전자상거래 시스템 구축을 위한 자바 애글릿 기반 설계

Java Aglet-based Design for Electronic Commerce System

김평중
충북과학대학 컴퓨터정보과학과 교수
김정호
한밭대학교 정보통신 · 컴퓨터공학부 교수
박진양
인하공업전문대학 컴퓨터정보과 교수

Phyoung-Jung Kim
Professor, Dept. of Computer Information Science, Chungbuk Provincial University of Science and Technology
Jeong-Ho Kim
Professor, Div. of ICAE., Hanbat National University
Jin-Yang Park
Professor, Dept. of Computer and Information System, Inha Technical College

중심어 : 전자상거래, 이동에이전트, 자바애글릿

요 약

전자상거래는 글로벌 생산과 분배 공급 체인을 통합하고 최대화하도록 기회를 제공한다. 고객 선호도와 요구 변화에 따라 신속한 대처 능력과 새로운 기술의 활용 능력은 가장 중요한 요소이다. 다양한 회사의 컴퓨터들은 상호 통신하면서 컴포넌트의 가격과 유용성을 결정하고, 주문과 확인을 수행하고, 배송 시간을 협상한다. 본 논문에서는 이동 에이전트인 자바 애글릿을 기반으로 상거래 시스템 구축을 기술한다. 구매자를 대신한 애글릿들과 판매자를 대신한 애글릿들은 상품 거래소 시장으로 보내진다. 시장에서 자율적으로 주문과 배송을 협상하고, 최종 매매 결과와 실행코드를 구매자 및 판매자에게 되돌려온다. 우리는 전통적인 소매 시장을 시뮬레이션 함으로써 전자상거래 시스템이 자바 애글릿 기술을 이용하여 적절히 구축될 수 있음을 보여준다.

Abstract

Electronic commerce offers the opportunity to integrate and optimize the global production and distribution supply chain. Rapid response to changes in demand and customer preference, and the ability to exploit new technologies, are becoming critical. The computers of the various corporations communicate with each other to determine the price and the availability of the components, to place and confirm orders, and to negotiate delivery time scales. In this paper, We describe a trading system that is based on mobile agent technology, called aglet. Aglets for the buyer and sellers are dispatched to the various marketplace, where they negotiate autonomously orders and deliveries, returning to the buyer and seller with their best deals for approval. We show that the electronic commerce system is feasibly built by using the Java aglet technology to demonstrate for simulating a traditional retail marketplace.

I. 서론

인터넷과 World Wide Web(WWW)은 전자상거래를 촉진시키고 있다. 상품을 보여주고 주문 받는 웹 기술은 비즈니스의 공통적인 양식이 되고 있다. 전자상거래는 컴퓨터 네트워크를 통하여 정보, 제품, 서비스 등을 팔고 사는 행위로 정의할 수 있고, 정보통신기술을 활용한 모든 종류의 거래 행위들을 포함한다. 전자상거래는 글로벌 생산과 분배 공급 체인을 통합하고 최대화하도록 기회를 제공한다. 고객 선호도와 요구 변화에 따라 신속한 대처 능력과 새로운 기술의 활용 능력은 가장 중요한 요소이다. 다양한 회사의 컴퓨터들은 상호 통신하면서 컴포넌트의 가격과 유용성을 결정하고, 주문과 확인을 수행하고, 배송 시간을 협상한다[1,2].

웹 기술을 이용하여 인터넷 세상에서 사용자가 원하는 상품을 효율적으로 검색하여 구매 또는 판매하는 것은 일반화된 현실로 받아들이고 있다. 상품 검색을 용이하게 할 수 있

도록 상품 비교 사이트가 출현하는가 하면, 개인들의 상품 매매를 위한 경매나 역 경매 사이트들의 수도 급속하게 늘어가고 있다[3,4].

이동 에이전트 기술은 실행 코드가 목표 달성을 위해 호스트를 이동하며 작업을 수행한다. 이동 에이전트는 네트웍 연결을 계속 유지한 상태로 작업하지 않고 실행 코드 자체가 시스템간을 이동하며 임무를 수행하기 때문에 네트웍 연결이 불안정하거나 부하가 많이 걸리는 환경에서 유용하게 활용될 수 있다[4,5,6,7].

본 논문에서는 이동 에이전트인 자바 애글릿[8]을 기반으로 상거래 시스템 구축을 기술한다. 구매자를 대신한 애글릿들과 판매자를 대신한 애글릿들은 상품 거래소인 시장으로 보내진다. 시장에서 자율적으로 주문과 배송을 협상하고, 최종 매매 결과와 실행코드를 구매자 및 판매자에게 되돌려온다. 우리는 전통적인 소매 시장을 시뮬레이션 함으로써 전자상거래 시스템이 자바 애글릿 기술을 이용하여 적절히 구축될 수 있음을 보여준다.

본 논문의 구성은 다음과 같다. II장의 전자상거래 시스템의 이동 에이전트 활용에서 자바 기반 이동 에이전트 시스템을 비교 분석하고 전자상거래 시스템에 적용된 대표적인 예를 고찰하고 III장에서는 자바 애글릿 이동 에이전트 시스템을 간단히 기술하였다. IV장에서 이를 기반으로 전자상거래 시스템 구축을 이동 에이전트 기반으로 설계하고 간단히 시범 구현하였다. 마지막으로 시범 구현한 전자상거래 시스템을 고찰하고 향후 연구 과제에 대하여 기술한다.

