사용자 호스트와의 단절 상태를 고려한 이동 에이전트 시스템의 구현

전호철°, 최중민
E-mail : {hcjeon, jmchoi}@cse.hanyang.ac.kr
한양대학교 전자계산학과

A Mobile Agent System for Handling User Host Disconnection

Hochul Jeon° and Joongmin Choi
Dept. of Computer Science and Engineering, Hanyang University

요 약

본 논문에서는 사용자 호스트와의 단절된 상태를 고려한 이동 에이전트 시스템의 구현에 대해 논한다. 기존의 이동 에이전트 시스템에서는 사용자의 호스트와 단절 혹은 연결된 상태에서 부여된 태스크의 수행 결과를 사용자에게 전달하는데 있어서 비용이 상당히 많거나 불가능하며, 또한 일방적인 통보 형식이므로 전달한 결과에 대해 동적으로 사용자와 상호 작용 할 수 없다.

본 논문에서 제안하는 시스템은 사용자 호스트와의 연결 상태에 상관없이 사용자에게 결과를 전달하고 전달한 결과에 대해 사용자와 동적으로 상호 작용하도록 구현하였다. 시스템은 IBM의 이동 에이전트 시스템인 AWB(Aglets Workbench)를 확장하였으며, 자바 메일 패키지(Java Mail Package)를 이용하여 작업 수행 결과를 사용자의 호스트나 개인 휴대 통신 기기 등을 통해 사용자에게 전달하고 사용자의 호스트가 이동 에이전트 시스템에 재접속하면 결과에 대해 상호 작용할 수 있도록 구현하였다. 이러한 시스템은 단절(disconnection) 상태에서도 수행이 가능한 이동 에이전트의 장점을 보다 더 증가 시킨다.

1. 서론

이동 에이전트란 사용자를 대신하여 독자적인 의지와 지능을 가지고 네트워크를 이동하면서 주어진 업무를 수행하는 소프트웨어다[1]. 이러한 이동 에이전트는 그 특성에 따라 이동 에이전트가 수행할 수 있는 환경을 제공하는 이동 에이전트 시스템을 필요로 한다.

기존의 이동 에이전트 시스템들로는 플랫폼 독립적인 자바(Java) 언어로 작성된 이동 에이전트 시스템들이 주류를 이루고 있으며, 대표적으로 IBM의 AWB(Aglets Workbech), ObjectSpace의 Voyager, General Magic의 Odyssey 등이 있다.

IBM사의 AWB는 Aglets(Agent Applets)라는 이동 에이전트의 개발툴(Development Toolkit)을 제공하는 이동 에이전트 시스템이다. 이는 자바 애플릿(Java Applets)의 모델을 그대로 이어받아 에이전트가 설계되어 자바 보안 관리자(Java Security Manager)에 대응하는 자체의 Aglet 보안 관리자(Aglet Security Manager)[3]를 두어 에이전트로부터 호스트를 보호하는 방법을 제시하는 이동 에이전트 시스템의 가장 일반적인 모델이다[4]. General Magic사의 Odyssey는 기존에 General Magic사에서 개발했던 Telescript를 자바에 근거하여 다시 만든 것으로 이는 오딧세이 클래스 라이브러리를 제공하며, 사용자는 이를 이용하여 자신의 이동 에이전트 응용을 작성한다. ObjectSpace사의 Voyager는 자체 ORB를 사용하여 동적 링크를 가능하게 하며, Corba에서 지원하는 여러 서비스를 제공하고 있다. 이들 시스템들은 이동 에이전트의 이동성 제공에 초점이 맞추어져 사용자로 하여금 특정 호스트에 접속하거나 또는 지속적으로 연결을 유지하도록 함으로써 비용이 많이 든다. 사용자 호스트의 전원이 꺼져 있거나 혹은 사용자 호스트내의 이동 에이전트 시스템이 사용 가능하지 않은 경우는 사용자가 부여한 작업에 대한 수행 결과를 사용자에게 전달할 방법을 제공하지 못하며, 또한 결과를 이용해 다른 작업을 수행할 수 있도록 결과에 대해 사용자와 이동 에이전트간에 동적으로 상호 작용하는 방법을 제공하지 않고, 새로운 이동 에이전트를 생성함으로써 비용이 증가 하게 된다.

