

이동 에이전트 엔진을 이용한 웹 기반 정보 검색 시스템의 설계 및 구현[†] (Design and Implementation of Web-based Information Searching System with Mobile Agent Engine)

오 동 석* · 김 승 권** · 박 순 철***
(Dong-Seok Oh* · Seung-Gwon Kim** · Soon-Cheol Park***)

요약 본 논문은 소프트웨어 에이전트의 한 종류인 이동 에이전트를 이용한 검색 시스템에 관한 내용으로서, 분산된 여러 서버에 이동성이 있는 객체들을 보내어 주어진 일을 수행한 후 에이전트로 돌아와 웹서버에게 결과를 전해주고 이를 웹에서 보여주는 시스템의 설계 및 프로토타입을 제시한다. 시스템 설계는 표준화된 객체지향 모델링 언어인 UML을 이용한다. 기존의 많은 시스템에서는 분산된 자료를 얻는 데 있어 지속적인 네트워킹을 전제로 하지만 본 논문에서 제시한 시스템은 도중에 네트워킹이 되지 않아도 주어진 일을 끝까지 수행하며 나중에 네트워킹이 되면 원하는 자료를 얻을 수 있는 장점이 있다. 이 시스템의 구현을 위하여 이동 에이전트로서 IBM사의 순수 자바 제품인 AGLET을 사용한다.

Abstract This paper proposes a searching system with a mobile agent which retrieve data from the distributed hosts. The proposed system dispatched the movable objects, so called AGLET, to the distributed hosts in order to perform a task given by a client or a server. The network of the existing system must not be disturbed during a task is performing. However our system has the strong point that a task can be performed even if the network is disconnected on the way. When the network is disconnected, the system can get the results later after the network system is connected again. Designing the system has been done by using UML(Unified Modeling Language) which is a standardized object-oriented modeling language. AGLET, a pure JAVA product of IBM, is used for the mobile agent.

1. 서 론

인터넷의 급속한 확산과 발전 및 여러 분야 정보의 범람으로 이를 처리하는 소프트웨어에 있어 단순히 수동적으로 응답하는 수준에서 벗어나 사용자의 요구를 능동적으로 해결할 수 있는 다양한 기능이 요구되는 소프트웨어가 필요시 되고 있다. 이에 대두되고 있는 컴퓨터 기술이 소프트웨어 에이전트 개념으로 이는 다양하게 연구되어 여러 속성을 가지는데 그 속성에 따라 여러 에이전트로 구분하고

있다. 여러 속성 중 이동성에 중점을 두어 나뉘어 진 것이 이동 에이전트로서, 인터넷 같은 분산 환경에서 사용자가 요구하는 정보를 망 내의 필요한 여러 호스트들을 자율적으로 이동하면서 사용자 대신 수집하여 제공해 주는 소프트웨어 프로세스를 말한다.[1,2,3]

현재 널리 퍼져있는 인터넷의 한 분야인 월드 와이드 웹(World Wide Web, 이하 웹)의 경우 호스트 서버에 접속하여 원하는 정보를 가져올 경우 지속적인 네트워킹을 전제로 하고 있다. 본 논문에서 구현한 시스템은 이동 에이전트 서버를 두어 여러 호스트에 객체들을 보내어 주어진 일들을 동시에 수행하게 한다. 각각의 작업들이 분산된 호스트에서 실행되기 때문에 효율적인 작업 수행이 가능하

† 본 연구는 전북대학교 부설 정보통신연구소의 연구지원에 의한 것임.
* 전북대학교 영상공학과
** 전북대학교 정보통신학과
*** 전북대학교 정보통신공학과 교수

며, 한번 파견되어 작업을 수행하는 동안에는 비동기 작업 수행으로 서버의 네트워크 부하를 줄일 수 있다. 작업이 끝나면 객체들은 네트워크 연결을 기다리게 되는데 나중에 네트워킹이 복원되면 서버로 돌아와 결과를 전달하게 된다. 위와 같은 장점을 검색엔진에 이용하여 본 논문이 구현한 시스템은 서버가 실행하는 작업의 부담을 줄이고 서버와 호스트간의 네트워크 부하를 줄이며, 또한 사용자의 서버에 대한 네트워크 유연성을 높이고자 한다.

본 시스템 설계에 있어서는 소프트웨어 시스템의 객체 지향적인 분석과 설계를 위한 산업 표준 객체지향 모델링 언어인 UML을 사용한다. UML 설계에서는 일련의 다이어그램을 작성하는데 있어 반복적인 작업을 거쳐 시스템을 구현하기까지 객체지향적 프로그래밍에 가깝도록 한다.

시스템 구현에는 프로그래밍 언어로 가장 객체지향적이며 플랫폼에 구애받지 않는 자바를 사용하고, 이동 에이전트 서버는 IBM사의 순수 자바 제품인 AGLET SDK를 사용한다. 그리고 시스템 가동에 필요한 데이터베이스는 ORACLE을 사용한다.

본 논문의 2장에서는 시스템 개발 연구의 이론적 배경을 소개하고, 3장에서는 시스템의 전체 구조와 세부적인 기능을 설명한다. 그리고 4장에서는 UML을 이용한 본 시스템의 설계를 여러 다이어그램을 들어 설명하고 5장에서는 구현 결과를 보여준다. 그리고 끝으로 6장은 결론 및 향후과제를 제시한다.

