여 그룹 미팅 등에 활용하고 있다. 본 논문의 비서 에이전트 시스템은 에이전트의 이동성, 협력성 등을 이용하여 비서 에이전트가 상사 스케쥴 관리뿐만 아니라 상사와 면담 예약자 사이에서 다양한 상호 작용 및 스스로의 판단 능력을 통하여 실제 비서의 동작과 유사하게 동작할 수 있는 환경을 제공한다. 이를 위해서 본 논문의 비서 에이전트 시스템은 데이터베이스 시스템과 지식기반 시스템을 포함하여 효율적인 스케쥴 자료 처리 및 스케쥴 자료에 대한 판단 능력을 부여하며, 비서 에이전트를 이동 코드로 구성하여 네트워크 상에서 효과적으로 이동할 수 있도록 한다.

본 논문의 비서 에이전트 시스템에서 서버 및 클라이언트에 위치한 각 에이전트는 자바 가상기계(Java Virtual Machine; JVM)[7, 13]에서 동작되며, 에이전트와 서버 사이의 전송 기법은 HTTP에 의한 클라이언트-서버 기술을 이용한다. 비서 에이전트를 위해서 자바를 사용하는 것은 네트워크 환경에서 에이전트들의 이동성(mobility) 능력을 매우 용이하게 부여할 수 있기 때문이다. 이러한 환경 하에서 사용자(상사 및 면담 예약자)는 필요시 서버에 요구만 하면 시간과 장소에 관계없이 해당 비서 에이전트가 전송되어 필요한 작업을 처리할 수 있다. 이것은 최근 인터넷의 대량 확산으로 인해서 자바 가상기계 또한 급속히 확산되고 있으며, 이미 모든 컴퓨터에서 사용되고 있는 웹 브로우저(browser)들도 모두 자바 가상기계를 포함하고 있기 때문이다.

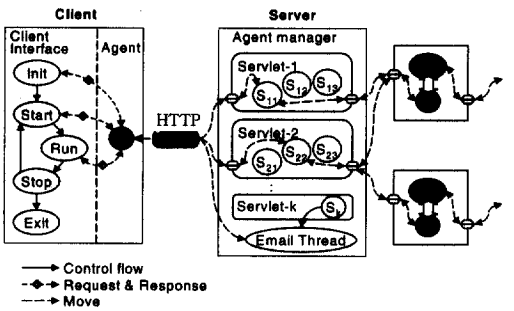
2. 비서 에이전트 시스템 모델

본 논문에서 기술하는 비서 에이전트 시스템은 클라이언트/서버/데이터베이스 시스템의 3단 모델(3-tier model)[9]과 클라이언트/서버/지식기반 시스템의 3단 모델을 혼합한 구조를 기반으로 한다. 사용자가 특정의 스케쥴 작업을 요청하면, 해당 비서 에이전트는 사용자의 요청을 가지고 서버로 이동하여 서버에서 제공하는 다양한 서비스를 이용하여 요청된 작업을 완수하고, 그 완수 결과를 가지고 원래의 위치로 되돌아온다. 본 논문의 비서 에이전트 시스템은 (그림 1)에서 보인바와 같이 에이전트 그룹, 에이전트 관리자, 데이터베이스 시스템 연결부, 지식기반 시스템 연결부, 스케쥴 데이터베이스, 지식기반 시스템으로 구성되며, 각각에 대한 설명은 이 장의 각 절에서 주어진다.

(그림 2)에서는 클라이언트 상의 비서 에이전트와 서버 상의 에이전트 관리자 사이의 상호 통신 관계를 자세히 보여주고 있다. (그림 2)에서 보인바와 같이, 사용자의 요청 시, 서버의 에이전트 그룹에 상주하던



(그림 1) 비서 에이전트 시스템의 설계 모델



(그림 2) 에이전트와 에이전트 관리자 사이의 통신

해당 비서 에이전트는 클라이언트 인터페이스와 동반하여 요청된 클라이언트 위치로 이동하여 사용자의 작업을 기다린다. 클라이언트 상의 비서 에이전트는 클라이언트 인터페이스를 통하여 사용자로부터 작업을 접수하고, 이 작업을 가지고 서버로 이동한다. 서버 상의 에이전트 관리자는 각 비서 에이전트의 작업을 처리하기 위해서 서비스 그룹 내에 여러 가지의 서블릿(servlet)[13]을 대기해 두고 있다. 서버로 이동해 온 비서 에이전트는 서블릿의 서비스 루틴들을 이용하여 작업을 처리하고는 그 결과를 가지고 원래의 클라이언트로 되돌아와서 클라이언트 인터페이스를 통하여 사용자에게 결과를 보여준다.