II. 전자상거래 시스템의 이동 에이전트 활용

최근 들어 전반적인 이동 에이전트 시스템의 경향은 자바 언어에 근거한 기반 구조를 개발하는 추세이다. 대표적인 예를 들어보면, 자바를 포함한 다수의 언어를 지원하는 Ara[9], D'Agent [5], Tacoma[10]가 있으며, 자바 언어만을 기반으로 하는 IBM Tokyo Research Lab.(TRL)의 애글릿(aglets)[8], Mitsubishi의 Concordia[11], ObjectSpace의 Voyager[12], General Magic의 Odyssey[13] 등이 있다. 대표적인 특징별 비교는 다음과 같다.

이동 에이전트 시스템들은 이동 기법에 따라 점프(jump) 모델과 진입점(known entry-point) 모델 등을 사용하고 있다. 첫째, 점프 모델은 시스템이 점프라는 프리미티브 연산을 제공한다. 이 연산은 자동적으로 에이전트의 코드, 객체 상태, 및 제어 상태를 포착하여 다른 시스템으로 보내고, 도착 시스템에서는 상태를 복원하여 점프한 지점에서부터 실행을 계속한다. 이러한 시스템은 D'Agent[5], Ara[9], Tacoma[10] 등이 있다. 둘째, 진입점 모델은 에이전트의 코드와 객체 상태만을 포착하여 다른 시스템으로 보내고, 도착 시스템에서는 진입점(특정 메소드)에서 실행을 계속한다. 이러한 시스템으로 Aglets[8], Concordia[11], Voyager [12], Odyssey[13] 등이 있다. 자바 기반 이동 에이전트 시스템은 자바가상머신에 아무 변경 없이 실행시키는 진입점 모델을 사용하고 있다. 점프 모델이 에이전트 프로그래머에게 훨씬 편리하지만, 제어 상태를 포착하는 루틴이 기존 인터프리터에 추가되어야 한다는 단점이 있다.

표 1. 자바 기반 시스템의 특징별 비교

시스템 / 특징	애글릿[7]	Voyager[12]	Odyssey[13]
Remote creation of agents	No	Yes	No
Remote Java message	None	State-of-the-art	None
Messaging to mobile agents	None	Transparent	None
Messaging mode	Sync, Future	Oneway, Future, sync	None
Life spans	Explicit	5 different modes	Explicit
Mobile naming service	None	Integrated	None
Move to program	Yes	Yes	Yes
Move to object	No	Yes	No
Itineraries	Yes, special API	Yes, no special API	Yes, special API
Connectivity	Restricted	Full	Restricted
Security	Security manager	Security manager	None
Partial connection	No	No	No

이동 에이전트 시스템은 악의를 가진 에이전트로부터 각 호스트를 방어해야 하기 때문에 보안이 중요하다. 이동 에이전트 소유주와 보낼 호스트는 디지털 서명을 하여 전송하고, 서명을 검증한 후 접수 또는 거부한다. 서명과 이동 약력에 따라 자원 접근을 할당하고 안전한 수행 환경에서 이동 에이전트를 실행시켜야 한다. 그러나, Ara[9], Aglets[8], Odyssey[13] 등 대부분의 시스템들은 호스트의 그룹이나 에이전트를 위한 보호막을 제공하지 못하고 있다.

Voyager는 원격의 이동 에이전트 클래스를 생성할 수 있는데 비하여 애글릿나 Odyssey는 원격 생성이 불가능하다. Voyager에서 일단 클래스가 생성되면 네트웍상의 어디에서나 쉽게 객체를 생성할 수 있고, 그 객체에 자바 메시지를 보낼 수 있다. Voyager는 synchronous, oneway, 및 future message 모드를 제공한다. 객체가 이동하면 새로운 위치로 이동된 객체에게 메시지를 전송하기 위하여 이동하기 전의 위치에 secretary를 남겨두고 이동한다. 이에 비해 애글릿은 특정 URL에 존재하는 고정 에이전트에게 스트링 명령어를 보낼 수 있지만 그 명령어를 수행하는 책임은 에이전트에게 달려있다[14].

전자상거래 시스템은 간단한 온라인 쇼핑 시스템으로부터 종합적이고 복합적인 시스템까지 다양하다. 이 시스템은 고객 구매 행동 모델[15]의 6단계 구현에 따라 구분될 수 있고, 대표적인 전자상거래 시스템은 다음과 같다.

BargainFinder[16]는 본질적으로 데이터베이스 검색 엔진 접근방법으로서 여러 온라인 음악 상점에서 최선 거래(best deal)를 제공하고, 등록한 소매상인에게만 제한적으로 매매 중개 서비스를 제공한다. Jango[17]는 BargainFinder와 유사하지만 Excite 검색 엔진을 이용하여 가격 비교 쇼핑 서비스를 모든 사람에게 제공한다. 이러한 시스템들은 지능적인 검색 엔진을 사용하여 공급자 정보들을 수집하고 특정 사이트에서 서비스하는 형태이다. 이러한 시스템들은 사용자가 실제 상품을 구매하고자 할 때, 공급자 사이트를 브라우징 해야하고, 부품으로부터 조립해서 이루어지는 상품을 획득하기 어렵고, 세계적으로 널리 퍼져있는 공급 체인 상품을 획득하기 어렵다. Kasbah[2]는 상품을 팔고 살 수 있는 온라인 WWW 시장이다. 구매 사용자는 구매 에이전트를 생성하여 선택기준을 제공하고, 시장으로 보낸다. 선택기준은 가격, 시간 조건, 상품 수량을 포함한다. 공급 에이전트는 공통 게시판에 상품 판매 가격을 제시하고 기다린다. 구매 에이전트는 사용자의 선택기준에 따라 상품을 필터링하고 난 후 거래를 협상하게 된다. 구매 에이전트와 판매 에이전트는 하나의 시장 서버에서만 동작하고, 각각 협상 결과를 가지고 되돌아간다. Kasbah는 하나의 시장 서버에서 다른 시장 서버로 이동하지 못한다.

