이동 에이전트 시스템

Mobile Agent System

金 坪 中*1
Kim, Pyeong Jung

尹 錫 煥*2
Yoon, Seok Hwan

ABSTRACT

As Information society evolves, there is a growing need that substitutes agents for users to do the given jobs in distributed computing environments. Agents can retrieve and browse the diverse information via Internet, as the users request, and can deal with the complicated routine jobs on behalf of users with timeliness and efficiency. Especially, mobile agent can support the paradigm for distributed application programs in Internet environments. This is suitable to jump-to-jump model without any hesitations because agent moves. Agents can be applied to various applications, but we restrict it to market information retrieval in this paper. On recognizing a query about merchandise information by an user, agent by itself retrieves the corresponding information by navigating the Internet and informs the result to the user. In this paper, we review the mobile agent configuration to do this kind of work, propose the system architecture based on this review. By using this system, users can be free from complicated routine jobs and pursue his/her wishes to promote ones welfare.

*1 전자계산조직응용기술사, 한국전자통신연구원 선임연구원.

*2 품질관리기술사, 공학박사, 한국전자통신연구원 책임연구원.

1. 서론

최근 컴퓨터 기술과 통신 기술의 급속한 발전에 힘입어 이들의 주된 기능을 결합한 새로운 형태의 컴퓨팅 서비스가 생성하고 있다. 이의 한 가지 형태인 휴대용 네트워크 컴퓨터나 PDA (Personal Digital Assistant) 등의 이동 호스트(mobile host)는 원하는 장소로 자유롭게 이동하면서 사용자의 요구에 따라 작업을 수행할 수도 있고, 필요한 경우 인터넷(Internet)에 접속하여 원하는 작업을 수행할 수도 있다. 이렇게 무선 통신 기술과 고속 통신망 기술 등을 결합한 새로운 컴퓨팅 환경을 이동 컴퓨팅(mobile computing)이라 하며[Brown 96], 이는 호스트에 이동성이 추가된 분산 컴퓨팅의 한 확장형으로 볼 수 있다.

이동 호스트는 필요하면 언제, 어디서든지 인터넷에 접속하여 작업할 수 있다는 장점 때문에 점차 널리 이용되고 있다. 사용자의 요구에 따라 여행 정보나 주식 정보 등을 인터넷에서 검색하여 보여 주거나 복잡한 일상 생활을 시기 적절하고, 효율적으로 대신 처리하도록 에이전트에게 위탁하여 처리할 수 있다.

에이전트는 보는 시각에 따라 다양하게 정의되고 있으나, 특정 목적의 업무를 수행하기 위하여 사용자를 대신하여 작업을 수행하는 자율적인 프로세스(autonomous process)라는 정의가 지배적이다[Franklin 96]. 에이전트는 수동적으로 주어진 작업만을 수행하는 것이 아니고, 자신의 목적을 가지고 그 목적 달성을 추구하는 능동적인 자세를 가진다. 또한, 사람 또는 조직의 권한(authority)을 대신하여 약 1주일 정도까지는 혼자 독립적으로 수행될 수 있고, 다른 에이전트와 만나서 상호 교류하기도 한다[White 96].

이동 에이전트(mobile agent)는 자신이 수행을 시작한 시스템에 묶여 있지 않는 에이전트를 말한다[White 96]. 고정 에이전트(stationary agent)에 비교해 보면, 서로 다른 시간에 서로 다른 시스템에서 수행될 수 있는 경우이다. 이동 에이전트는 이질적인 망(heterogeneous network)에서 자신의 제어로 호스트 사이를 이주(migration)하고, 각 호스트의 다른 에이전트와 상호 동작하거나 자원을 이용하여 맡겨진 임무를 수행하고, 수행이 끝났을 경우 고향(home)으로 되돌아온다.

에이전트가 인터넷을 돌아다니며 수행되기 위해서는 코드뿐만 아니라 상태도 이동되어야 한다. 코드는 이동될 모든 호스트에 똑같은 형식으로 동작되어야 하고, 직접 해석(interpretation)되거나 다시 컴파일(recompilation) 하지 않고 수행될 수 있는 이식성 있는 중간 언어(portable intermediate interpreter-based language)로 작성되어야 한다. 상태는 지속성(persistency)이 있어야 한다. 이동 코드의 수행 위치, 중간 결과 등을 계속 유지해야 하기 때문이다. 기존에 수행되었던 곳에서부터 다시 수행 되도록 하기 위하여 지속성 상태를 코드의 부분으로서 표현되거나 부호화된 스택(encoded stack)으로 전송되어야 한다.