2.1 에이전트 그룹

에이전트 그룹은 상사의 스케줄이나 면담 예약을 처리할 수 있는 비서 에이전트와 클라이언트 인터페이스의 쌍들을 포함하고 있다. 사용자가 비서를 필요로 하면, 그 사용자에 해당되는 비서 에이전트는 클라이언트 인터페이스와 함께 해당 클라이언트로 이동한다. 클라이언트로 이동한 비서 에이전트는 클라이언트 인터페이스를 통해 사용자로부터 작업을 의뢰받고는 그 작업을 가지고 다시 서버로 이동한다. 서버로 이동한 에이전트는 작업을 처리하고 그 결과를 원래의 클라이언트 위치로 되돌아와서 클라이언트 인터페이스를 통해 사용자에게 보여준다.

본 논문에서의 비서 에이전트는 자바 객체로 구성되며 소속 ID, 요구 명령, 요구 명령 파라메터, 객체 저장소 및 실행 루틴들을 포함하고 있다. 소속 ID는 현재의 에이전트가 어떤 사용자에게 소속되어 있는지를 나타내며, 요구 명령은 에이전트에게 맡겨진 작업을 표시하고, 요구 명령 파라메터는 작업 수행에 필요한 선택 사항들을 가지고 있다. 객체 저장소는 작업 수행에 필요한 데이터나 작업 처리 결과를 간직할 수 있도록 하는 가변 용량의 임의의 객체를 지칭한다. 실행 루틴은 에이전트 생성이나 소멸 시에 동작되는 루틴을 포함하고 있다. 비서 에이전트 객체는 자바 가상기계 상의 객체 전송 방법인 직렬화 코드(serializable code) [7, 11, 13]로 구현되어, 클라이언트나 서버 또는 다른 위치로 용이하게 이동할 수 있도록 한다.

(그림 2)에 보이듯이 클라이언트에 이동한 클라이언트 인터페이스는 초기화 단계(init), 기동 단계(start), 실행 단계(run), 중지 상태(stop), 종료 단계(destroy)의 여러 단계들을 거치게 되는데, 각 단계마다 필요시 비서 에이전트에게 작업을 요구하고 결과를 받게 된다. 초기화 단계에서는 서버의 지역시간대(time zone)를 요구할 수 있으며, 기동 단계에서는 초기 화면 구성에 필요한 사용자의 초기 스케줄 정보를 요구할 수 있다. 실행 단계에서는 사용자 상호 작용이 계속적으로 발생하는 단계로서, 사용자의 스케줄 저장, 조회 및 예약 작업 등의 다양한 요구를 처리하는 단계이다. 중지 상태나 종료 상태가 되는 경우 이를 서버에게 알리는 역할도 비서 에이전트가 담당하게 된다.

2.2 에이전트 관리자

에이전트 관리자는 비서 에이전트가 클라이언트로부터 가져온 작업을 처리할 수 있도록 하는 환경을 제공한다. 만일, 특정 사용자가 자신의 비서 에이전트를 필요로 하면, 에이전트 관리자는 그 사용자에 해당되는 비서 에이전트와 클라이언트 인터페이스를 사용자에게 전송시키고는 비서 에이전트가 사용자의 작업을 가져오기를 기다린다. 비서 에이전트가 특정의 작업을 서

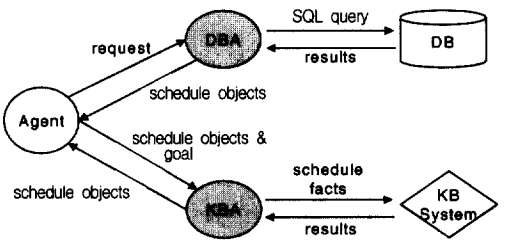
버로 가져오면 에이전트 관리자는 그 작업에 관련된 서블릿[13]을 초기화시킨 후(한번 초기화되면 항상 대기상태로 존재) 그 작업을 처리할 수 있는 서비스 루틴(메쏘드)을 호출하여 비서 에이전트가 작업 결과를 클라이언트로 가져갈 수 있도록 한다. 에이전트 관리자는 특정의 사용자 요구 작업을 처리할 수 있도록 하는 서비스 루틴들을 포함하는 다양한 서블릿들을 포함하고 있다. 비서 에이전트의 요구를 처리하기 위한 서블릿으로는 사용자 정보 처리, 초기 정보 처리, 스케줄 정보 처리, 예약 정보 처리, E-mail 작업 처리 등을 위한 서비스 루틴을 제공하는 서블릿들이 있다. 사용자 정보 처리는 비서 에이전트 호출시 사용자 확인을 위한 것이며, 초기 정보 처리는 클라이언트 인터페이스의 초기화 및 시작 단계에서 기본적으로 필요한 사항을 요구하는 작업을 처리하기 위한 서블릿이다. 스케줄 정보 처리나 예약 정보 처리는 클라이언트 인터페이스의 실행 단계에서 사용자에 의해서 발생되는 다양한 요구 사항, 즉 스케줄 저장 및 조회, 예약 처리 등에 필요한 서비스 루틴들을 제공한다. E-mail 작업 처리는 스케줄 상기(remind)나 예약 작업 처리 결과를 사용자에게 통보하기 위한 것이다.