2. 이론적 배경

2.1. 이동 에이전트

2.1.1. 소프트웨어 에이전트

소프트웨어 에이전트 기술은 최근에 "1990년대에 정보통신 분야에 나타난 가장 중요하고도 흥미로운 개념이며, 사람들이 컴퓨터를 다루는 방식에서뿐만 아니라 매우 복잡한 시스템을 개념화, 설계, 구현하는 방식에서 일대 변환을 가져다 줄 것"으로 알려지고 있다. 소프트웨어 에이전트의 연구는 다양한 분야에서 자체적으로 진행되어 온 만큼 그 정의도 여러 가지 형태로 나타나고 있는데 이 들이 가지고 있는 속성에 따라 구분을 하고 있다. 중요한 구분의 기준이 되는 속성으로는 자율성(Autonomy), 지능성(Intelligence), 이동성(Mobility), 사회성(Social Ability)이 있는데, 이중 독립성만이 모든 에이전트에 공통된 속성이며 본 시스템에 응용을 하게 되는 이동 에이전트에 대하여 알아본다.[1]

2.1.2. 이동 에이전트의 개요

소프트웨어 에이전트의 여러 분야 중 이동성에 중점을

둔 이동 에이전트는 인터넷 등과 같은 분산 환경에서 사용자의 요구에 충족하는 정보를 망 내의 필요한 여러 호스트들을 자율적으로 이동하면서 대신 수집 제공해 주는 소프트웨어 프로세스를 지칭한다. 이러한 기능을 제공해 주기 위해서는 대체로 에이전트를 프로그램 할 수 있는 프로그램 언어와 이를 수행하는데 필요한 인터프리터, 그리고 에이전트의 전송과 서로간의 통신을 위한 프로토콜로 구성되어 있다. 프로그램 언어는 에이전트 개발자가 원하는 에이전트 기능(예를 들면, 특정 서비스 제공자에게 옮겨 그 곳의 에이전트 서버나 다른 에이전트와 접촉하거나 특정 알고리즘을 수행 등)을 시스템 플랫폼에 관계없이 간단하게 구현하게 하며, 인터프리터는 프로그램 되어진 에이전트나 에이전트 서버가 시스템 플랫폼에 관계없이 수행되어 지는 환경을 제공하며, 통신 프로토콜은 에이전트들이 하부의 네트워크 프로토콜에 상관없이 서로 이동할 수 있는 통로를 제공한다.[1,2]

이동 에이전트 기술에 많은 관심이 쏠리는 이유는 이 기술을 활용할 경우 얻을 수 있는 다양한 장점들이 있기 때문이다. 이들을 정리하여 크게 둘로 분류해보면 성능의 향상과 유연성의 증가일 것이다. 본 연구의 시스템에서 주요하게 필요로 하는 이동 에이전트의 중요한 특성을 살펴보면 다음과 같이 정리할 수 있다.

- 이동 에이전트가 이동할 때 객체의 모든 것이 이동하는데 자체 코드, 데이터, 실행상태, 그리고 지나는 경로 등을 포함한다.
- 언제, 어디서, 무엇을 해야 하는지에 대한 충분한 정보를 가지고 있다.(자율성)
- 자신의 프로세스를 가지고 있어 비동기적으로 실행된다.
- 다른 이동 에이전트나 지역적으로 고정된 호스트와 상호작용한다.
- 네트워크 연결에 관계없이 자신의 일을 수행하며 에이전트가 이동할 필요가 있다면 다시 연결될 때까지 기다린다.
- 둘 이상의 에이전트가 파견되어 서로 다른 장소에서 병렬적으로 일을 수행한다.

위의 특성으로 인하여 이동 에이전트는 불필요한 네트워크 트래픽을 줄이며 사용자의 임무를 감소시킴으로써 해서 시스템의 안정성이 증가된다. 또한 분산 일처리를 자동화할 수 있고 적당한 소프트웨어 분배로 유연성이 증가된다.[1,11]

2.1.3. 이동 에이전트 제품

1990년 초부터 학계, 산업계, 연구기관 등 여러 곳에서 이동 에이전트 기술을 개발했거나 연구 중에 있으며 이들 대부분은 이동 에이전트를 생성, 이동, 수행, 종료하는데

필요한 시스템, 프로토콜, 언어 등을 포함한 이동 에이전트 프레임워크를 제공하고 있는데, 그 중 대표적인 프레임워크로는 IBM의 Aglets, General Magic의 Telescript, Mitsubishi의 Concordia, Dartmouth 대학의 Agent Tcl, 그리고 Maryland 대학의 TKQML 등이 있다.[1]

이중에서 시스템 구현에 사용된 에이전트 엔진은 IBM사의 Aglets을 사용하였다. Aglets은 순수 자바 제품으로서 플랫폼에 독립적인 뿐 아니라 인터넷을 목적으로 하고 URL(Universal Resource Locators)을 사용하며 HTTP Request를 메시지로써 받을 수 있기 때문에 웹과의 연결이 용이한 장점이 있다.[11]

2.2. UML(Unified Modeling Language)

용어의 의미 그대로 '통합된 모델링 언어'로서 설계를 표현하기 위한 표시법(Notation)인데 선이나 화살표, 사각형 등과 같은 기호를 이용해 시각적으로 표현되므로 이해하기가 쉽다. UML은 산업 표준 객체지향 모델링 언어로 객체지향 소프트웨어 개발자는 분석과 설계를 위해 UML을 많이 사용한다. UML은 팀을 이뤄 시스템을 개발할 경우 개발자간의 공통의 의사소통 도구로서도 큰 역할을 하는데 시스템 개발 도중 한 개발자가 바뀌더라도 원활한 인수인계를 할 수 있다. 또한 개발된 시스템이나 그 일부를 다시 사용할 수 있도록 해주는 큰 장점을 지니고 있다.