통신 형태에 따라 분류하면, 이동 에이전트는 요구-지원 모델(client-server model) 보다는 오히려 대등 관계 모델(peer-to-peer model)이다. 각 에이전트는 경우에 따라 요구자 또는 지원자 역할을 수행할 수 있다. 예를 들면 사용자의 요구를 담은 에이전트가 이것을 처리해줄 수

있는 지원자 시스템으로 직접 가서 서비스를 받은 후 그 수행 결과를 요구자에게 다시 가져오는 형태이다. 이 때 사용자 요구를 담은 에이전트는 요구-지원 모델의 요구자이든 지원자이든 특정 역할로만 동작되지 않고, 단지 대등 관계 형태로서 동작한다.

이동 에이전트 시스템은 분산 응용 프로그램 작성에 용이한 형태를 제공하고 특히, 이동 호스트가 포함된 경우 아주 효율적인 수행 환경을 제공한다[Gray 97]. 에이전트가 언제, 어디로 이동한다고 기술되어 있으면 시스템이 자체적으로 전송하기 때문에 분산 응용 프로그램을 작성하는 데에 편리하고, 효율적이며, 강력한 프로그래밍 형태를 제공한다. 에이전트가 이동하기 때문에 자연스럽게 Jump-do-jump 모델에 적합하다.

이하에서는 이동 에이전트 시스템의 분류 및 개발 동향, 기본 개념, 구성요소에 대해 살펴보며, 시장 정보 검색을 위한 응용 및 이동 컴퓨팅 환경에 적합한 이동 에이전트 시스템의 구조 모델을 제시한다.

2. 개발 동향

이동 에이전트는 RPC(Remote Procedure Call)와 RP(Remote Programming)의 확장형으로 볼 수 있다. RPC는 요구자가 표준화된 procedure호출 기법을 사용하여 제공자 시스템의 오퍼레이션을 호출하는 것이며, RP는 요구자가 하나의 서브프로그램(subprogram)을 서버에게 보내고, 그 서브프로그램은 서버에서 수행되게 되고, 그 결과가 다시 요구자에게 보내지는 것이다. 이동 에이전트는 RP를 확장한 개념으로서 에이전트를 임의의 장소로 이동한다.

이동 에이전트 시스템은 이동 에이전트를 생성, 이동, 수행, 전송, 해석, 및 폐기 등 에이전트의 생명 주기를 관리할 수 있는 플랫폼으로서, 언어(language), 라이브러리, 인터프리터(interpreter), 프로토콜 및 보안(security) 등의 측면에서 분류할 수 있지만, 본 고에서는 언어 측면에서 분류하여 각각의 연구 동향을 살펴보기로 한다. 이동 에이전트 프로그래밍 언어는 자바(Java)에 기반한 이동 에이전트 시스템과 Java에 기반하지 않은 이동 에이전트 시스템으로 크게 분류할 수 있다.

〈표 1〉 언어에 따른 이동 에이전트 시스템의 분류

구분	Java-based Mobile Agent System	Non-Java Mobile Agent System
개발동향	Odyssey(General Magic)	Telescript(General Magic)
	Aglets Workbench(IBM Tokyo R. Lab)	Ara(University of Kaiserlauterm)
	Mobile Objects and Agents(OSF)	Tacoma(Univ. of Tromo & Cornel Univ.)
	Mole(University of Stuttgart)	Agnet Tcl(Cartmouth College)
	Concordia(Mitsubishi)	Obliq(DEC)