(그림 2)에 보이듯이 에이전트 관리자는 모든 서블릿을 초기화시켜 항상 대기(ready) 상태로 있게 하는데, 특정 비서 에이전트로부터 요구가 들어오면 대기상태의 서블릿이 가지고 있는 서비스 루틴을 실행시켜 요구를 처리하고 응답을 전달한다. 이 때, 만일 데이터베이스 작업이나 지식기반 시스템의 추론 기능이 필요한 경우에는 해당 에이전트를 데이터베이스 시스템 연결부(DB connectivity)나 지식기반 시스템 연결부(KB connectivity)로 이동하여 작업을 처리하도록 한다. 본 논문의 에이전트 관리자는 JSDK 2.0(Java Servlet Development Kit)[13]으로 구현된다.

2.3 데이터베이스 및 지식기반 시스템 연결부

비서 에이전트 시스템은 데이터베이스 시스템의 저장 능력과 지식기반 시스템의 추론 능력을 이용함으로써 그 유용성이 증대될 수 있다. 그러나 에이전트 시스템이 다루는 스케줄 데이터는 이들 두 시스템들이 서로 다른 자료구조를 가지기 때문에 직접적으로 이용될 수 없다. 데이터베이스 및 지식기반 시스템 연결부는 이러한 자료구조의 차이를 제거할 수 있다.

데이터베이스 시스템 연결부의 DB 에이전트(DBA)는 비서 에이전트와 통신하면서 작업을 전달받아 이를 SQL 문장으로 변환하여 데이터베이스 시스템에 작업 처리를 의뢰하고, 처리된 결과를 비서 에이전트를 위한 스케줄 데이터로 변형하여 전달해 준다. 또한 DBA는 이와 반대로 사용자가 새로이 입력한 스케줄 데이터를 데이터베이스로 저장할 수 있다. 이 과정에 대한 다이어그램은 (그림 3)에 주어져 있다. 본 논문의 데이터베이스 시스템 연결부는 JDBC(Java Database Connectivity)[9, 13]를 이용하여 구현된다.



(그림 3) DBA와 KBA의 자료 변형 다이어그램

지식기반 시스템 연결부의 KB 에이전트(KBA)는 비서 에이전트가 가져온 특정의 목표와 스케줄 데이터들을 지식기반 시스템의 사실 표현으로 변형하여 지식베이스에 저장하고 지식기반 시스템의 스케줄 작업을 위한 규칙들을 동작시켜 추론을 진행시킨다. 추론이 완료되면 KBA는 추론 결과를 스케줄 객체로 변형하여 비서 에이전트에게 전달해 준다. 이의 변형 과정은 (그림 3)에 주어져 있다. 지식기반 시스템 연결부는 C로 구현된 지식기반 시스템과 Java로 구현된 비서 에이전트와의 연결을 위해 JNI(Java Native Interface)[13]를 이용하여 구현된다.

2.4 스케줄 데이터베이스

스케줄 데이터베이스는 스케줄 자료, 사용자 등록 정보, 예약 자료 등에 관련된 테이블로 구성되는 관계형 데이터베이스이다. 본 논문의 비서 에이전트 시스템을 위한 데이터베이스 시스템은 자료 관리 작업을 위해서 많이 이용되고 있는 MySQL[14] 데이터베이스 시스템을 채택한다. 본 논문의 스케줄 데이터베이스는 <표 1>에서 보이는 바와 같이 스케줄 테이블, 사용자 테이블, 예약 테이블로 구성된다.

스케줄 테이블의 구성 필드에서 'uID'는 현재 비서

〈표 1〉 스케줄 데이터베이스의 테이블

Schedule Table

uID	fromTime	toTime	status	kind	weeklyDay	validDate	remType	remind	···

User Table

uID	name	tel_no	email	room_no	passwd	memo	···

Reservation Table

toWhom	fromTime	toTime	name	email	position	comments	···

에이전트의 사용자인 상사의 등록 사용자명으로 현재 레코드의 스케줄에 대한 권한을 가진 사용자임을 의미한다. 'fromTime'과 'toTime' 필드는 스케줄의 시작 시간과 종료 시간을 나타내는 것으로서, 대정수(long integer) 자료형을 사용한다. 이는 GMT 기준으로 1970년 1월 1일 0시를 기준으로 현재 시간까지를 밀리초(millisecond) 단위로 나타낸 것이다. 이러한 시간 표현은 본 논문의 에이전트 시스템을 구성하는데 있어서 시간과 날짜에 관련된 모든 자료에서 사용되는 표현이다. 'status'는 현재 스케줄이 정식 약속인지 아니면 앞으로의 면담을 위해 미리 비워둔 시간인지를 나타낸다. 미리 비워둔 스케줄은 예약자의 예약 때 우선적으로 배정될 수 있는 스케줄을 의미한다. 'kind'는 이 스케줄이 한시적(plain), 주간(weekly) 또는 매일(daily)의 스케줄인지를 나타낸다. 'weeklyDay'는 주간 스케줄의 경우에 스케줄이 있는 요일을 정수로 표시하며, 'validDate'는 매일 또는 주간 스케줄의 경우에 유효 날짜를 대정수로 나타낸다. 'remType'과 'remind'는 스케줄 시간을 사용자에게 상기시켜주기 위한 필드로서, remType은 상기 시간의 종류(예를 들면, 스케줄 시간의 30분전 또는 하루 전에 상기시킴)를 나타내며, remind는 그에 해당하는 시간 간격을 밀리초로 나타내는 필드이다.