MAGMA(Minnesota Agent Marketplace Architecture)[18]는 인터넷에서 전송될 수 있는 아이템들을 대상으로 거래하는 가상 시장 프로토타입이다. 이것은 Java 기반 거래 에이전트(구매 에이전트와 판매 에이전트), 분류된 광고 서비스를 제공하는 광고 서버, 및 재정과 지불을 담당하는 은행으로 구성된다. 각 에이전트는 유일한 ID를 지정 받고, 에이전트간 메시지 경로설정을 위해 중계 서버에 등록한다.

MAgNET[6]는 구매자, 구매 이동 에이전트(자바 애글릿), 및 공급자로 구성된 전자상거래 시스템을 구축하고 있다. 기본 목적은 구매자와 판매자가 상호 관심이 있는 제품을 직접 거래하도록 한다. 구매 애글릿이 원하는 제품의 공급자를 접근하는 pull 모델에 근거하고 있다. 구매자는 상품에 따라 잠재적인 공급자 리스트를 갖고, 공급자에게 가서 최선 결과를 거래한 후 결과를 구매자에게 보고한다.

III. 자바 애글릿 이동 에이전트 시스템

자바 애글릿[8]은 IBM TRL에서 개발한 것으로서 처음 개발된 자바 기반 시스템 중의 하나이다. Concordia[11], Voyager[12,14], Odyssey[13]등의 상용 시스템에서처럼 에이전트 이동시 에이전트의 쓰레드 상태를 포착하지 못한다. 쓰레드 상태를 포착하기 위해서는 표준 자바 가상 머신을 변경해야 하기 때문이다.

따라서 D'Agent[5]나 Ara[9]의 go() 보다는 오히려 Tacoma[10] 모델의 변종을 사용한다. 이동 에이전트의 수행은 이동한 후 잘 알려진 지점으로부터 재시작 한다. 특히 애글릿 시스템은 이벤트 모델을 사용한다. 에이전트가 이동할 때 dispatch() 메쏘드를 호출한다. 애글릿 시스템은 에이전트의 onDispatching() 메쏘드를 호출한다. 이 메쏘드는 응용 클린업을 수행하고 에이전트의 쓰레드를 죽이고 에이전트의 코드와 객체 상태를 직렬화하여 다른 시스템으로 보낸다. 에이전트를 받은 시스템에서 onArrival() 메쏘드를 호출한다. 이 메쏘드는 응용 초기화를 수행하고 에이전트의 수행을 재시작하기 위하여 에이전트의 run() 메쏘드를 호출한다.

이 밖에 애글릿은 하드디스크에 코드와 객체 상태를 저장하는 지속성 기능, 위치 투명성을 제공하기 위한 애글릿 객체의 대리자인 proxy 기능, 이동 에이전트를 찾는 룩업 기능, 에이전트간 통신을 위한 메시지 전달 기능 등을 포함하고 있다. 애글릿의 보안 모델은 D'Agent[5], Ara[9], 및 다른 자바 기반 시스템 등과 유사하고, 이동 에이전트가 특정 머신의 제어문맥에 들어오고자 할 경우 제어문맥은 인증 소유주와 제작자의 기준에 따라 자원 접근 면허를 준다.

IV. 전자상거래시스템의 자바 애글릿 기반 설계

전자상거래 시스템의 이동 에이전트 기반 설계는 구매자

에이전트, 판매자 에이전트, 구매 애글릿, 판매 애글릿, 및 시장 에이전트로 구성한다. 구매자 에이전트와 판매자 에이전트, 및 시장 에이전트는 사용자를 위한 고정 에이전트(stationary agent)이고, 구매 애글릿과 판매 애글릿이 이동 에이전트(mobile agent)이다. 이동 에이전트는 대응 관계(peer-to-peer) 모델이기 때문에 마스터(MMmaster)와 서버(MMserver)로 나누어 설계한다.

그림 1에서 이동 호스트에 마스터가 설치되고 이동 서버에는 서버가 설치된다. 공통적으로 자바 가상 머신(JVM)과 애글릿 서버가 설치되어 있다. 애글릿 서버는 ATP(Agent Transfer Protocol) 데몬, 애글릿 제어문맥, 및 애글릿 뷰어(Tahiti)로 구성된다[8]. ATP 데몬은 현재 동작중인 애글릿과 통신하거나 전송하기 위하여 ATP 메시지를 처리한다. 애글릿 제어문맥은 서버에 존재하는 애글릿을 수행시키며, 애글릿의 생명 주기를 제어한다. 애글릿 뷰어는 이동 에이전트 자바 클래스를 생성하여 동작시키는 사용자 인터페이스와 이동 에이전트의 생명 주기를 제어는 사용자 인터페이스를 갖는다.

MMmaster는 이동 호스트의 고정 에이전트로서 사용자와 인터페이스 하면서, MMslave의 인스턴스(instance)인 자바 애글릿을 생성시켜 이동 서버인 MMserver로 보내고, 처리 결과를 보고 받아 사용자에게 그 결과를 보여준다. MMslave는 MMmaster로부터 생성되는 자바 애글릿으로서 방문하고자 하는 서버 목록에 따라 사용자의 구매 또는 판매 물품에 대한 요구 사항을 싣고 이동 서버인 MMserver로 직접 이동하여 이동 서버에서의 처리 결과를 가지고 이동 호스트로 다시 돌아와 MMmaster에게 처리 결과를 보고한다. 한편, MMserver는 이동 서버에 존재하는 고정 에이전트로서, MMmaster가 보낸 MMslave가 도착하면 이를 대기 큐에 넣고 요구 사항에 따라 처리하고 그 결과를 MMslave에 넘겨준다.