UML을 사용하여 모델링한다는 것은 서로 다른 관점에서 시스템을 구조를 표현한 일련의 다이어그램을 작성하는 중에 추가, 수정, 삭제의 반복 작업을 거치면서 객체 지향적 프로그래밍에 도달한다는 것이다. 이러한 작업을 용이하게 해주는 여러 툴 중에 본 연구에서는 Rational사의 'Rational Rose Java'를 사용하였다.[4]

3. 이동 에이전트 검색 시스템

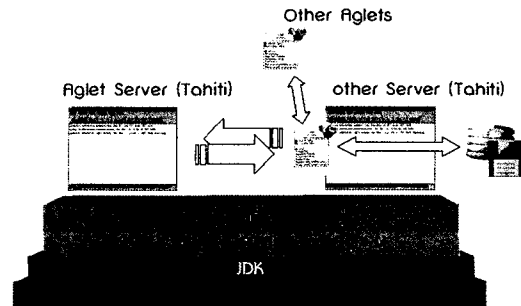
3.1. IBM의 AGLET

2.1.에서 언급한 특성을 가진 이동 에이전트 엔진인 AGLET은 자바 객체로서 서론에서 언급한 본 시스템에서 호스트에 파견되는 검색 객체의 기능을 수행하게 된다. AGLET은 인터넷상에서 호스트간을 이동할 수 있는데 한 호스트에서 주어진 일을 수행하다가 중간에 멈추고 다른 호스트로 이동하여 그 일을 다시 시작할 수 있다. 이때 AGLET은 자체 프로그램 코드와 데이터를 가지고 이동하는데 이것이 자율성을 지원하게 된다. 이러한 AGLET은 개념적으로 이동 에이전트의 특성인 자율성과 역동성 등을 가진다.

하부적으로 AGLET은 자바로 만들어져 구현이 용이하고 자체 인터페이스(J-AAPI, Java-AGLET Application

Programming Interface)를 지원하는 호스트이면 어느 플랫폼에서도 구동이 가능하다. 그리고 수행에 있어 필요한 인터프리터로서 Java Development Kit(이하 JDK)를 요하며, 전송과 서로간의 통신을 위한 표준 API(J-ATCI, Java-AGLET Transfer Communication Interface)를 제공한다. 또한 웹에서 AGLET을 구동시키기 위하여 플러그인과 라이브러리를 지원한다. 이를 기반으로 에이전트의 이동과 통신을 가능하게 하며 이 외에 보안과 WWW 통합의 특성을 가진다.[11]

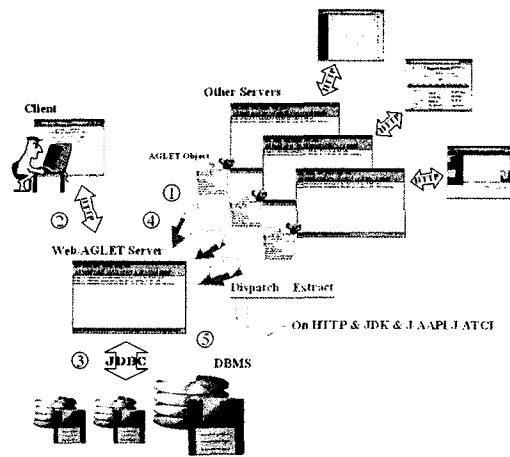
개념적으로 AGLET은 <그림 1>과 같은 구조를 가진다.



<그림 1> AGLET의 하부구조

3.2. 시스템의 구조

본 시스템의 전체 구조를 도식화하면 <그림 2>와 같다.



<그림 2> 이동 에이전트 검색 시스템의 구조

시스템의 대략적인 데이터 검색 알고리즘은 다음과 같다.

① 서버는 분산된 데이터베이스에 AGLET을 보내어 각 호스트가 가지고 있는 자료에 대한 정보를 서버의 데이터

베이스에 두게 된다.

② 사용자는 쉽게 접할 수 있는 웹으로 브라우저를 통해 AGLET으로 구현한 웹서버에 접속하여 원하는 데이터를 요구하게 된다.

사용자가 요구한 정보가 서버의 데이터베이스에 있을 경우에는,

③ 서버는 관리자가 수집한 자료에서 사용자가 원하는 정보를 바로 보여주게 된다.

그렇지 않을 경우에는,

④ AGLET 검색 객체를 여러 호스트에 보내어 관리자가 지정한 다른 검색 서버로부터 자료를 가져와 사용자에게 보여줄 수가 있다.

⑤ 서버는 객체를 호스트에 보내어 가져온 결과를 따로 저장하여 사용자가 임의로 또는 네트워크 문제나 시간 초과 등의 이유로 서버와의 접속이 끊어져도 다음에 다시 서버에 접속하면 그 결과를 볼 수 있게 된다.

본 시스템의 구성 요소는 크게 사용자/관리자의 인터페이스를 담당하고 AGLET 검색 객체를 담당하는 'Web/AGLET Server', 시스템의 사용자/호스트/검색 서버의 정보와 데이터를 관리하는 'DBMS', 그리고 호스트에 파견되어져 작업을 수행하는 'AGLET Object'로 이루어진다.