사용자 테이블은 각 상사의 사용자명, 이름, 전자우편 주소, 전화번호, 암호 등의 각 사용자의 개인 등록 정보를 저장하는 테이블이다. 이 테이블에서 'email' 필드는 비서 에이전트가 해당 상사(uID)에게 특정 스케줄을 상기시킬 때, 또는 예약 스케줄에 대한 확인 작업을 위해서 메시지를 보낼 전자우편 주소이다.

예약 테이블은 각 상사에게 주어지는 모든 예약들을 임시로 저장하는 테이블로서, 예약자에 대한 정보 및 예약 내용, 예약 시간 등의 정보를 저장한다. 이 테이블에서 'toWhom' 필드는 이 예약자가 누구에게 예약을 하는지를 나타내기 위한 것으로서, 상사의 사용자명이 주어지며, 'fromTime'과 'toTime'은 스케줄 테이블과 마찬가지로 예약 시작 및 종료 시간에 대한 대정수 표현이다. 그 다음의 'name', 'email', 'position' 등의 필드는 예약자에 대한 정보이며, 특히 'email' 필드는 예약자의 전자우편 주소로서, 이 예약에 대한 상사의 확인(승인 또는 거부)을 비서 에이전트가 받아서 이 예약자에게 전달해 준다.

2.5 지식기반 시스템

지식기반 시스템은 비서 에이전트 시스템 내에서 각 에이전트가 스케줄 자료의 시간에 대한 판단 정보를 추출할 수 있도록 추론 능력을 부여하기 위한 시스템으로, 스케줄 정보를 이용한 스케줄에 대한 시간적인 정보를 추출하는 것이 목적이다. 본 논문의 비서 에이전트 시스템에서의 지식기반 시스템은 객체지향 지식표현과 규칙기반으로 운용되는 HEXPERT[2] 시스템을 이용한다. HEXPERT의 지식베이스는 사실베이스와 규칙베이스로 구성되는데, 사실베이스는 대상(object), 상태(situation) 등의 성질(property) 또는 속성(attribute) 등을 객체 지향형 표현으로 나타내는 선언적 기술들의 집합이며, 규칙베이스는 대상, 상태 등에 대한 일반적인 법칙을 조건부-실행부의 형태로 나타내는 규칙들의 집합이다. HEXPERT의 추론엔진은 효율성이 널리 알려진 Rete 매칭 알고리즘[1]을 객체 지향형 지식을 처리할 수 있도록 확장한 전방향 추론 기법, 결론 또는 목표를 사실로 가정하고 역방향으로 추론하여 결론 또는 목표를 증명하는 역방향 추론 기법, 그리로 이 두

기법을 상황에 따라 적절히 혼합하여 사용하는 혼합형 추론 기법을 모두 가지는 통합형 추론 엔진이다. HEXPERT의 구조 및 지식표현에 대한 자세한 설명은 [2]를 참조할 수 있으며, 본 논문에서는 자세한 기술을 생략한다. 또한 비서 에이전트 시스템을 위한 지식기반 시스템의 동작과정은 다음 장에서 자세히 기술한다.

3. 비서 에이전트의 동작

이 장에서는 하나의 비서 에이전트가 특정의 클라이언트에 생성되고, 서버로 이동하여 서버에서 자신의 작업을 완료하고 다시 원래의 클라이언트로 되돌아오는 모든 과정을 기술한다. 이러한 비서 에이전트의 동작을 효과적으로 기술하기 위해 간단한 상황에 대한 스케줄 작업을 예제로 이용한다. 설명의 편의를 위해서 날짜나 시간에 대한 데이터베이스 내부 표현인 대정수 표현을 배제하고 실생활의 시간 표현을 그대로 이용한다.

[예제]

- 사용자 요구 : ***"1999. 1. 25.(Mon)의 13 : 00에서 17 : 00사이의 빈 시간을 찾아라"***
- 스케줄 데이터베이스는 다음 데이터를 포함한다.

```
    ⋮
1999. 1.4(Mon) 16:00~14:30 ··· Weekly ···   Due: 1999. 5. 31 ···
    ⋮
1999. 1.25.(Mon) 13:30~14:30 ···
1999. 1.25.(Mon) 15:30~16:00 ···
    ⋮
```

먼저 특정의 사용자가 웹 브로우저를 이용하여 서버에 비서 에이전트를 요청하면, 서버 상의 에이전트 관리자는 사용자 확인을 거친 후에, 그 사용자에 해당하는 클라이언트 인터페이스와 비서 에이전트의 쌍을 요청한 클라이언트에 전송한다. 사용자는 전송된 클라이언트 인터페이스를 통해 원하는 작업을 입력할 수 있으며, 비서 에이전트는 이러한 사용자의 요구를 클라이언트 인터페이스를 통해 접수받고, 그 요구를 가지고 서버로 이동한다.