그림 2는 이동 호스트에서 동작하는 MMmaster와 MMslave의 수행 흐름을 보여준다.

① Tahiti 사용자 인터페이스를 사용하여 MMmaster를 실행시킴과 동시에 MMwin 화면을 띄운다. MMwin 화면은 사용자 정보와 물품 정보의 입력을 받는다. 정보 입력을 마치면 buy 버튼 또는 sell 버튼을 누른다.

② 사용자가 buy 버튼 또는 sell 버튼을 누르면 MMmaster는 이동 에이전트인 MMslave 인스턴스를 생성시켜 목적지로 보낸다. 이 때 우선 순위에 따라 복수개의 MMslave 인스턴스를 생성시켜 복수개의 목적지로 보낼 수 있다.

③ MMslave 인스턴스가 임무 수행을 완료한 후 MMmaster로 되돌아와 수행 결과를 보고한다. MMmaster는 수행 결과를 MMresult 화면에 보여준다.

그림 3은 이동 서버에서 동작하는 MMserver와 MMslave의 수행 흐름을 보여준다. MMserver는 고정 에이전트로서 이동 에이전트인 MMslave로부터 사용자의 요구를 받아들여 전자상거래 서비스를 수행한다.

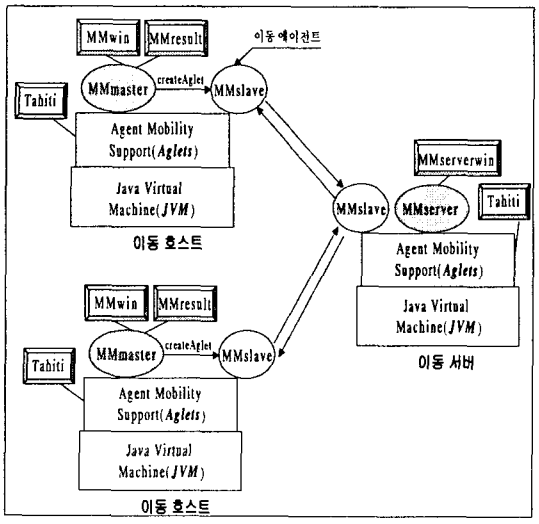

그림 1. 전자상거래 시스템의 애글릿 기반 전체적인 설계

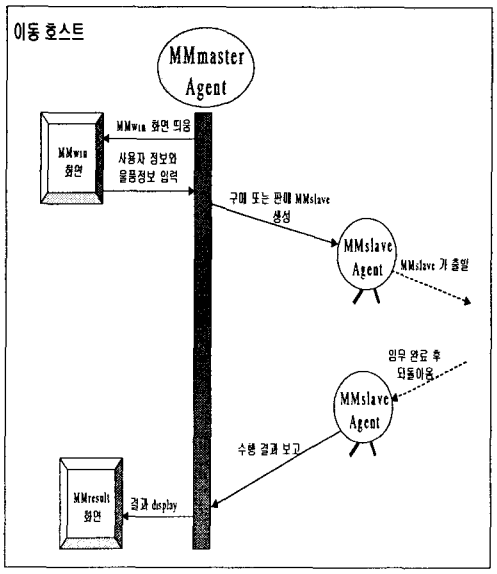

그림 2. 이동 호스트에서 MMmaster와 MMslave의 수행 흐름

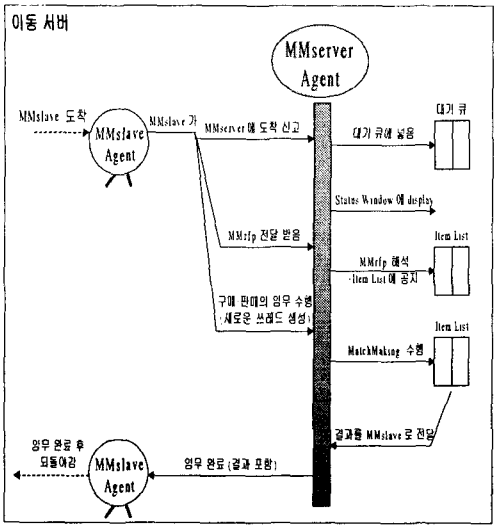

그림 3. 이동 서버에서 MMserver와 MMslave의 수행 흐름

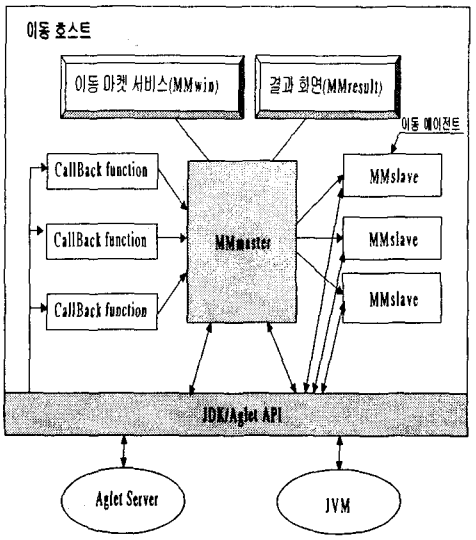

그림 4. MMmaster 수행 중 구성 요소들 간의 관계

① 이동 에이전트인 MMslave가 도착하면 MMserver에 도착 신고를 하고 대기 큐에서 기다린다. MMserver는 상태 윈도우에 MMslave의 상태를 보여준다.