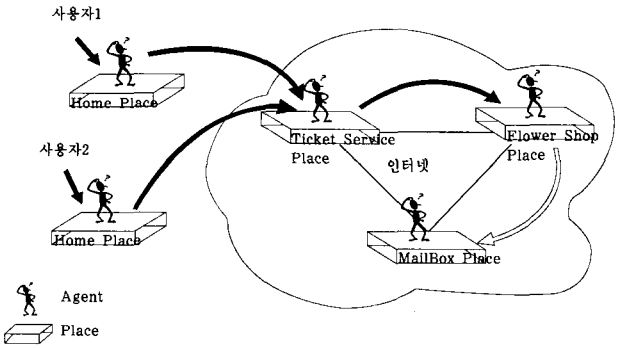
먼저, Java에 기반하지 않은 이동 에이전트 시스템은, 미국의 General Magic사가 전자 상거래를 위하여 Telescript 언어를 사용하여 개발한 Telescipt[White 96], 독일의 Kaiserslautern 대학에서 UNIX 시스템에 스크립트 언어인 TCL(Tool Command Language)/Tk를 이용하여 개발한 Ara[Peine 97], 노르웨이 Tromso 대학과 미국 Cornell 대학이 공동으로 UNIX 시스템의 Tcl/Tk에 근거하여 개발한 Tacoma, Dartmouth대학에서 정보 검색을 효율적으로 하기 위하여 Tcl/Tk를 이용하여 개발하고 있는Agent Tcl[Gray 97] 등을 들 수 있다.

Java에 기반한 이동 에이전트 시스템의 개발 동향을 살펴보기로 한다. Odyssey는 General Magic사가 Java를 적용하여 Telescript를 다시 개발하고 있는 것으로서, 사용자는 Odyssey class library를 이용하여 자신의 응용을 작성할 수 있다. IBM Tokyo Research Lab.에서 개발하고 있는 Aglets Workbench[Lange 96]는 네트웍 기반의 응용을 작성하는 데에 visual builder라는 시각적 환경을 도입하는 것이다. Aglet은 이동할 때 수행중인 자바 프로그램(코드와 상태)을 임의의 장소로 전송할 수 있기 때문에 자바 애플릿(Java applet) 모델을 확장한 것으로 볼 수 있다. OSF RI(Open Software Foundation Research Institute)에서 개발하고 있는 MOA[Milojicic 96] 프로젝트는 Java 객체의 체크 포인트 재시작(checkpoint restart), 통신 채널의 보존 방법 등에 대한 기술 검토를 마치고, 설계서가 작성된 상태이고, 올해 말에 1차 구현될 예정이다. 독일의 Stuttgart 대학에서는 Mole[straber 96]을 개발하고 있고, 일본의 Mitsubishi Electronic의 정보기술센터에서는 Concordia를 개발하고 있다. 이밖에도 많은 이동 에이전트 시스템이 개발되고 있는데 전반적인 동향은 Java 언어에 근거한 기반 구조(infrastructure)를 개발하는 데에 초점을 두고 있다.

3. 이동 에이전트의 개념

이동 에이전트는 독립적이고 자의적으로 이벤트가 발생했는지 등의 상태를 볼 수 있고(watch), 인터넷상에서 원하는 정보가 어디에 있는지를 이동하면서 찾을 수 있고(search), 여러 가지 서비스를 조화롭게 처리할(orchestrate) 수 있다. 이에 따른 응용은 전자 상거래(electronic commerce), 그룹 공동 작업(group collaboration), 이벤트 모니터링(event monitoring), 작업 흐름 자동화(workflow automation), 정보 검색(information retrieval), 망 관리(network management), 및 이동 컴퓨팅(mobile computing) 등이다.

분산 응용 서비스는 〈그림 1〉과 같이 에이전트의 집합으로 모델링할 수 있고, 각 에이전트는 생존의 터전인 플레이스(place)를 사용한다. 인터넷은 광활한 정보의 보고로서, 플레이스의 집합으로 모델링할 수 있다.



〈그림 1〉 분산 응용 서비스를 위한 에이전트 집합

에이전트(Agent)는 인간 또는 조직을 위해 자동적으로 행동하는 프로세스로서, 자신의 의도에 따라 수행할 수 있도록 쓰레드(thread)를 갖는다. 이동 에이전트는 수행을 시작한 시스템을 벗어나는 에이전트로서, 자신의 의지에 따라 한 플레이스에서 다른 플레이스로 옮기고, 서로 다른 시간에 서로 다른 플레이스를 사용(occupy)한다.