비서 에이전트가 서버에 도착하면 서버의 에이전트 관리자는 비서 에이전트가 특정의 작업을 처리할 수 있도록 관련된 서비스 루틴을 제공한다. 만일 비서 에이전트가 특정의 스케줄 자료를 요구하면, 비서 에이전트를 데이터베이스 시스템 연결부로 이동하도록 하여 작업을 DBA에 의뢰하도록 한다. DBA는 비서 에이전트의 작업을 전달받아 이를 SQL 문장으로 변형한 뒤 스케줄 데이터베이스에 작업처리를 의뢰하며, 작업처리 결과를 다시 비서 에이전트에게 전달 해준다. 이 예제에서 사용자 요구인 "1999. 1. 25.(Mon)의 13 : 00에서 17 : 00사이의 빈 시간"을 찾기 위해서, 먼저 이 시간대에 포함되는 스케줄들을 찾게 되는데, 비서 에이전트는 이 시간대에 포함되는 스케줄 자료의 조회를 DBA에 요구하게 된다(이 예제의 경우 1월 4일의 주간(weekly) 스케줄도 같은 요일이며 유효기간에 포함되므로 이 요구에 반영된다). DBA는 이러한 요구를 SQL 문장으로 변환하여 데이터베이스에 작업을 의뢰하고 그 결과를 다시 스케줄 자료로 변형하여 비서 에이전트에게 전달해 준다.

또한, 이 예제에서 주어진 목표를 달성하기 위해서는 조회된 스케줄 자료를 이용하여 주어진 목표를 만족하는 시간대들을 추출하기 위한 판단 과정이 필요하다. 이를 위해서 비서 에이전트는 조회된 스케줄 자료들을 가지고 지식기반 시스템 연결부로 이동하여 조회결과와 목표 조건을 KBA에게 전달해준다. KBA는 비서 에이전트에게서 받은 스케줄 자료와 목표 조건을 HEXPERT 시스템의 사실 표현 형태로 변환하고 HEXPERT의 추론 엔진을 기동시킨다. 이 예제에서의 조회된 스케줄 자료를 KBA가 변형한 형태는 다음과 같다. 여기서 생성되는 각 스케줄 사실은 'plan'이라는 FACTCLASS의 인스턴스로 생성되며, 그 중 목표를 나타내기 위한 사실은 아래의 'p4'와 같이 상태가 'subgoal'로 표시된다.

```
FACT p1(plan)
{
 uID : "chpark"
 Ts : 99/1/25 13:30
 Te : 99/1/25 14:30
 interval : 100
 state : null
    ⋮
}

FACT p2(plan)
{
 uID : "chpark"
 Ts : 99/1/25 15:30
 Te : 99/1/25 16:00
 interval : 0:30
 state : null
    ⋮
}

FACT p3(plan)
{
 uID : "chpark"
 Ts : 99/1/4 16:00
 Te : 99/1/4 16:30
 interval : 0:30
 state : null
 kind: "weekly"
 valid: 99/5/31
    ⋮
}

FACT p4(plan)
{
 uID : "chpark"
 Ts : 99/1/25 13:30
 Te : 99/1/25 17:30
 interval : 0:40
 state : "subgoal"
    ⋮
}
```

위에서 주어진 사실들을 이용하여 아래의 규칙 'rule1'은 반복적으로 적용되어 새로운 부목표 사실들을 생성하게 되며, 규칙 'rule2'는 주간 스케줄에 대해서 날짜 부분만 목표 날짜로 변경(99/1/4이 99/1/25로)하여 현재의 스케줄 조정에 이용될 수 있도록 한다.

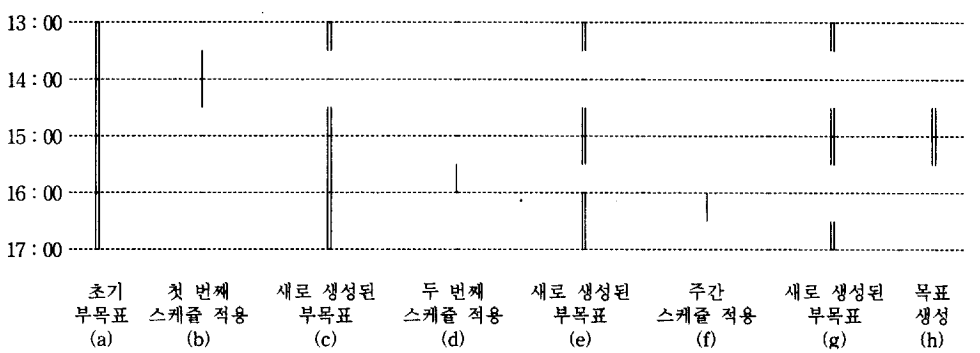
```
RULE rule1(RuleClass1)
{
  for: (px instanceof plan)
       (py instanceof plan)
  if: (px.state == null)
      (px.Ts > py.Ts)
      (px.Te < py.Te)
  do: (create(plan) { Ts=py.Ts, Te=px.Ts,
          state="subgoal", interval=py.interval } )
      (create(plan) { Ts=px.Te, Te=py.Te,
          state="subgoal", interval=py.interval } )
      (delete px, py)
}

RULE rule2(RuleClass1)
{
  for: (px instanceof plan)
       (py instanceof plan)
  if: (px.kind == "weekly")
            (py.state = "subgoal")
      (px.Te <= py.Ts)  // 목표의 날짜가 weekly 스케줄의
      (px.valid >= py.Te) // 유효기간 내에 포함되는 경우
   do: (px.Ts=changeDate(px.Ts, py.Ts))
                    // changeDate(x, y)는 x의 날짜부분만
      (px.Te=changeDate(px.Te, py.Ts))
                                // y의 날짜부분으로 변경
}
```