② MMserver는 MMslave가 가져온 사용자 요구를 해석하여 Item list에 표시하고 새로운 쓰레드를 생성하여 Matchmaking을 수행하고 그 결과를 MMslave에게 보낸다.

③ MMslave는 수행 결과를 받으면 사용자 요구 사항과 함께 이동 호스트의 MMmaster에게 되돌아간다.

1. MMMaster

MMmaster는 사용자 인터페이스 MMwin으로부터 사용자의 요구 사항을 입력받고 Buy 버튼이나 Sell 버튼이 눌려지면 MMslave 인스턴스를 생성시킨다. MMslave로부터 수행 결과를 받으면 MMresult 화면을 띄워 임무 수행 결과를 사용자에게 보고한다. 그림 4는 MMmaster 클래스, MMwin 클래스, MMresult 클래스, 애글릿 서버가 제공하는 API, JDK1.1.8이 제공하는 API, 콜백 (callback)함수, MMslave간 수행 관계를 보여준다.

사용자가 xmas.demo.MMmaster를 Tahiti 화면에서 실행하면, MMmaster는 MMS 화면인 MMwin을 보여준다. MMwin 화면에서 사용자 이름, 소속, 전화번호, e-mail 등의 사용자 정보와 물품 종류, 제목, 수량, 단가 등의 물품 정보를 입력하면, MMmaster는 MMrfp 인스턴스를 생성시키고 Buy 버튼이

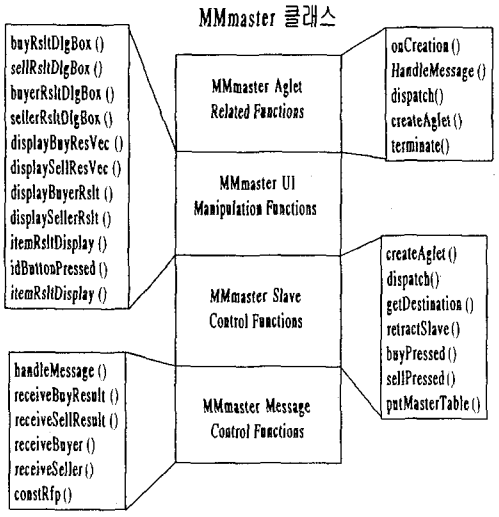

그림 5. MMmaster 클래스의 기능별 메쏘드들

나 Sell 버튼을 누르면 MMslave를 생성시킨다. MMmaster는 생성된 MMrfp 인스턴스를 이동 에이전트인 MMslave에게 전달하고 방문 목록에 등록된 곳 (MMserver)으로 dispatch한다. MMslave는 주어진 임무를 완수하고 홈 에이전트인

MMmaster에게 되돌아와 수행 결과를 보고한다. 이를 위해 MMslave는 MMmaster Id와 MMmaster URL를 갖고 있다. MMmaster는 생성된 MMslave의 정보(MMslave Id, MMmaster Id)를 관리하고, 수행 후 돌아와 앉아 있을 자리를 유지한다. MMslave가 자리에 앉아 있고 자기 차례가 되면 MMmaster에게 보고 메시지를 띄워 보고한다. MMmaster는 MMresult 화면을 통하여 수행 결과를 보여주고 MMslave를 폐기한다.

그림 5는 MMmaster 클래스의 메쏘드들을 보여 준다. MMmaster 클래스에는 애글릿 관련, 사용자 인터페이스, 메시지 제어, 및 MMslave 제어 관련 메쏘드들을 포함한다.

2. MMslave

MMslave 클래스는 이동 에이전트를 구현하는 것으로서 애글릿 서버에서 제공되는 API들을 기본적으로 이용한다. 주요 기능은 목적지에 따라 MMserver로 이동하여 부여된 임무를 수행하고, MMserver에 도착하여 등록하고, MMserver에게 가져온 MMrfp 정보를 전달하여 수행시키고, 그리고 수행 결과를 받아 MMmaster로 되돌아와 결과를 보고하는 기능 등을 갖는다. 그림 6은 MMslave 클래스의 기능별 메쏘드들을 보여 준다.

3. MMserver

이동 서버에 존재하는 MMserver는 그림 7과 같이 MMserver 클래스를 중심으로 구성된다. MMserver에서는 여러 개의 MMslave들이 이동 방문하여 사용자의 구매 또는 판매 정보를 주고받는다. 이동 호스트로부터 MMserver를 방문한 이동 에이전트인 MMslave들은 MMserver에 등록한 후 상태 목록에 표시한다. MMserver는 MMslave의 요구 정보에 따라 구매 MMslave이면 구매자 대기소, 판매 MMslave이면 판매자 대기소에 넣고 가져온 정보를 해석한다. 각 MMslave가 가져온 MMslave ID와 사용자 정보, 물품 정보들을 Item List에 추가하여 표시한다. MMserver는 MMslave가 가져온 물품 정보에 따라 MatchMaking을 처리한다. MMserver는 물품 정보가 일치되면 바로 그 결과를 MMslave에게 되돌려준다. MatchMaking은 각 MMslave가 도착할 때마다 수행되고 Item List에 표시한다. MMslave가 만족한 결과를 얻었을 경우, MMserver는 MMslave들을 홈 에이전트인 MMmaster에게 되돌아가도록 한다.

MMserver 클래스는 애글릿 서버에서 제공되는 API들을 기본적으로 이용하여 설계하였으며, MMserver 클래스의 기능별 메쏘드들은 그림 8과 같다.