본 논문에서 제안하는 시스템은 자바 메일 패키지(Java Mail Package)를 이용하여, 사용자의 호스트 또는 개인 휴대 통신 기기로 수행 결과를 전달하도록 구현하였기 때문에 사용자 스트가 단절된 상태일지라도 사용자는 개인 휴대 통신 기기를 통해 결과를 받아 볼 수 있으며, 이동 에이전트 시스템에 재접속시 대기하고 있던 이동 에이전트와 결과에 대해 상호 작용할 수 있다. 또한, 이동 목적지 결정에 있어서 작업 기반 어드레싱을 함으로써 사용자에게 위치 투명성을 부여하도록 구현했으며, 각 작업간의 관계 정보를 이용하여 순차/비순차적인 다중 작업의 병렬 처리와 순서가 내재되어 있는 작업 및 사용자 순서 정의에 따른 다중 작업의 병렬 처리가 가능하도록 구현하였다.

논문은 다음의 내용으로 구성된다. 2장에서는 본 논문에서 제시하는 시스템의 각 구성요소와 기능에 대해 기술하고, 3장에서는 시스템의 구현에 대해 설명한다. 4장에서는 결론과 앞으로의 연구 과제에 대해서 기술한다.

2. 시스템의 구성

본 논문에서 제안하는 시스템의 구조는 그림1과 같다.

2.1 인터페이스(Interface)

인터페이스는 사용자가 원하는 여러 가지 작업을 선택하거나 취소할 수 있도록 상호 작용 하는 기능을 하며, 인터페이스는 Aglet Server, 사용자 인터페이스(User Interface), 통보 Aglet(Notify Aglet)으로 구성된다.

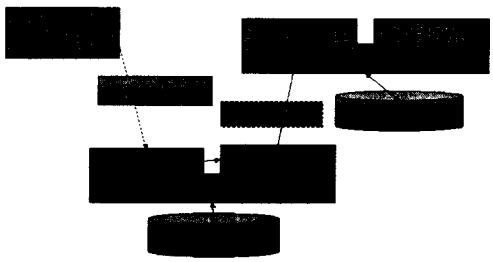

그림 1. 전체 시스템 구조도

사용자 인터페이스는 실제 사용자가 원하는 작업과 이에 대한 정보를 입력받는 GUI 부분과, 이러한 작업들을 적절히 처리할 수 있는 공급자의 주소를 문의하기 위해 관리자가 있는 호스트로 이동하여 관리자에게 문의하는 Aglet부분으로 구성된다. GUI부분은 사용자에게 공급자들에 대한 위치 투명성을 보장하기 위한 작업단위의 tree와 사용자가 선택한 작업들을 나타내는 부분, 작업에 따라 사용자로부터 정보를 입력받는 부분으로 구성된다. 통보 Aglet은 사용자의 Aglet Server가 시작될 때 생성되며, 관리자의 호스트로 이동하여 이전에 사용자가 위탁한 작업 결과의 존재 여부를 문의하는 기능을 한다.

2.2 공급자(Supplier)

공급자는 외부에서 유입된 Aglet이 요구하는 서비스의 제공과 Aglet이 활동할 수 있는 환경을 제공하며, Aglet Serve, 서비스 공급자(Service Supplier), 데이터들을 저장하고 있는 데이터베이스로 구성된다.
각 서비스 공급자는 생성시 자신이 제공할 수 있는 서비스 또는 처리 가능한 작업에 대한 정보와 서비스 공급자의 주소, 이름 등 자신에 대한 정보를 관리자에게 등록 요청하며, 유입된 Aglet과 상호 작용하는 기능을 한다.

2.3 관리자(Manager)

관리자는 외부에서 유입된 Aglet이 요구하는 서비스를 적절하게 처리할 수 있는 공급자들을 주소 저장소(Address Pool)에서 찾아 새로운 Aglet을 생성하여 해당 공급자들로 이동 시킨다.
관리자는 Aglet Server, 주소 관리자(Address Manager), 주소 저장소(Address Pool)로 구성된다.