3.2.1 Web/AGLET Server

<그림 2>를 보면 AGLET Web Server라고 되어 있는 윈도우가 AGLET을 관리하는 툴인 'Tahiti'이다. 이는 독립 서버로서 AGLET 환경 제공, AGLET 전송 프로토콜 데몬, 그리고 AGLET 뷰어의 기능을 수행할 수 있다. 이 툴 안에서 AGLET은 생성되고 HTTP 메시지를 주고받을 수 있고 다른 호스트의 'Tahiti'로 옮겨 다닐 수 있다.[11]

사용자는 브라우저를 통하여 본 검색 시스템에 접속하게 된다. 이를 위해 시스템은 웹 서버를 두는데 본 시스템에서는 전문적인 웹 서버를 두어 AGLET 서버와 연동하지 않고 AGLET으로 시스템의 일부인 웹 서버를 구현한다. AGLET의 메시지 전송은 HTTP 메시지에 포함되어 전달된다. 따라서 HTTP 메시지를 직접 받을 수 있기 때문에 Tahiti 안에서 생성된 AGLET은 시스템에서 멀리 떨어진 웹 브라우저에서 AGLET을 관리할 수 있을 뿐만 아니라 사용자로부터 HTTP/CGI 요구를 받아 일을 수행한 후 브라우저로 결과 내용을 보낼 수가 있다.

또한 사용자가 직접 AGLET 객체를 호스트로 보낼 때에도 사용자의 브라우저에서 직접 보내지 않는다. 본래 AGLET 사양에는 이를 지원하는 툴인 Fiji(Agent Web Launcher)가 있지만 아직 불안정한 데다가 IBM에서도 이 툴의 개발을 중지한 상태이다. 그래서 본 연구에서는 사용자가 시스템에 접속하여 객체를 호스트로 보낼 것을

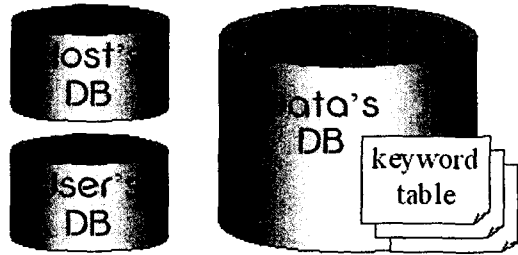
요구하면 웹 서버가 AGLET 서버를 통하여 그 일을 수행할 수 있도록 하였다.[11]

관리자 역시 브라우저를 통하여 서버에 접속하여 AGLET 서버를 통해 객체를 호스트로 보낼 수 있다.

그림에서 보면 'Tahiti' 안에 세 개의 AGLET이 떠있는 데 각각 관리자 웹 서버, 사용자 웹 서버, AGLET 서버이다.

3.2.2. DBMS

DBMS로는 ORACLE을 사용하며 JDBC를 통해 접근한다. 데이터베이스는 크게 data's DB, host's DB, user's DB 로 나뉜다. 관리자가 보낸 객체가 가져온 결과는 data's DB에 있는 '검색 키워드-IP 주소', 'IP 주소-웹 페이지 설명' 테이블에 저장되어 사용자가 요구하면 서버가 이를 바로 보여주게 되며 AGLET 객체가 보내어질 호스트에 대한 정보는 host's DB에 저장한다. 그리고 사용자의 로그인과 마지막 검색단어는 user's DB에 저장되어 사용자가 전에 검색을 요구하고 접속을 중단했을 때 다시 접속하여 이를 통해 그 결과를 볼 수 있도록 하였다. 검색 객체가 가져온 결과가 data's DB에 저장될 때 각 검색 키워드마다 별도의 테이블이 자동적으로 만들어지게 하여 사용자가 요구할 때 데이터베이스에서 찾는 속도를 빠르게 하였다. <그림 3>은 데이터베이스의 대략적인 구성을 나타낸다.



<그림 3> 데이터베이스의 구조

3.2.3. AGLET Object

서버로부터 파견되어 다른 호스트에 도착한 AGLET 객체는 주어진 웹 검색을 하게 된다. 관리자가 보낸 객체는 메타 태그를 검색하여 서버로 돌아오고 사용자가 보낸 객체는 다른 검색 서버에 접속하여 그 결과를 가지고 돌아온다. 이 AGLET 객체가 다른 호스트로 파견되기 위해서는 호스트도 서버와 같은 환경을 가져야 한다. 즉 자바 기반에 AGLET 라이브러리 기반을 가져야 하며 Tahiti 데몬을 띄워야 한다. 이렇게 해야 파견된 AGLET이 머무를 수 있는 환경이 만들어지는 것이다. 호스트에 머물러 검색을 하

는 동안 서버와의 네트워크는 무관하게 되어 서버의 부하를 줄일 수 있으며, 여러 호스트에서 로컬 태스킹이 병렬적으로 수행되기 때문에 작업의 효율성이 증가된다.

4. 이동 에이전트 검색 시스템의 설계

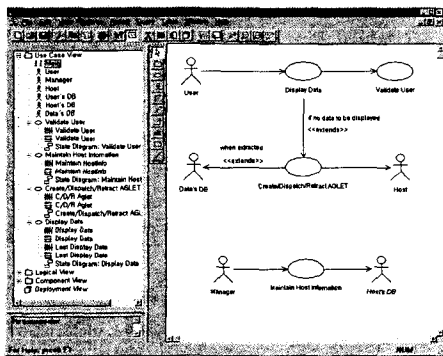
4.1. UML을 이용한 객체지향적 설계

본 연구에서는 요구분석 및 설계를 위해서 객체지향 설계의 표준인 UML을 사용한다. 2.2에서 언급하였듯이 UML은 시스템을 바라보는 시각에 따라 여러 가지 다이어그램으로 구성된다. 본 연구에서는 Use Case 다이어그램을 통해서 사용자의 요구분석을 표현하였고, Sequence 다이어그램과 Collaboration 다이어그램을 통해서 각 Use Case의 전체적인 흐름을 표현하였다. 그리고 Package 다이어그램과 Class 다이어그램으로 시스템의 세부구조를 설계하였다. 그리고 Use Case 다이어그램과 Class 다이어그램의 복잡한 구조는 State 다이어그램으로 세부화하였다.