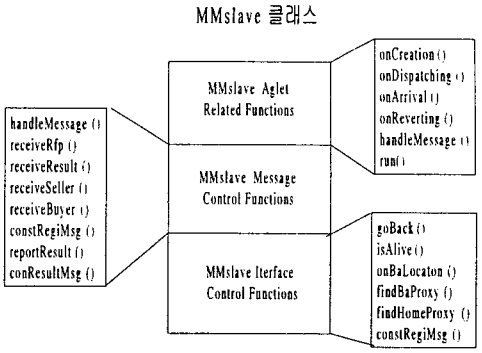

그림 6. MMslave 클래스의 기능별 메쏘드들

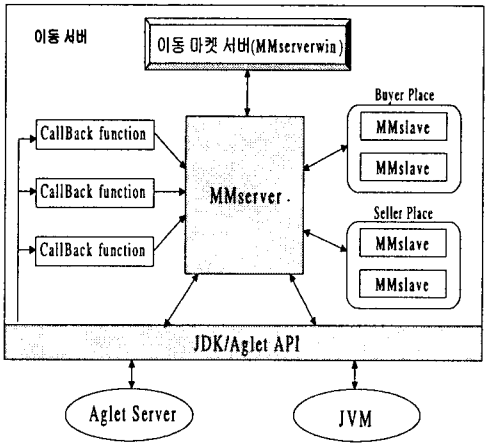

그림 7. MMserver 수행 중 구성 요소들 간의 관계

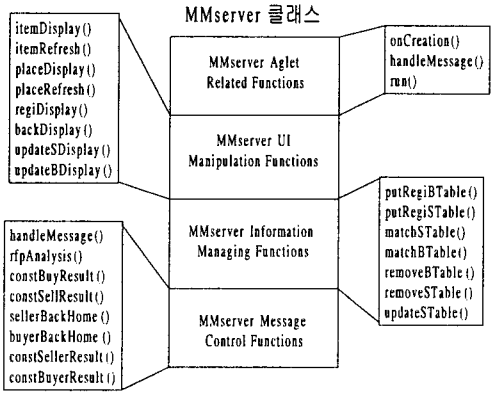

그림 8. MMserver 클래스의 기능별 메쏘드들

V. 전자상거래시스템의 이동에이전트 기반 시범구현

전자상거래 시스템은 운영 체제로 Windows NT 4.0, 프로그래밍 언어로 자바 JDK1.1.8, 이동 에이전트 시스템으로 IBM 애글릿 Workbench 1.0.3 버전을 사용하여 구현하였다. 구현 시스템은 애글릿 기반의 이동 에이전트 기술을 이용하여 MMS(Mobile Market System)를 구현하였다. MMS는 이동 에이전트가 사용자를 대신하여 사용자의 구매 또는 판매 정보를 가지고 시장에 돌아다니면서 적절한 구매자 또는 판매자를 찾아 관련 정보 및 상품을 협상하거나 교환하는 서비스이다.

MMS에서는 물건을 구매하거나 판매하고자 하는 사람을 MMmaster로, 구매자 또는 판매자들의 물품 매매를 대행하는 사람을 MMslave로, 물품의 매매가 이루어지는 시장을 MMserver로 시뮬레이션 하였다. MMserver에는 MMmaster들의 물품 매매 대리인인 MMslave들이 모여 매매를 체결하는 장소가 된다.

사용자는 그림 9에서와 같이 Tahiti를 통하여 이동 호스트의 MMmaster와 이동 서버의 MMserver를 각각 기동시킨다. MMmaster는 그림 10의 MMwin 화면을 띄우고 MMserver는 그림 11의 화면을 띄운다.

MMmaster 클래스는 이동 호스트에서 동작하는 모든 기능을 제어하는 핵심 클래스이다. 사용자가 MMS를 invocation하면 MMmaster 클래스의 onCreation() callback 함수가 불리고, 이 함수 내에서 MMwin 클래스를 호출함으로써 전자상거래 서비스 화면이 나타난다. 이 화면에서 Buy 버튼이나 Sell 버튼을 누르면 전자상거래 서비스 화면의 사용자 정보(사용자의 이름, 소속, 전화 번호)와 물품 정보(물품 종류, 물품 이름, 수량, 단가)를 MMrfp 인스턴스에 저장하여 MMmaster 클래스에게 전달한다. 이 클래스는 buyPressed()나 sellPressed() 함수를 호출하여 애글릿을 생성한다. 애글릿은 요구받은 정보를 받아 주어진 목적지로 실행 코드와 함께 이주한다.

이동 서버의 MMserver는 도착한 Buy 또는 Sell 이동 에이전트를 프로세스로 생성하여 초기화시키고 주어진 임무를 수행하도록 한다. 이동 서버에 도착한 이동 에이전트는 그림 11의 화면과 같이 Market Status List에 도착을 알리는 메시지를 보여 주고 구매 또는 판매 의뢰에 따라 Buying Item List 또는 Selling Item List에 물품 정보를 등록하고 새로운 쓰레드를 생성하여 요구된 작업을 수행한다.