위의 'rule1'이 반복 실행된 후에는 초기 부목표로 주어진 사실(p4)은 제거되고 새로운 부목표들이 생성된다. 최종적으로 목표를 생성하기 위해서는 목표 조건을 만족하는 아래의 규칙이 실행되어야 한다.

```
RULE rule-final(RC1)
{
  for: (px instanceof plan)
       (py instanceof plan)
  if: (px.state == "subgoal")
      ((px.Te-px.Ts) > px.interval)
     -(py.state==null)
  do: (px.state="goal")
}
```

이제 이 규칙이 적용되어 생성되는 사실은 목표 조건을 만족함으로 이 사실에 주어진 시간은 예약 가능한 시간이 될 수 있다. 이와 같이 진행되는 추론 과정을 도식적으로 설명하면 대략적으로 (그림 4)와 같다. (그림 4)를 간단히 설명하면, 먼저 초기 부목표는 13 : 00~17 : 00가 모두 빈 시간으로 가정한다(a). (b)에서 첫 번째 스케줄 사실이 'rule1'에 적용되면 새로이 두 개의 부목표(13 : 00~13 : 30, 14 : 30~17 : 00가 빈 시간)가 생성되고(c), 반복적으로 (d), (e)에서 부목표가 세 개가 된다. 여기에 주간 스케줄 사실이 적용(f)되면, (g)에서와 같은 부목표만 남게되고 더 이상 적용할 사실들이 없게 된다. 이제 마지막으로 'rule-final'이 적용되면 (h)에서와 같이 하나의 목표 사실만 남게 되어 추론은 종료한다.

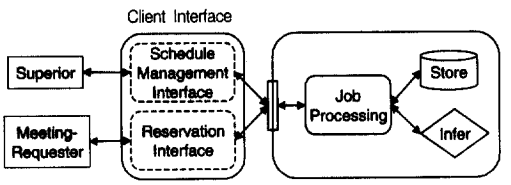
마지막으로 KBA는 목표 조건을 만족하는 사실들을 지식기반 시스템에서 전달받아 스케줄 자료로 변형하고 이를 비서 에이전트에게 전해준다. 최종적으로 비서 에이전트는 결과를 가지고 원래의 클라이언트 위치로 되돌아 와서 클라이언트 인터페이스에게 결과를 전해주고 사용자에게 보여주도록 한다.



(그림 4) 비어 있는 시간대의 추론 과정

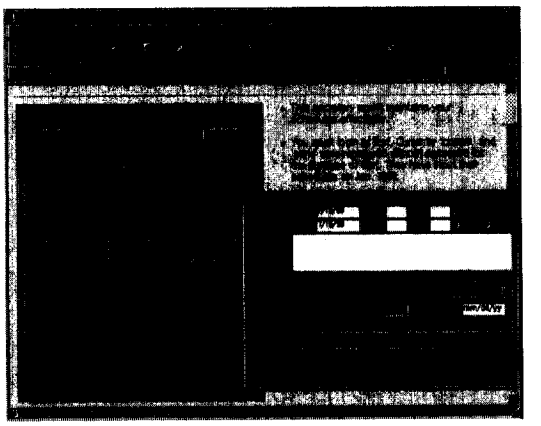
4. 비서 에이전트의 사용자별 관점

본 논문에서 제시하는 비서 에이전트의 사용자는 두 그룹으로 분류될 수 있는데, 한 그룹의 사용자는 자신의 스케줄 관리를 요구하는 상사에 해당하는 사용자들이며, 다른 한 그룹은 상사에게 면담을 요청하는 사용자의 그룹이다. 그러나 어떤 사용자의 관점에서 보더라도 자신이 접근하고 있는 비서 에이전트는 동일한 저장 능력 및 판단 능력을 지니고, 사용자로부터 특정의 작업을 받고 결과를 보여주는 하나의 개체로 보여진다. (그림 5)는 이와 같은 사용자의 관점에서 보는 비서 에이전트의 개념적인 구조를 보여주고 있다. 즉, 비서 에이전트 호출 시에 사용자의 확인을 통하여 비서 에이전트는 해당 클라이언트 인터페이스를 동반할 수 있다.