4.2. USE CASE Diagram

UML을 이용한 시스템 개발은 Use Case 모델링부터 시작된다. 당연히 시스템 개발의 첫번째 단계는 사용자의 요구사항이 무엇인지를 파악하는 것이다. Use Case 다이어그램은 사용자가 시스템에 기대하는 것이 무엇인지를 파악하기 위해 사용된다. 시스템 외부에 존재하면서 시스템과 대화하는 대상은 행위자(Actor)로 설정되고 이에 대한 시스템이 하는 일은 Use Case로 나타낸다.[5,6,7]

3.2에서 얘기한 본 시스템의 데이터 검색 알고리즘을 토대로 Use Case 다이어그램을 표현하면 <그림 4>과 같다.[13]



<그림 4> 이동 에이전트 검색 시스템의 Use Case Diagram

User → Display Data → Validate User : 사용자는 서버에 접속하여 'Display Data'가 수행되기 전 'Validate

User'가 수행되어 접속한 사용자가 이미 등록된 계정인지 아닌지를 판별한 후 신규 접속이면 새로 등록을 하고 난 후 요구하는 데이터를 보여주게 된다.

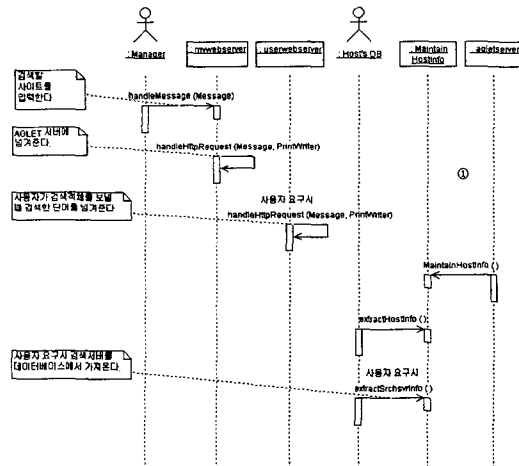
Display Data → Create/Dispatch/Extract AGLET : 사용자가 요구한 데이터가 서버의 데이터베이스에 없다면 AGLET 서버에게 요청하여 검색 객체를 호스트로 보내어 데이터를 검색하고 돌아온다. 그리고 가져온 결과를 서버의 데이터베이스에 저장한다.

Manager → Maintain Host Information → Host's DB : 검색 객체가 보내어질 호스트에 대한 정보와 사용자가 요청하여 호스트에 보내어진 객체가 검색을 요구할 다른 검색 사이트에 대한 정보를 관리한다.

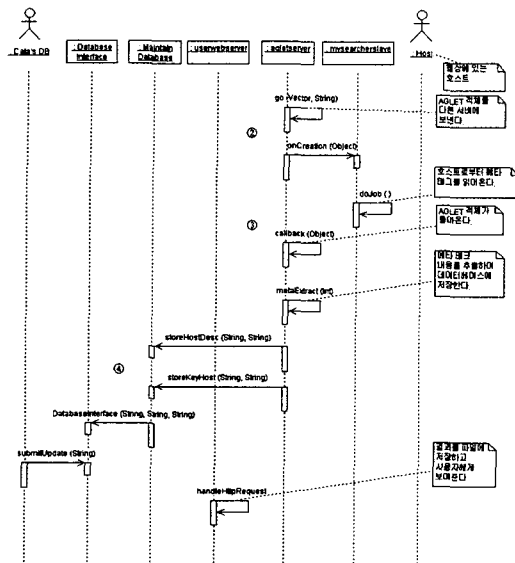
4.3. SEQUENCE / COLLABORATION Diagram

Interaction 다이어그램은 단일 Use Case 내부에 존재하는 객체들과 각 객체간에 주고받는 상호 작용을 묘사하기 위해 사용된다. UML에서는 Interaction 다이어그램으로 Sequence 다이어그램과 Collaboration 다이어그램 등 2가지 유형을 제공하고 있다. 전자는 객체들을 명시한 다음 시간의 흐름에 따라 각 객체들 간에 주고받는 메시지를 표현하여 분석 단계에서 업무 흐름을 파악하기 위한 수단으로 사용하고, 후자는 시간의 흐름과 주고받는 모든 메시지를 평면적으로 표현하여 설계 단계에서 클래스간의 관계를 파악할 수 있다.[4,5,6,7]

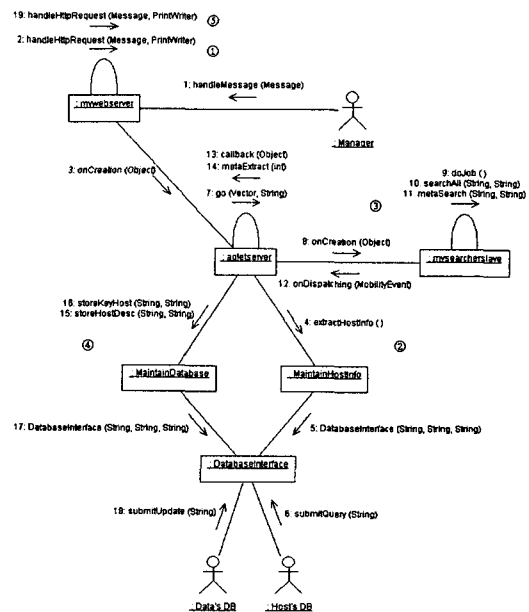
여러 Use Case에 대한 다이어그램 중에서 본 시스템의 핵심이라 할 수 있는 'Create/Dispatch/Extract AGLET'에 대한 다이어그램을 살펴보기로 한다. Sequence 다이어그램은 <그림 5>와 <그림 6>과 같다.