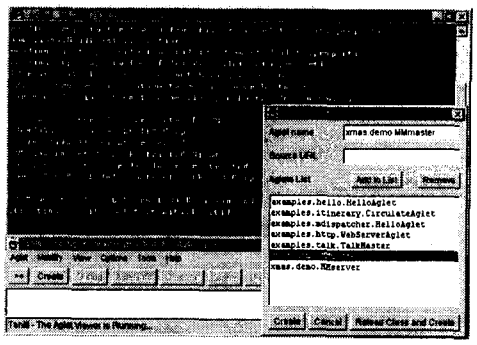

그림 9. MMmaster 실행

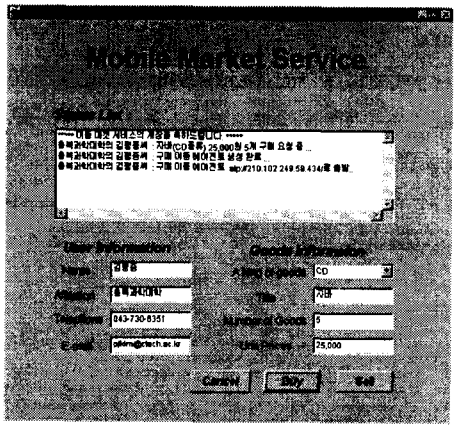

그림 10. MMwin 화면으로부터 정보 입력

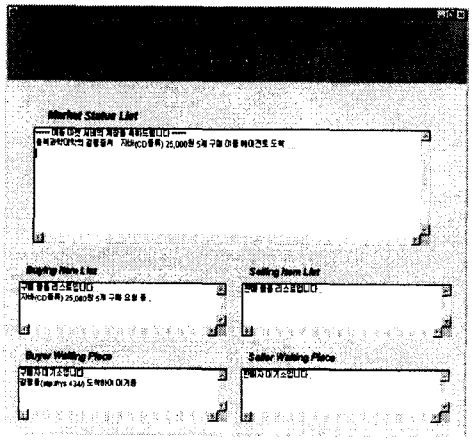

그림 11. 전자상거래 서버에 도착한 Buy 자바 애글릿

MMserver 클래스는 이동 서버에서 동작하는 모든 기능을 제어하는 핵심 클래스이다. 사용자가 전자상거래 서버 응용 프로그램을 실행시키면 MMserver 클래스의 onCreation() callback 함수가 불리고, 이 함수 내에서 MMserverwin 클래스를 호출함으로써 전자상거래 서버 화면이 나타난다. 그리고 run() callback 함수가 불린다. 외부로부터 메시지를 접수받기 위하여 handleMessage() callback 함수를 사용한다.

이동 서버에서 구매자와 판매자 사이에 매매가 체결되면 그 결과를 그림 12과 같이 Status List, Item List, Waiting Place에 적절한 메시지를 표시한다. 매매가 체결된 이동 에이전트들은 실행 결과를 갖고 자신의 홈 에이전트인 이동 호스트의 MMmaster로 되돌아간다. 이동 서버에서의 매매 체결은 구매자와 판매자의 상품 정보(상품명과 가격)를 비교함으로써 이루어진다.

Buy 또는 Sell 이동 에이전트가 처리 결과를 가지고 이동 호스트로 되돌아오면 MMmaster는 Status List에 상태를 표시하고 그림 13의 MMresult 화면에 매매 결과를 보여 준다.

VI. 결론 및 향후연구

본 논문에서는 전자상거래 시스템을 구축하기 위해서 자바 애글릿을 이용하여 설계될 수 있음을 보였다. 사용자를 대신하여 복수 개의 자바 애글릿들이 상거래 서버 시장으로 직접 이동하여 서비스나 상품을 자율적으로 협상하거나 매매하고, 그 결과를 사용자에게 실행 코드와 함께 결과를 보고하였다. 자바 애글릿은 실행 코드와 함께 사용자 요구사항을 가지고 직접 시장에 이동하기 때문에 이동 컴퓨팅의 전자상거래 시스템 구축에 용이함을 알 수 있었다. 이것은 전자상거래의 소비자 구매 행동 모델에서 2단계의 "상품 탐색(무엇을 살 것인가)"과 3단계의 "판매자 탐색(누구로부터 살 것인가)"에 해당되는 과정을 자바 애글릿 기술을 사용하여 구현한 것으로서, 특히, 사용자가 상품을 검색할 경우 복수개의 자바 애글릿을 생성하여 복수개의 전자상거래 서버로 병렬 전송하여 수집함으로써 병렬 정보 검색 가능성을 보여주고 있다.

향후 연구로는 전자상거래 서비스 중에서 판매자 탐색에서 가격과 수량만을 이용하였으나, 배달 조건 · 유지 보수 · 지불 방법 등을 고려한 4단계의 협상 과정을 추가 구현하여야 한다. 또한, 시범 구현된 전자상거래 시스템은 자바 애글릿 기술을 전자상거래에 적용할 수 있음을 보인 초보적인 서비스이며, 향후 사용자의 구매 패턴(구매 빈도가 높은 물품 종류, 협상 조건 등)을 이용한 이동 서버의 선택, 이동 에이전트의 이동 서버 탐색 방안에 대한 연구, 협상 과정의 구현, 웹 상의 쇼핑 사이트들로부터 상품 정보의 자동 추출, 필터링 기법 등의 연구를 진행할 예정이다.

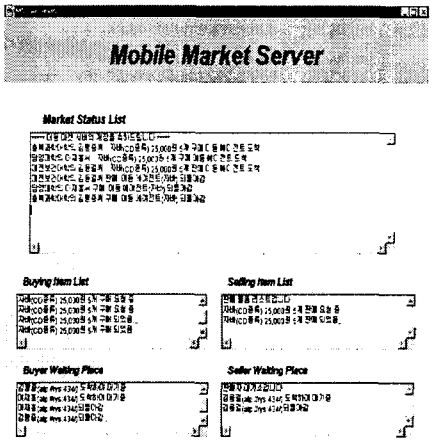

그림 12. 전자상거래 서버에서 매매 체결

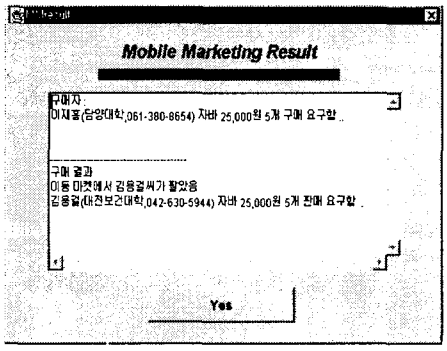

그림 13. 자바 애글릿의 수행 결과 화면

참 고 문 헌

[1] M. Bloch, Y. Pigneur, & A. Seegev, "On the Road of Electronic Commerce - a Business Value Framework, Gaining Competitive Advantage and Some Research Issues," http://www.hec.unil.ch/mblock/docs/roadtoed.ec.html, Mar. 1996.