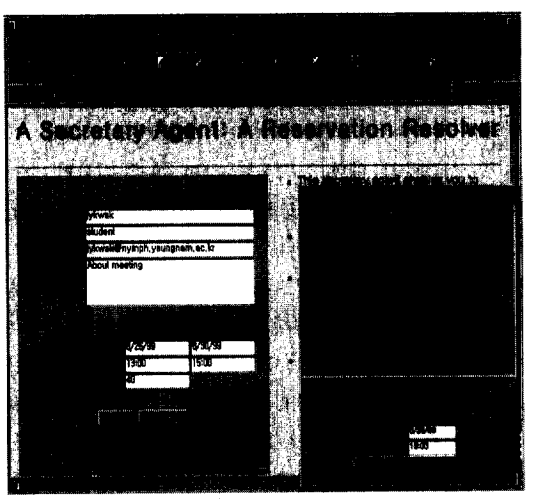
(그림 5) 비서 에이전트 시스템의 개념적 구조

그러나, 비서 에이전트 입장에서는 사용자에 따라서 수행해야 하는 작업은 아주 다르다. 먼저, 상사에 해당하는 사용자들을 위한 비서 에이전트는 주로 상사의 스케줄을 저장, 조회하는 일을 담당하며, 회의 시간 전에 상사의 전자 메일을 통하여 특정의 스케줄을 미리 상기시켜주는 역할을 담당한다. 상사를 위한 비서 에이전트의 주요 작업은 다음과 같다. 먼저 상사로부터 스케줄을 입력받고 입력이 다른 스케줄과 겹치지 않는지를 판별하고 다른 스케줄과 겹치지 않으면 이를 저장한다. 상사가 특정한 날짜에 스케줄이 없는 시간을 찾기를 원하면 비서 에이전트는 그 날의 스케줄에 대한 추론을 통하여 빈 시간들을 알려 준다. 만약 스케줄이 매일의 스케줄이면 이를 모든 날짜에 적용하고, 또는 주간 스케줄이면 요일을 체크하여 모든 주의 같은 요일에 적용한다. 만약, 상기를 원하는 스케줄인 경우 날짜와 시간을 검사하여 해당 시간에 상사에게 전자메일을 전송한다. 이러한 스케줄 작업 중에 외부에서 예약이 들어오면 이를 상사에게 알려서 이를 확인할 수 있도록 하며, 확인한 결과를 받아서 다시 그 결과를 예약자에게 전자메일로 전송한다. 이러한 작업을 수행하는 비서 에이전트를 위한 클라이언트 인터페이스는 (그림 6)에 주어져 있다.



(그림 6) 스케줄 관리를 위한 비서 에이전트 화면

예약 처리를 위한 비서 에이전트의 주요 작업은 다음과 같다. 예약 요청자와 접속이 되면 비서 에이전트는 먼저 예약자에 대한 명세를 입력하도록 하여 예약자 정보를 상사가 확인할 수 있도록 한다. 예약 절차를 수행하기 위해서 예약을 원하는 기간과 시간대, 그리고 소요시간 등(예를 들면, 1999년 1월 25일에서 1월



(그림 7) 예약 처리를 위한 비서 에이전트 화면

28일 사이에 13 : 00에서 17 : 00 사이에 40분 정도 면담을 원함)의 예약 조건을 입력받은 후, 비서 에이전트는 이 예약 조건과 그 시간대의 스케줄 자료들을 데이터베이스에서 검색하여 지식기반 시스템에게 예약 가능한 시간을 추출해 주기를 의뢰한다. 추출된 예약 가능한 시간을 예약자에게 보여주고 예약자로 하여금 하나를 선택하도록 하여 하나의 예약을 접수한다. 예약이 접수되면 비서 에이전트는 바로 이 예약을 상사에게 알려 상사가 이를 확인할 수 있도록 한다. 예약 접수를 위한 비서 에이전트의 클라이언트 인터페이스는 (그림 7)에 주어져 있다.