2.3.1 주소 관리자(Address Manager)

주소 관리자의 기능은 첫째, 서비스 공급자들의 등록 요구에 따라 서비스 공급자의 정보를 주소 저장소에 저장한다. 둘째, 사용자로부터 주어지는 관련 정보를 이용하여 사용자가 요구한 작업들 중 연관성이 있는 작업들을 따로 분류하며, 이들의 순차적 수행을 위해 모니터를 생성한다. 셋째, 모니터/중재자로부터의 작업 결과를 주소 저장소 내 사용자 테이블에 저장하며, 이때 사용자의 전자 우편 주소를 함께 저장함으로써 사용자들을 식별한다. 넷째, 사용자의 Aglet Server가 이용 가능할 때 관리자 호스트로 유입되는 통보 Aglet이 사용자의 이전 작업 수행 결과의 존재 여부를 문의하면 통보 Aglet이 제공하는 사용자의 전자 우편 주소를 기초로 SQL문장을 생성하고, 질의 문의 결과를 전달한다.

2.3.2 중재자(Coordinator)

중재자는 비순차적인 작업들의 병렬처리를 위해 각 작업에 대한 서비스 공급자들의 주소 단위로 Aglet을 생성하고, 이들로부터 얻어지는 결과들을 정렬하여 사용자에게 적합하다고 여겨지는 결과를 선택해 전자 우편을 통해 전달한다. 중재자는 선택된 결과들을 전달한 Aglet들에 대해 대기 하도록 메시지를 전달한다. 사용자의 통보 Aglet으로부터 사용자가 선택한 결과에 대한 정보가 전달되면 중재자는 이러한 정보를 기초로 대기 중에 있는 Aglet들에게 사용자가 취한 행동을 전달한다. 각 Aglet들은 전달된 사용자의 행동에 따라 수행하고 그 결과를 중재자에게 다시 전송한다.

2.3.3 모니터(Monitor)

모니터는 관련 있는 다중 작업들을 순차적으로 처리하기 위해 Aglet들간의 조정 기능을 수행하며, 각 작업 단위로 Aglet을 생성한다.
각 Aglet은 서비스 공급자의 Aglet Server로 이동 후 서비스 공급자에게 작업 처리를 요구하며, 결과를 모니터에게 전달한 후 공급자의 Aglet Server에서 대기하며, 이는 순차적으로 진행된다. 모든 작업이 끝난 후 모니터는 사용자에게 전자 우편을 통해 결과를 전달하고 관리자 Aglet Server에서 대기한다. 관련 정보는 내부적으로 정해지거나 또는 사용자가 관련 여부를 지정해 주는 두
가지 경우가 있다. 사용자가 관련 여부를 지정해 주는 경우 사용자는 자신이 선택한 작업들 중에서 관련 있는 작업의 번호와 관련지은 작업들 내에서의 순서 정보를 입력한다. 모니터는 이러한 정보를 이용하여 순차적으로 작업을 처리 한다.

3. 시스템 구현

이 절에서는 시스템의 순차/비순차적 다중 작업, 사용자 호스트와의 단절 상태에서의 결과 전달과
결과에 대한 동적 상호 작용에 대해 구현하였다.
구현은 전자 상거래의 도서 구매와 판매, 국내·외 여행, 영화/연극/식당 예매, 주식 거래 아이템에 적용하였으며, 전체 시스템은 ASDK 1.0.3을 기반으로 작성하였다.

3.1 등록

그림 내의 1번 과정으로 모든 서비스 공급자는 생성시 반드시 거치는 과정이며, 제공하는 서비스 종류와 제공하는 서비스에 대한 세부적인 정보 및 서비스 공급자의 주소, 이름 등 서비스 공급자 자신에 대한 정보를 주소 관리자에 메시지 패싱(Message Passing)을 통해 전달하며,
서비스 공급자로부터 등록 요구가 있을 때 주소 관리자는 SQL문장을 생성한다.