<그림 5> Sequence Diagram 1 for 'Create/Dispatch/Extract AGLET'



<그림 6> Sequence Diagram 2 for 'Create/Dispatch/Extract AGLET'



<그림 7> Collaboration Diagram for 'Create/Dispatch/Extract AGLET'

AGLET 서버는 관리자/사용자 웹 서버와 연결되어 다른 호스트로 AGLET 객체를 보내고 받는 일을 수행한다. 그 과정을 알고리즘으로 나타내면 다음과 같다.

관리자가 AGLET 객체를 보낼 것을 요구하면,
 ① 관리자가 검색할 사이트를 입력하므로 AGLET 서버는 Host's DB에서 호스트 정보만을 가져온다.

사용자가 AGLET 객체를 보낼 것을 요구하면,
 ① AGLET이 파견될 호스트 정보와 검색 서버 정보를 가져온다.

② 이를 바탕으로 AGLET 서버는 여러 호스트에 AGLET 객체를 동시에 보내게 되는데 이 클래스가 'mysearcherslave'이다.

③ 호스트에 도착한 이 객체는 주어진 일을 수행하고 AGLET 서버로 돌아온다.

관리자가 보낸 객체이면,
 ④ 검색 결과는 Data's DB로 저장되고 관리자 웹 서버인 'mywebserver'로 넘겨준다.

사용자가 보낸 객체이면,
 ④ 검색 결과는 파일로 저장하여 다음 접속 때 사용자가 요구할 것에 대비한 다음 사용자 웹 서버로 넘겨준다.

그리고 <그림 7>는 'Create/Dispatch/Extract AGLET' Use Case에 대한 Collaboration Diagram을 나타낸다.

Collaboration 다이어그램은 Sequence 다이어그램과 내용이 같지만 클래스간에 주고받는 메시지 중심이므로 서로 간의 관계를 보다 잘 파악할 수 있다. 관리자의 요구로 검색 객체를 보내는 알고리즘은 다음과 같다.

① 관리자의 인터페이스를 담당하는 'mywebserver'는 관리자로부터 검색할 사이트를 입력받는다.

② AGLET 검색 객체를 관리하는 'agletserver'는 'Host's DB'로부터 호스트에 대한 정보를 가져오기 위하여 'MaintainHostInfo'와 'DatabaseInterface'를 거친다.

③ 호스트로 파견된 AGLET 객체는 주어진 일을 수행하고 돌아온다.

④ AGLET 객체가 가져온 결과는 'Data's DB'에 저장하기 위하여 'MaintainDatabase'와 'DatabaseInterface'를 거친다.

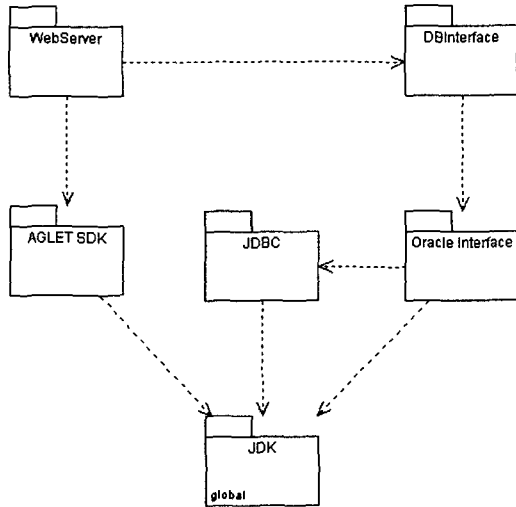
⑤ 관리자 웹 서버인 'mywebserver'에게 수행한 결과를 넘겨준다.

4.4. CLASS Diagram

객체지향 프로그래밍을 한다는 것은 결국 클래스를 만드는 것이고, 설계한 클래스들은 소스코드 작성과 가장 밀접한 관련이 있으므로, 클래스 다이어그램은 UML은 물론 모든 객체지향 방법론에서 핵심적으로 다루어진다.

Use Case 다이어그램과 Sequence 다이어그램의 분석을

토대로 Class 다이어그램을 나타내는데 먼저 Package 단위의 전체 구조를 그려보면 <그림 8>과 같다. 여기서 화살표는 그 패키지에 의존함을 나타낸다.

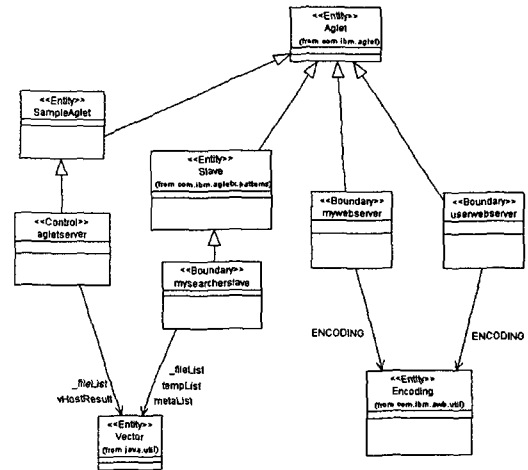


<그림 8> 이동 에이전트 검색 시스템의 Package Diagram

위 다이어그램에서도 보듯이 본 시스템은 자바로 구현된 시스템으로 'JDK'를 기반으로 한다. 그리고 에이전트 엔진인 'AGLET SDK'를 필요로 하며 데이터베이스 접근을 위하여 'JDBC'를 이용한다. 데이터베이스를 접근함에 있어 한글 입출력을 위하여 추가적으로 'Oracle Interface'를 사용한다. 이러한 패키지를 기반으로 사용자/관리자의 인터페이스와 AGLET 객체를 담당하는 'WebServer' 패키지와 데이터베이스 입출력을 담당하는 'DBInterface' 패키지가 있다.