플레이스(Place)는 에이전트를 수행시키는 환경을 제공한다. 홈 플레이스(home place)는 이동 에이전트의 출발점과 도착점 기능을 제공한다. 플레이스는 이동 에이전트를 받아들이고, 수행 환경을 제공한다. 플레이스는 호스트 자원을 활용하도록 해주거나 사용자 인터페이스 등을 포함한다.

여행(Travel)은 에이전트가 한 플레이스에서 다른 플레이스로 이동하는 기능을 제공한다. 여행은 에이전트가 원격에서 제공된 서비스를 받기 위하여 필요할 때 이동하고, 서비스를 받은 후 홈 플레이스로 되돌아온다. 이를 위하여 절차와 상태가 이식성(portability)을 갖고 처리될 수 있어야 한다. 이것은 소프트웨어 다운로딩(downloading)과 다르다. 왜냐하면, 프로그램이 수행 중에 옮기기 때문이다.

모임(Meeting)은 같은 플레이스에서 두개 이상의 에이전트가 만나는 기능으로서, 에이전트들이 서로의 절차를 호출하는 것이다. 모임의 형태는 여러 형태가 존재한다. 첫째, 사용자 에이전트가 특정 서비스를 이용하기 위하여 특정 서버의 특정 플레이스로 여행한다. 특정 서비스를 제공하는 고정 에이전트(stationary agent)와 만나기 위함이다. 둘째, 2개의 이동 에이전트가 서로 만나기 위하여 같은 플레이스로 여행한다.

연결(Connection)은 서로 다른 플레이스에 있는 2개의 에이전트가 통신할 수 있도록 선로를 열어준다. 연결은 주로 대화식 응용의 사용자가 종종 사용한다. 예를 들면, 극장 표를 찾는 이동 에이전트가 홈 에이전트에게 유용한 좌석을 보여주는 극장 다이아그램을 보내면, 홈 에이전트는 사용자에게 평면도를 보여주고, 사용자가 선택한 좌석의 위치를 이동 에이전트에게 보내는 경우에 사용한다.

권한(Authority)은 에이전트 또는 플레이스가 다른 것들과 분별하기 위한 것이다. 예를 들어, 파일 접근 통제를 하려면, 파일 서버는 파일을 열람하거나 삭제하는 프로시쥬어의 권한을 알아야 통제를 할 수 있다. 권한을 사용하여 다음과 같은 제한을 할 수 있다. 플레이스는 들어오기를 시도하는 에이전트의 권한(agents authority)을 식별하여 특정 권한을 가진 에이전트들만 받아들이도록 할 수 있다.

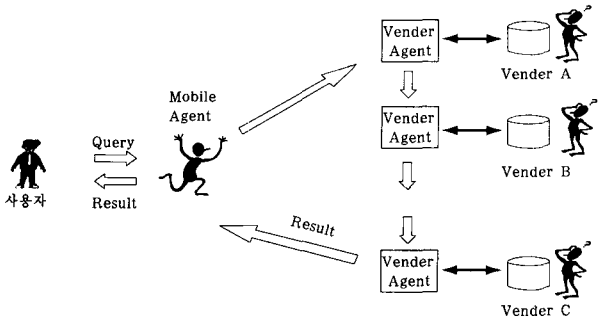
영역(Region)은 같은 권한(same authority)에 의하여 운영되는 플레이스들의 집합이다. 에이전트의 권한(agents authority)이 목적지 영역(destination region)에 만족하지 못하면 그 에이전트가 들어오는 것을 거부한다.

4. 시장 정보를 검색하는 이동 에이전트

시장 정보를 검색하는 이동 에이전트는 사용자가 시장에 가서 물품 구매를 하기 직전 상태를 시뮬레이션 한다. 물품 정보 검색에 대한 사용자의 입력을 받아 사용자를 대신하여 인터넷 상을 돌아다니며 정보를 검색하여 주는 종합 정보 시스템의 형태이다.

〈그림 2〉는 사용자가 시장 정보를 검색하기 위하여 질문(query)을 했을 경우, 이동 에이전트가 직접 시장을 돌아다니면서 요구된 정보를 검색하여 결과를 사용자에게 되돌려 준다.