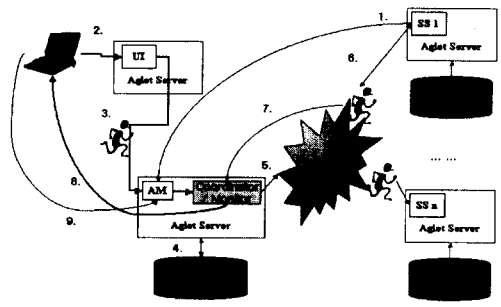

그림 2. 시스템 흐름도

3.1 등록

그림 내의 1번 과정으로 모든 서비스 공급자는 생성시 반드시 거치는 과정이며, 제공하는 서비스 종류와 제공하는 서비스에 대한 세부적인 정보 및 서비스 공급자의 주소, 이름 등 서비스 공급자 자신에 대한 정보를 주소 관리자에 메시지 패싱(Message Passing)을 통해 전달하며, 서비스 공급자로부터 등록 요구가

있을 때 주소 관리자는 SQL문장을 생성한다. 각 서비스 공급자들이 제공하는 정보는 서비스 단위로 주소 저장소에 저장된다.

3.2 작업 선택

그림 내의 2번 과정으로 사용자가 GUI를 통해 원하는 작업들을 선택 후 Aglet을 전송하며, 이때 사용자는 Aglet이 작업 결과를 전달 할 때까지 Aglet Server의 유지 또는 단절을 선택할 수 있다. 사용자로부터 입력받는 정보가 작업마다 다르기 때문에 각 작업을 선택할 때마다 각 작업에 맞는 프레임이 나타나게 된다.

사용자는 순서없이 여러 개의 작업을 입력하여 이들간의 관련성 여부를 설정 할 수 있도록 구현하였으며, GUI는 그림3과 같다.

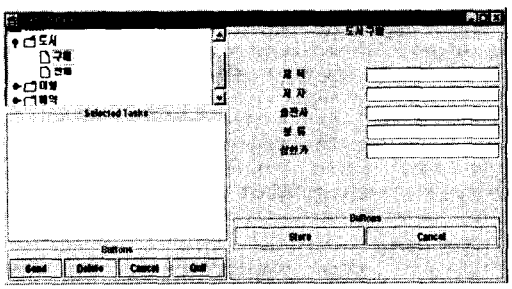

그림 3. 사용자 인터페이스

3.3 다중 작업

주소 관리자는 외부로부터 유입되는 Aglet 단위로 쓰레드(thread)를 생성하여 요구를 받아들이도록 구현하였다.

순차적 다중 작업은 Aglet 단위의 쓰레드가 작업 단위의 쓰레드를 생성한다. 이러한 작업 단위의 쓰레드들은 먼저 각 작업의 관련성 여부를 판단하여 관련있는 작업들에 대해 하나의 모니터를 생성하도록 구현했으며, 그렇지 않은 작업들은 작업 단위로 중재자를 생성하도록 구현하였다.

3.3.1 순차적 다중 작업

순차적 다중 작업은 두 가지 경우가 있다. 첫째 사용자가 임의로 여러 작업들에 대해 순서를 부여한 경우가 있고, 둘째, 여행과 같이 내부적으로 목적지에 따라 교통편이 결정되고, 그 이후 숙박장소가 결정되는 등의 순서가 존재하는 작업이 있다.

사용자가 임의로 여러 작업들에 대해 순서를 부여하는 경우에는 사용자가 작업들간의 관련성과 수행 순서를 정해 주도록 구현하였다. 이러한 구현은 상대적으로 적은 학습능력과 지식을 지니는 이동 에이전트의 특징을 고려한 것이다.

3.3.2 비순차적 다중 작업

중재자는 서비스 공급자 단위의 Aglet을 병렬적으로 생성하며, 각 Aglet은 주어진 서비스 공급자에게 이동한다.

각 Aglet은 작업 결과를 중재자에게 전달한 후 대기하며, 이때 중재자는 결과를 정렬하여 사용자에게 적합한 정보를 제공하는 Aglet에게 대기하도록 하며, 나머지 Aglet은 소멸된다.