이중에서 'WebServer' 패키지 내부의 여러 클래스간의 관계를 보면 <그림 9>의 Class 다이어그램과 같다. 전체적인 관계를 위하여 각 클래스 내부의 attribute와 method는 생략하였다.

사용자/관리자 인터페이스를 담당하는 'mywebserver'와 'userwebserver'는 'Aglet'를 부모 클래스로 하고 웹 서버 구현을 위하여 'Encoding'을 내부에서 사용하며, AGLET 객체를 담당하는 'agletserver'와 파견되는 객체인 'mysearcherslave'는 각각 'SampleAglet'과 'Slave'를 부모 클래스로 하고 데이터 저장을 위하여 'Vector'를 내부에서 사용한다.



<그림 9> Class Diagram for 'WebServer' Package

5. 구현 및 결과

5.1 구현

4에서의 UML 분석 및 설계를 통하여 구현된 이동 에이전트 검색 시스템은 Windows NT에서 구현하였으며 세부 프로그래밍은 JDK 1.1.8을 인터프리터로 하여 MicroSoft사의 Visual J++ 6.0을 사용하였다. 설계부터 구현까지 객체지향적 프로그래밍에 충실하였으므로 후에 다른 사람이 이어서 개발한다고 하더라도 시스템을 쉽게 파악할 수 있을 뿐 아니라 확장이 용이하다.

본 시스템은 다음과 같은 특징을 가진다.

- 현재 널리 퍼져있는 인터넷의 한 분야인 웹을 이용함으로써 사용자는 손쉽게 서버에 접속할 수가 있다.
- 관리자가 호스트에 AGLET 객체를 보내어 정보를 수집하는 동안 네트워크가 끊어져 있어도 정보수집은 이루어진다.
- 여러 호스트로 파견된 객체들은 로컬 태스킹을 병렬적으로 처리하여 빠른 효율성으로 서버의 부담을 줄인다.
- 사용자가 서버에게 정보를 요구한 후 네트워크가 끊어져도 나중에 다시 접속하여 요구했던 정보를 얻을 수 있다.

5.2 결과

사용자가 로그인한 후 검색어를 입력하여 결과를 얻는 것은 일반 검색 사이트와 유사하므로 생략하고 본 시스템의 가장 큰 특징이라 할 수 있는 네트워크 유연성에 맞추

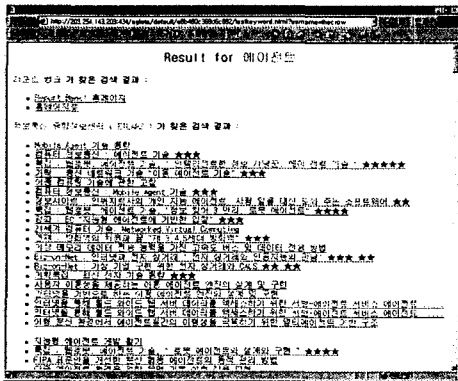
어 설명한다.

만일 사용자가 요구한 검색어에 대한 결과를 서버가 가지고 있는 데이터베이스에 없을 경우에는 <그림 10>과 같이 '검색 객체 보내기'를 선택할 수 있는 창이 보여진다.



<그림 10> 사용자의 검색 객체 보내기 창

<그림 10>의 'Aglet 보내기'를 누르면 서버는 바로 AGLET 객체를 호스트로 보내어 다른 검색 사이트에서 검색을 해 결과를 가져와 사용자에게 보여주고 파일로 저장한다. <그림 11>은 그 결과를 보여주고 있다.

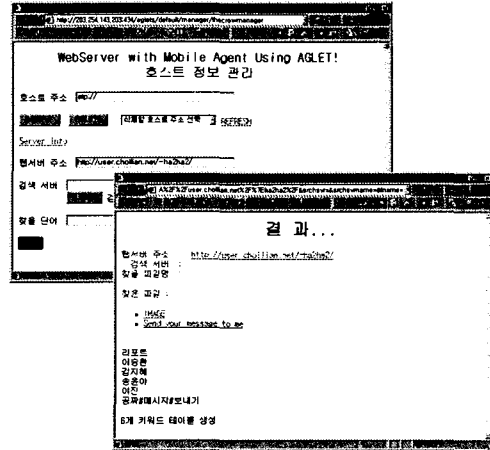


<그림 11> 이전 검색 결과 보기

사용자가 'Aglet 보내기'를 실행하고 검색 객체가 결과를 가져오기를 기다리는 동안 혹은 사용자가 원해서 서버와의 접속을 끊는다 하여도 그 검색 객체는 계속 일을 수행하게 된다. 여기에서 서버와 검색 객체가 머무르고 있는 호스트와의 네트워크가 끊어져도 일을 계속 수행된다. 그래서 주어진 일을 다 마친 객체는 네트워크가 다시 복원되면 서버에게 돌아올 수 있으며 사용자 역시 다음에 다시 서버에 접속하여 이전에 검색 객체를 보내어 수집한 결과(그림 8)를 볼 수가 있다.

그리고 관리자가 접속하는 창은 <그림 12>와 같으며 호스트 정보와 검색 서버 정보를 관리하며, 메타태그 검색을

위한 사이트를 입력하고 객체를 보내게 되면 그 페이지에서 링크된 페이지와 메타 태그에서 키워드의 내용을 골라 Data's DB에 저장됐음을 알려준다.