〈그림 2〉를 좀더 상세히 기술하면, 사용자는 이름, 주소 등의 개인 정보와 원하는 물품 정보를 사용자 에이전트(Personal Agent)에게 알린다. 사용자 에이전트는 인증 절차를 마친 후, 사



〈그림 2〉 시장 정보 검색하는 이동 에이전트

용자가 요구한 일을 수행하는 이동 작업 에이전트(mobile task agent)를 생성시킨다. 이동 작업 에이전트는 각 벤더 에이전트(vender agent) 시스템을 방문하여 벤더 에이전트와 ACL (Agent Communication Language)을 사용하여 통신한다. 작업을 마치면 다른 벤더 에이전트 시스템을 방문하여 검색하고, 원하는 결과를 얻었을 경우, 원래의 홈 에이전트(home agent)인 사용자 에이전트에게 결과 값을 가지고 되돌아온다.

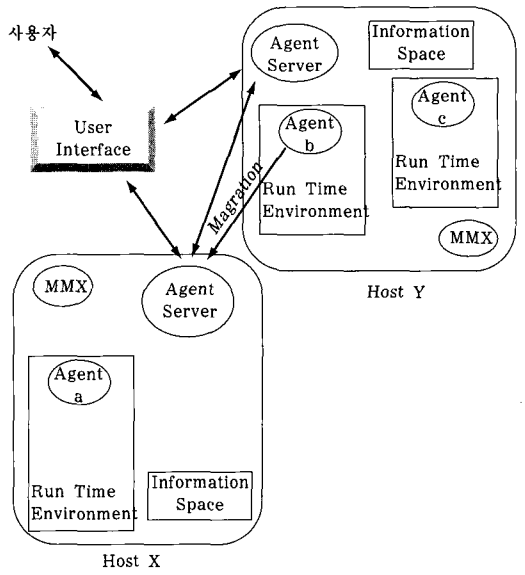
이때, 중요한 것은 에이전트의 이동성과 에이전트 간의 통신 및 중개(brokering) 기능이다. 중개 기능은 원하는 물품을 공급하는 대상을 찾거나 수요하는 거래 대상을 찾는 것이다. 이를 위하여 중개 에이전트(brokering agent)를 사용한다. 중개 에이전트는 물품 도매인 별로 벤더 에이전트들의 주소 및 이름을 저장하고 있는 계층적 구조의 데이터베이스를 가지고 있다. 중개 에이전트는 벤더 에이전트와 구매자 에이전트와 통신한다. 벤더 에이전트는 자신의 기능을 등록시켜 중개 에이전트의 데이터베이스에 반영하고(register/unregister), 구매자 에이전트는 자신이 구매하고자 하는 물품들을 취급하는 벤더 에이전트들의 목록을 요구하는 메시지(request /reply)를 보낸다.

5. 이동 에이전트 시스템의 개념 구조 모델

에이전트 시스템은 에이전트의 생성, 해석, 수행, 폐기 등의 생명 주기를 관리하는 플랫폼이다. 이동 에이전트 시스템은 에이전트 시스템에 추가적으로 에이전트의 이동성과 통신 기능, 허가 받지 않은 에이전트로부터의 접근 제한 등의 기능을 제공한다. 기본적인 이동 기능은 Jump 명령어를 사용한다. 이것은 하나의 플레이스에서 다른 플레이스로 에이전트를 이주시킨다. 이때 에이전트의 내부 상태를 수집하여, 이것을 부호화하고, 목적지 플레이스의 AS에게 보낸다. AS는 에이전트를 받아들여 보안 검색을 한 다음, 인터프리터를 시작시킨다. 인터프리터는 상태 이미지를 복구하고 Jump 명령어 바로 다음에서부터 에이전트를 수행시킨다.

이동 에이전트 시스템의 주요 하부 기능은 첫째, 호스트 시스템의 자원을 안전하고 효율적으로 사용하기 위하여 호스트와 상호 동작을 해야 하고, 둘째, 사용자의 요구에 따라 서비스를 제공하기 위하여 다른 에이전트와 협동 작업을 할 수 있어야 한다. 셋째, 이질적인 망을 옮겨 다닐 수 있어야 한다. 넷째, 에이전트의 생명 주기를 관리할 수 있어야 한다. 〈그림 3〉은 이러한 이동 에이전트 시스템의 기본적인 개념 구조를 보여 준다.