[2] A. Chavez and P. Maes, "Kasbah: An Agent Marketplace for Buying and Selling Goods," Proc. of 1st International Conference on the Practical Application of Intelligent Agents & Multi Agent Technology, 1998.

[3] R. B. O. Doorenbos and D. Weld, "A Scalable Comparison-Shopping Agent for the World Wide Web," Proc. of the First International Conference on Autonomous Agents(Agents97), pp.39-48, 1997.

[4] Michael P. Wellman and Peter R. Wurman, "Real Time Issues for Internet Auctions," Proc. of the First IEEE Workshop on Dependable and Real_time E-Commerce Systems(DARE98), 1998.

[5] R. Gray, D. Kotz, S. Nog, D. Rus and G. Cybenco, "Mobile Agents for Mobile Computing," Proc. of the 2nd Aizu Int'l Symposium on Parallel Algorithms/ Architectures Synthesis (pAs97), p.17, Mar. 1997.

[6] P. Dasgupta, and et. al, "MAgNET: Mobile Agents for Networked Electronic Trading," IEEE Transactions on Knowledge and Data Engineering, Vol. 11, No. 4, Jul. 1999.

[7] Jae-Hong Lee and Pyeong-Jung Kim, "Mobile Agent System Architecture for the Partial Connection Problem Management," Proc. of International Conference on Advanced Ccommunication Technology (ICACT2000), pp.165-169, 2000.

[8] D.B. Lange and M. Oshima, Programming and Deploying Mobile Agents with Aglets, Addison-Wesley, ISBN: 0-201-32582-9, 1998.

[9] H. Peine and T. Stolpmann, "The Architecture of the Ara Platform for Mobile Agents," Proc. of the First Int'l Workshop on Mobile Agents(MA97), Apr. 1997.

[10] D. Johansen, F. B. Schneider and R. Renesse, "Operating System Support for Mobile Agents," Proc. 5th IEEE Workshop on Hot Topics in Operating Systems, 1998.

[11] D. Wong, and et. al, "Condordia: An Infrastructure for Collaborating mobile Agents," Proc. of 1st Int'l Workshop on Mobile Agent(MA'97), 1997.

[12] ObjectSpace, "Voyager : ORB3.0 Developer Guide," ObjectSpace Inc. http://www.objectspace.com/Voyager/, 1999.

[13] J. White, "Mobile Agents White Paper," General Magic Whitepaper, http://www.genmagic.com/agents/Whitepaper /whitepaper. html, p. 28,1996.

[14] ObjectSpace, "A Comparison : ObjectSpace Voyager, General Magic Odyssey, IBM Aglets," http://www.objectspace.com/ Voyager/, 1997.

[15] R. Guttman, A. Moukas, and P. Maes, "Agent-Mediated Electronics Commerce: A Survey," Knowledge Engineering Review, Vol.13, No.2, Jun. 1998.

[16] J. P. Bailey and Y. Bakos, "An Exploratory Study of the Emerging Role of Electronic Intermediaries," Int'l J. Electronic Commerce, Vol.30, No.3, Spring, 1997.

[17] R. Doorenbos, O. Etzioni, and D. Weld, "A Scalable Comparison/Shopping Agent for the World Wide Web," Proc. First Int'l Conf. Automous Agents, Feb. 1997.

[18] M. Tsvetovatyy, and et. al, "MAGMA: An Agent-based Virtual Market for Electronic Commerce," Applied Artificial Intelligence, Vol.11, No.6, Sep. 1997.

김평중(Phyoung-Jung Kim) 정회원

1985년 2월 충남대학교 계산통계학과 (학사)
1995년 2월 KAIST 대학원 전산학과(공학석사)
2000년 2월 충남대학교 대학원 컴퓨터과학(이학박사)
1995년 10월 전자계산기조직응용기술사 취득
1988년 2월 ~ 1998년 2월 한국전자통신연구원 멀티미디어연구부 선임연구원
1998년 1월 ~ 2000년 12월 한국정보처리학회 학회지 편집위원
1998년 3월 ~ 현재 충북과학대학 컴퓨터정보과학과 교수
<관심분야> : 이동에이전트, 전자상거래, 컴퓨터네트워크, 분산 멀티미디어

김정호(Jeong-Ho Kim) 정회원

1980년 2월 경북대학교
전자공학과(학사)
1983년 경북대학교 대학원
전자공학과 (공학석사)
1994년 단국대학교 대학원
컴퓨터공학과(공학박사)
1983년 3월 ~ 1996년 2월 한국전자통신연구소
(책임연구원, 실장)
1989년 정보처리기술사
1990년 공업계측제어기술사
1991년 정보통신기술사
1996년 3월 ~ 현재 국립한밭대학교 정보통신·
컴퓨터공학부 교수
<관심분야> : 데이터통신, 컴퓨터네트워크, 통신서비스

박진양(Jin-Yang Park) 정회원
현재 인하공업전문대학 컴퓨터정보과 교수