<그림 12> 관리자 인터페이스

6. 결론 및 향후과제

인터넷의 발달과 정보의 범람으로 이를 요구하는 사용자는 갈수록 증가하며 이에 따라 정보 검색 서버는 보다 지능적이고 빠르고 확실하게 대처해야 한다. 본 연구에서는 이에 대한 일환으로 이동 에이전트 개념을 도입하여 서버가 호스트로부터 정보를 가져올 때에 이동성이 있는 객체를 보내어 정보를 추출함에 있어 병렬적인 로컬 태스킹으로 효율성을 증대시킬 수가 있고 그 객체가 작업을 수행하는 동안 서버와 호스트간의 네트워크 문제는 문제되지 않아 네트워크 부하를 줄일 수가 있다. 다시 말하면, 객체를 보낼 때만 네트워크가 연결되면 그 후에 네트워크가 끊어져도 작업은 계속 수행되며 작업이 끝난 후 객체는 네트워크가 다시 연결되기를 기다려 서버에게 돌아올 수 있게 된다.

웹 상에는 많은 검색 사이트가 있고 저마다의 기술로 엔진을 두고 자료를 저장하고 사용자에게 보여준다. 대부분이 공개하지 않는 하부 기술이어서 비교하기는 힘들지만 이동 에이전트를 이용하여 정보 수집을 하는 동안에 항상 서버를 바라보지 않아도 되어 네트워크의 부담을 줄이며 로컬 태스킹으로 인한 작업의 효율성 증대는 일반적인 검색 엔진보다 높은 효과가 있을 것으로 예상되며 또한 이 시스템은 인트라넷이나 어떤 네트워크 그룹의 분산된 여러 호스트에서 장시간 많고 복잡한 일을 수행하게 할 때에 탁월한 효과를 보리라 예상된다.

그리고 또한 사용자가 서버에게 위와 같은 이동 에이전트를 보내지 않더라도 사용자를 서버가 기억하게 함으로써 검색 도중 네트워크가 중단되어도 서버에서는 작업이 수행되어 그 결과가 저장됨으로써 나중에 사용자가 다시 접속하여 그 결과를 얻을 수 있는 시스템을 구현하였다.

이렇게 구현한 본 시스템은 검색 엔진으로서 자동 검색 로봇의 기능을 갖추고 있지 않다. 추후에 이동 에이전트 개념을 활용하여 서버가 주기적으로 검색 객체를 파견하고 이를 처리할 수 있도록 해야 한다. 그리고 사용자에게 보다 나은 서비스를 위하여 고급 데이터베이스 설계가 이루어져야 하고 또한 사용자가 서버를 통하여 객체를 파견하지만 사용자 중심의 시스템을 위하여 사용자의 브라우저에서 직접 객체를 보낼 수 있도록 하여 이동 에이전트를 응용한 검색 엔진의 새 장을 열고자 한다.

참 고 문 헌

[1] Mobile Agent 기술 동향, 주간기술동향, 1997.10

[2] Robert gray, "Mobile Agents: The Next Generation in Distributed Computing", Proceedings of the Second AIZU International Symposium on Parallel Algorithms/Architecture Synthesis, 1997, pp.8-10

[3] Marcelo Goncalves Rubinstein, "Service Location for Mobile Agent Systems", Proceedings of the SBT/IEEE International Telecommunications Symposium - Volume 2, 1998.9.8, pp.623

[4] 최일훈, "표준화된 객체지향 모델링 언어 UML", 마이크로소프트웨어, 1998. 12. pp.374-382, 1999. 2. pp.354-362

[5] Martin Fowler with Kendall Scott, "UML Distilled: Applying the Standard Object Modeling Language", Addison Wesley, 1997

[6] Terry Quatrani, "Visual Modeling With Rational Rose And UML", Addison Wesley, 1998

[7] Hans-Erik Eriksson, Magnus Penker, "UML Toolkit", John Wiley & Sons, 1998

[8] Laura Lemay, Charles L. Perkins, "자바 21일 완성", 한국 썬 마이크로시스템즈 편역, 1996

[9] 이현우, 김형국, 홍성민, "Java Programming Bible ver. 2", 영진출판사, 1999

[10] 김종근, 홍준호, 송건철, "Oracle Bible ver. 8.x", 영진출판사, 1999

[11] <http://www.tri.ibm.co.jp/aglets/>

[12] <http://www.omg.org>

[13] <http://www.rational.com>

[14] <http://msdn.microsoft.com/library/devprods/vs6/visualj/vjref/>

[15] <http://java.sun.com/products/jdk/1.1/docs/api/>

[16] <http://www.oracle.com>



오 동 석

1991-1997 전북대학교 물리학과
1998-1999 전북대학교 대학원
영상공학과(정보통신공학
인터넷응용서비스연구실)
2000.2 전북대학교대학원 졸업예정
관심분야 : Mobile Agents, 무선
인터넷 (WAP), 데이터베이스



김 승 권

1991-1999 전북대학교 정보통신공학과
1999-현재 전북대학교 대학원 정보통신학과
관심분야 : Mobile Agents, 무선 인터넷(WAP), Fuzzy -SQL/데이터베이스



박 순 철

1900-1979 인하대 응용물리학과
1900-1991 미국 루이지애나 주립대
전자계산 박사
1991-1993 한국 전자통신 연구소
데이터베이스 연구실 과제
책임자

1993-현재 전자정보공학부 부교수

관심분야 : 데이터베이스, 네트워크 컴퓨팅, SAN(Storage Area Network),