3.4 단절 상태를 고려한 결과 전달

모니터 또는 중재자는 수행 결과를 사용자에게 전달하기 위해 먼저 사용자 호스트의 사용가능 여부를 ping을 통해 확인하며, 가능한 경우 사용자의 호스트로 결과를 전달하고, 불가능한 경우 사용자가 지정한 전자 우편 주소로 자바 메일 패키지를 이용하여 전달 하도록 구현하였다.

전자 우편 주소는 하나 이상을 사용할 수 있으며, 개인 휴대 통신 기기 또한 가능하다. 사용자는 결과를 받고자 하는 전자 우편 주소를 사용자의 개인 휴대 통신 기기의 전자 메일 주소로 지정함으로써 사용자 호스트의 전원을 끄더라도 결과를 알 수 있게 된다. 또한 전자 우편을 통해 결과를 전달함으로써 사용자는 작업 수행이 끝나는 시기를 알 수 있을 뿐아니라 이동 에이전트 시스템에 재접속하여 결과에 대한 상호 작용할 시기를 결정할 수 있다.

3.5 결과에 대한 동적 상호 작용

개인 휴대 통신 기기 또는 사용자가 지정한 전자 우편 주소를 통해 위탁한 작업의 결과를 전달 받은 사용자는 자신의 Aglet Server를 이용할 수 있게 함으로써 통보 Aglet을 생성 하도록 구현하였다. 통보 Aglet은 주소 관리자의 Aglet Server로 이동하여 사용자의 전자 우편 주소를 이용하여 이전에 사용자가 위탁한 작업에 대한 결과를 관리자의 Aglet Server로 가져와서 사용자와 상호 작용한 결과를 관리자의 Aglet Server에 대기하고 있던 모니터 또는 중재자에게 수행을 요청한다. 이때 모니터나 중재자는 사용자의 전자 우편 주소로 사용자의 통보 Aglet임을 식별하도록 구현하였다. 수행 요청을 받은 작업에 대해 모니터 또는 중재자는 대기 중인 Aglet들과 동적으로 연결하여 전달하도록 구현하였다.

4. 결론 및 향후 과제

본 논문에서 제안한 이동 에이전트 시스템은 사용자 호스트의 Aglet Server의 가용상태 또는 호스트의 가용상태와 상관없이 사용자에게 수행 결과를 통보할 수 있도록 함으로써, 결과 전달 방법에 있어 융통성을 지니도록 하였다. 사용자 호스트의 Aglet Server가 가용해질 때 대기 중에 있는 Aglet과 동적으로 수행 결과에 대해 상호 작용할 수 있도록 함으로써 결과를 활용하여 다른 작업을 용이하도록 하였다.

현재 Sun사에서 개발 중에 있는 Java Phone과 TTS(Text-to-Speech) 기법을 활용, mail과 함께 사용자에게 음성으로 결과를 통보해 줄 수 있도록 할 것이며, 동적 링크시 Corba의 Naming Service를 활용토록 하는 연구가 필요하다.

[참고 문헌]

[1] Colin G. Harrison, David M. Chess and Aaron Kershenbaum, "Mobile Agents : Are they a good idea ?," IBM Research Report, Rc 19887, from Internet, http://www.research.com/iagents/publications.html, 1995.

[2] Danny B. Lange and Mitsuru Oshima, "Programming and Deploying Java Mobile Agents with Aglets," Addison-Wesley Pub., 1998

[3] Bill Venners, "Under the Hood : The Architecture of Aglets," from Internet, http://javaworld.com/javaworld//jw-04-1997/jw-04-hood.html, 1997

[4] Joseph Kiniry and Daniel Zimmerman, "Special Feature : A Hands-On Look at Java Mobile Agents," IEEE Internet Computing, Vol. 1, No. 4, July-August 1997

[5] Daivid Kots, Robert Gray, Saurab Nog, Daniela Rus, Sumit Chawla, and George Cybenko, "AgentTcl : Targeting the Needs of Mobile Computers," IEEE Internet Computing, Vol. 1, No. 4, July-August 1997

[6] David Wong, Noemi Paciorㄷㅏ, and Dana Moore, "Java-based Mobile Agents," Communications of the ACM, Vol. 42, No. 3, March 1999