〈그림 3〉에서 개념 구조는 AS(Agent Server), 플레이스(place) 개념을 구현한 RTE(Run-Time Environment), MMA (Mir-



〈그림 3〉 이동 에이전트 시스템의 개념 구조 모델

ror Master Agent), IS(Information Space) 및 UI(User Interface) 등으로 구성된다.

AS(Agent Server)는 모든 호스트에 동작되어 이동 에이전트를 관리한다. 즉, 이동 에이전트를 받아들이고(accept), 적당한 실행 환경인 RTE를 생성하여 자기 시스템에 정박시키고(launch), 그들의 수행을 감시하는(supervise) 역할을 수행한다. 이 때 AS는 IS의 AMS에게 에이전트의 관리를 보고한다. AS는 IS(DF, ANS, AMS)를 통하여 호스트 시스템에 있는 모든 에이전트들을 접근할 수 있다. 또한, AS는 에이전트와 그 에이전트의 소유주와의 통신, 에이전트들 사이의 통신, 보안(security) 등의 역할을 수행한다.

RTE(Run-Time Environment)는 플레이스(place) 개념을 구현한 실체로서, 이동 에이전트의 실행 환경을 제공한다. RTE는 에이전트가 호스트 시스템의 자원을 안전하게 접근하도록 해야 한다. RTE는 에이전트를 수행시키기 위하여 Information space를 접근하거나 실행 중간 결과 또는 질문을 owner에게 보내기 위하여 AS, 사용자 등과의 통신 등을 관리한다.

IS(Information Space)는 에이전트의 생명주기를 관리한다. IS는 멀티에이전트 시스템의 DF(Directory Facilitator), ANS(Agent Name Server), AMS(Agent Management Server) 등을 포함한다. DF는 하나의 도메인 내의 디렉토리를 관리하는 에이전트이다. ANS는 한 플랫폼 상에 있는 모든 에이전트들에 대하여 이름과 주소 사이를 매핑하는 기능을 갖는 에이전트이다. AMS는 에이전트가 생성, 제거 및 이동되었을 경우 동적으로 등록하여 관리하는 에이전트이다.

MMA(Mirror Master Agent)는 이동 호스트와 인터넷 사이의 무선 연결을 효율적으로 관리하는 에이전트이다. 인터넷과 이동 호스트 사이의 연결이 안될 경우, 이동 호스트로 가는 에이전트는 그 이동 호스트의 거울 시스템의 큐(queue)에 저장된다. MMA는 무선 연결을 감시하다가 연결되었을 경우 큐에 있는 모든 에이전트를 이동 호스트로 보낸다. 반대의 경우, 이동 호스트의 큐에 저장되어 있는 모든 에이전트들을 연결되자 마자 인터넷으로 보내는 역할을 담당하는 에이전트이다.

UI(User Interface)는 사용자와 AS간의 인터페이스를 제공한다. 사용자의 요구에 따라 시스템이 수행할 수 있도록 이동 에이전트를 시스템에게 제출하고, 그래픽 인터페이스를 제공한다. 그래픽 인터페이스는 AS와 상호 동작하여, 이동 에이전트의 수행 상태를 보여줄 수 있다.

6. 결 론

이동 에이전트의 응용 분야는 새로운 응용 분야를 개척하는 것이 아니고, 기존의 분산 응용을 이동 에이전트를 통하여 효율적이고 편리하게 제공할 수 있다는 점이다. 또한, 에이전트가 이동하기 때문에 보다 확장성(scalable) 있는 응용을 작성할 수 있다.