5. 결 론

최근 인터넷 및 정보통신 기술이 급속하게 발달함에 따라서 네트워크를 기반으로 한 소프트웨어 및 기술들이 많이 개발되고 있으며, 또한 과거의 인공지능 분야에서 많은 연구가 있었던 에이전트 기술이 최근의 네트워크 기술과 결합한 에이전트 시스템 개발에 많은 연구들이 진행 중이다. 본 논문에서는 통상의 비서업무 중에서 상사의 스케줄 업무를 대행하는 스케줄링 비서 에이전트 시스템의 모델을 설계하고 이의 구성 기법을 기술하였다. 본 논문의 비서 에이전트 시스템은 서버 및 각 에이전트가 자바 가상기계에서 수행되며, 에이전트와 서버 사이의 전송 기법은 HTTP에 의한 클라이언트-서버 기술을 이용하였다. 본 논문의 비서 에이전트 시스템은 에이전트의 이동성, 협력성 등을 이용하여 비서 에이전트가 상사 스케줄 관리뿐만 아니라 상사와 면담 예약자 사이에서 다양한 상호 작용 및 스스로의 판단 능력을 통하여 실제 비서의 동작과 유사하게 동작함을 보였으며, 이는 일반적인 개인 스케줄 관리 프로그램이나 네트워크로 연결된 여러 사용자의 스케줄을 일괄적으로 처리하는 그룹 스케줄링과는 다르다. 이를 위해서 본 논문의 비서 에이전트 시스템은 데이터베이스 시스템과 지식기반 시스템을 포함하여 효율적인 스케줄 자료 처리 및 스케줄 자료에 대한 판단 능력을 부여하며, 비서 에이전트를 이동 코드로 구성하여 네트워크 상에서 효과적으로 이동할 수 있도록 하였다.

참 고 문 헌

[1] C. Forgy, "Rete : A Fast Algorithm for the Many Pattern/Many Object Pattern Problem," *Journal of Artificial Intelligence*, Vol.19(1). pp.17-37, 1982.

[2] Suk I. Yoo, Il K. Kim, Chang H. Park, Hea J. Chang, Tae G. Kim, Mee K. Min, "HEXPERT : An Expert System Building Tool," *Third IEEE International Conference on Tools for Artificial Intelligence*, pp.510-511, 1991.

[3] Ben Smith, Howard Eglowstein, "Scheduling Across the Enterprise," BYTE, June, 1994.

[4] M. Wooldridge, N. R. Jennings, "Intelligent Agents : Theories, Architectures, and Languages," Lecture Notes in AI, Vol.890, Springer-Verlag, 1995.

[5] D. S. Milojicic, M. Condict, F. Reynolds, D. Bolinger, P. Date, "Mobile Objects and Agents," *Second USENEX Conference on Object Oriented Technologies and Systems(COOTS)*, 1996.

[6] Stan Franklin and Art Graesser, "Is it an Agent, or just a Program? : A Taxonomy for Autonomous Agents," *The 3rd International Workshop on Agent Theories, Architectures, and Languages*, Springer-Verlag, 1996.

[7] D. Flanagan, "Java in a Nutshell," Second Edition, O'Reilly, 1997.

[8] D. Wong, N. Paciorek, T. Walsh, J. DiCelie, M. Young, B. Peet, "Concordia : An Infrastructure for Collaborating Mobile Agents," *First International Workshop on Mobile Agents*, 1997.

[9] Graham Hamiltion, Rick Cattell, Maydene Fisher, "JDBC : Database Access with Java : A Tutorial and Annotated Reference," Addison Wesley, 1997.

[10] M. Straβer, J. Baumann, F. Hohl, "MOLE : A Java Based Mobile Agent System," *European Conference on Object Oriented Programming*, pp.301-308, 1997.

[11] P. Chan, R. Lee, "The Java Class Libraries," Second Edition, Vol.2, Addison-Wesley, 1998.

[12] S. Green, L. Hurst, B. Nangle, P. Cunningham, F. Somers, R. Evans, "Software Agents : A Review," Trinity College Dublin and Broadcom Éireann Research Ltd, IAG Report, 1997.

[13] "The Java Tutorial : A Practical Guide for Programmers," Sun Microsystems, Inc., "http : //java.sun.com/docs/tutorial/index.html".

[14] "MySQL," T.c.X, "http : //www.mysql.com/".

박 창 현

e-mail : park@cse.yeungnam.ac.kr

1986년 경북대학교 공과대학 전자공학과 전산학전공(공학사)

1988년 서울대학교 자연과학대학 계산통계학과 전산학 전공(이학석사)

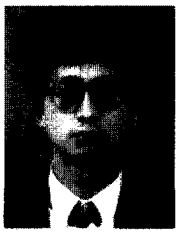
1992년 서울대학교 자연과학대학 계산통계학과 전산학 전공(이학박사)

1992년~1993년 서울대학교 컴퓨터신기술공동연구소 특별연구원

1998년~1999년 University of Maryland, Institute of Advanced Computer Systems, Visiting Researcher

1993년~현재 영남대학교 컴퓨터공학과 부교수

관심분야 : 인공지능, 지식기반 시스템, 데이타 마인닝 및 지식 발견, 에이전트 시스템



정 호 열

e-mail : hoyoul@ynucc.yeungnam.ac.kr

1988년 아주대학교 공과대학 전자공학과 전자공학전공(공학사)

1990년 아주대학교 공과대학 전자공학과 전자공학전공(공학석사)

1993년 아주대학교 공과대학 전자공학과 전자공학전공(공학박사 수료)

1998년 (프)리옹국립응용과학원(INSA de Lyon) 전자공학전공(공학박사)

1998년~1998년 (프)CREATIS 박사후 과정

1999년~현재 영남대학교 전자정보공학부 전임강사

관심분야 : 음성/영상 신호처리, 인공지능, 디지털 워터마킹 등