이동 에이전트는 요구-지원 모델 보다 오히려 대등 관계 모델이기 때문에 다음과 같은 장점을 갖는다. 첫째, Jump-do-Jump 형태의 분산 응용 프로그래밍에 적합하다. 둘째, 요구-지원 모델에서 요구자는 제공자가 정의한 고정 서비스만 사용할 수 있지만 이동 에이전트는 수행 능력과 어느 정도의 지능이 있는 코드가 직접 가서 수행하기 때문에 융통성을 높여준다. 셋째, 전체적인 시스템 부하의 균형을 촉진할 수 있다. 이동 에이전트가 시스템 부하를 고려하여 라우팅 한다면 시스템 부하가 적은 곳에서 수행하기 때문이다. 넷째, 망 대역폭(network bandwidth)을 효율적으로 이용할 수 있다. 예를 들어, 기상 정보 데이터를 메시지 형태로 가져와서 분석하는 것보다 에이전트가 데이터 있는 곳으로 직접 가서 분석한 다음 그 결과만 본다면, 망 대역폭을 훨씬 절약할 수 있기 때문이다. 다섯째, 분산 응용이 이동 컴퓨팅 환경에서도 효율적으로 수행할 수 있는 환경을 제공한다. 이동 호스트에서 이동 에이전트를 보내놓고 이동 호스트의 전원을 끈 다음 결과를 보고자 할 경우 다시 연결하여 볼 수 있다.

이동 컴퓨팅을 위한 이동 에이전트 시스템은 분산 응용 서비스를 융통성 있고 효율적으로 제공한다. 특히, 이동 호스트가 망에 연결되는지의 여부에 관계없이 에이전트의 이동성을 보장하기 위하여 망을 감시하고 라우팅 할 수 있어야 한다. 이러한 투명한 이동성을 바탕으로 이동 에이전트 시스템은 이동 컴퓨팅 환경에서 분산 응용을 효율적으로 수행시킬 수 있다.

참 고 문 헌

[Brown 96] K. Brown and S. Singh, "A Network Architecture for Mobile Computing", IEEE INFOCOM96, San Francisco, California, Mar. 1996, pp.1388-1396.

[Chess 95] D. Chess, B. Grosof, C. Harrison, D. Levine and C. Parris, "Itinerant Agent for Mobile Computing", IBM Research Report RC 20010, IBM T.J. Watson Research Center, Mar. 1995, p.28.

[Franklin 96] S. Franklin and A. Graesser, "Is it an Agent, or just a Program? : A Taxonomy for Autonomous Agents", Proceedings of the 3rd International Workshop on Agent Theories, Architectures, and Languages, Springer-Verlag, 1996, p.10.

[Gray 97] R. Gray, D. Kotz, S. Nog, D. Rus and G. Cybenco, "Mobile Agents for Mobile Computing",

Proc. of the 2nd Aizu Intl Symposium on Parallel Algorithms/Architectures Synthesis (pAs 97), Fukushima, Japan, Mar. 1997, p.17.

[Harrison 95] C. G. Harrison, D. M. Chess, and A. Kershenbaum, "Mobile Agents: Are they a good idea?", Technical Report, IBM T. J. Watson Research Center, Mar. 1995, p.21..

[Genesereth 94] M. R. Genesereth and S. P. Ketchpel, "Software Agents", *Communications* of the ACM, Vol. 37, No. 7, Jul. 1994, pp.48-53.

[Lange 96] D. B. Lange and D. T. Chang, "IBM Aglets Workbench : Programming Mobile Agents in Java, A White Paper", IBM White paper, http://www.trl.ibm.co.jp/aglets/whitepaper.html, Sep. 1996, p. 9.

[Lingnau 95] A. Lingnau and O. Drobnik, "An Infrastructure for Mobile Agents: Requirements and Architecture", Proc. of 13th DIS Workshop, Orlando, Florida, Sep. 1995, p.9.

[Milojicic 96] D. S. Milojicic, M. Condict, F. Reynolds, D. Bolinger, and P. Dale, "Mobile Objects and Agents", 2nd USENIX Conference on Object Orien-ted Technologies and Systems (COOTS) : Distributed Object Computing on the Internet, Toronto, Canada, Jun. 1996. p.4.

[Peine 97] H. Peine and T. Stolpmann, "The Architecture of the Ara Platform for Mobile Agents", Proceedings of the First Intl Workshop on Mobile Agents (MA97), Berlin, Germany, Apr. 1997. p.12.

[Straber 97] M. Straber, J. Baumann and F. Hohl, "Mole-A Java Based Mobile Agent System", Proceedings of the 2nd ECOOP Workshop on Mobile Object Systems, Jul. 1996. p.12.

[White 96] J. White, "Mobile Agents White Paper", General Magic White paper, 1996, p. 28.

(원고 접수일 1997. 9. 22)