論文
3-2-10

Aglet을 이용한 웹 기반 병렬컴퓨팅 환경설계 (Design of Web-based Parallel Computing Environment Using Aglet)

김 윤 호*
(Yoon-ho Kim)

요 약

웹은 브라우저를 통한 단순한 정보의 전달과 정보의 공유수단으로서가 아니라, 수 많은 컴퓨터 자원이 연결되어 있는 병렬 컴퓨팅을 위한 기반구조로서 이용될 수 있는 잠재적인 가능성을 가지고 있다. 웹을 기반으로 한 병렬컴퓨팅의 접근방법은 기존의 다른 방법들에 비하여 일반 사용자들의 접근에 대한 용이성, 확장성, 비용대비 효과적인 병렬시스템 구축의 용이성, 기존의 네트워크를 활용할 수 있다는 측면에서 많은 장점을 가진다. 자바언어에서의 이동코드(mobile code)의 개념을 가지고 있는 applet은 많은 계산을 필요로 하는 프로그램이 독립된 병렬작업으로 분할되어 웹상의 여러 노드들로 이동되어 실행이 되어질 수 있는 가능성을 제시하여 주고 있다. 그러나 자바 applet은 보안에 대한 모델상의 제약으로 인하여 제한된 범위 내에서만 실행이 가능하며 클라이언트가 applet을 포함하고 있는 호스트들에 접속을 해야 한다는 점에서 유연성이 부족하다. 따라서 본 논문에서는 applet의 개념에 자치적으로 작업을 처리할 수 있는 기능을 추가하여 이동형 에이전트라 할 수 있는 Aglet(Agile applet)을 이용하여 웹 기반 병렬 컴퓨팅 환경을 설계하였으며, 웹 기반 병렬컴퓨팅 환경을 구축할 때 필요한 기술과 구조가 분석되었다. 또한 applet 기반의 방식과 비교하여 간단한 시뮬레이션과 분석이 이루어졌다.

ABSTRACT

World Wide Web has potential possibility of infrastructure for parallel computing environment connecting massive computing resources, not just platform to provide and share information via browser. The approach of Web-based parallel computing has many advantages of the ease of accessibility, scalability, cost-effectiveness, and utilization of existing networks. Applet has the possibility of decomposing the independent/parallel task , moving over network, and executing in computers connected in Web, but it lacks in the flexibility due to strict security semantic model. Therefore, in this paper, Web-based parallel computing environment using mobile agent, Aglet (Agile applet) was designed and possible implementation technologies and architecture were analyzed. And simple simulation and analysis was done compared with applet-based approach.

* 정회원 : 상명대학교 자연과학대학 소프트웨어학과 조교수

논문접수 : 2002. 2. 1.
심사완료 : 2002. 2. 23.

※ 본 연구는 2000년도 상명대학교 자연과학연구소 연구비 지원으로 이루어졌음.

1.서론

컴퓨터가 개발된 이래로 더 빠른 컴퓨터를 원하는 사용자의 요구와 이러한 요구에 부응하기 위해 컴퓨터의 성능을 향상시키려는 과학자와 공학자의 노력은 끊임없이 지속되어 왔으며, 앞으로도 계속 될 것이다. 컴퓨터 시스템의 성능을 향상시키려는 방법은 기본적인 하드웨어 소자의 성능 향상으로부터 시작해서, 시스템 소프트웨어의 성능 향상, 새로운 계산 모델에 바탕을 둔 시스템 구조와 조직을 통한 성능 향상 등의 여러 가지 접근 방법이 가능 하다[1]. 이러한 보다 더 빠른 고성능 컴퓨터를 만들려는 병렬처리에서의 연구는 네트웍 기술의 발전으로 인하여 MPI(Message Passing Interface), PVM(Parallel Virtual Machine), DSM(Distributed Shared Memory)시스템등의 네트웍 기반의 컴퓨팅환경으로 발전 연구되어져 왔다. 그러나 이러한 네트웍 기반의 컴퓨터 클러스터는 상용화된 기존의 컴퓨터들을 이용하여 경제적인 시스템을 구축할 수 있다는 장점을 가지나, 서로 상이한 시스템들로 구성이 될 경우에는 여러 시스템에서 컴파일, 디버깅을 해야 하며, 라이브러리 등을 해당 시스템에 맞게 각각 설치해야 하는 부담과 실행성능을 위하여는 네트워크상의 상이한 컴퓨터들에 대한 상세한 지식이 필요하다는 문제점이 존재한다. 또한 서로 상이한 시스템들로 인하여 하드웨어/소프트웨어에 대한 유지보수 및 관리비용이 높다는 경제적인 문제점도 수반하게 된다. 웹에 대한 폭발적인 성장과 초고속 통신망의 급속한 발전으로 인하여 웹은 브라우저를 통한 단순한 정보의 전달과 정보의 공유수단으로서가 아니라, 수 많은 컴퓨터 자원이 연결되어 있는 병렬 컴퓨팅을 위한 기반구조로서 이용될 수 있는 잠재적인 가능성을 가지고 있다. 웹을 기반으로 한 병렬컴퓨팅은 기존의 방법들에 비하여 일반 사용자의 접근에 대한 용이성, 확장성, 비용대비 효과적인 병렬시스템 구축의 용이성, 기존의 네트워크를 활용할 수 있다는 측면에서 많은 장점을 가진다. 따라서 접근성이 용이한 웹에 연결되어 있는 컴퓨팅 노드들을 대규모 병렬처리에 컴퓨팅자원으로 활용하려는 연구들이 활발히 진행되어지고 있다.또한 웹 기술의 근간을 이루고 있는 자바언어에서의 이동코드(mobile code)의 개념을 가지고 있는 applet은 많은 계산을 필요로 하는 프로그램이 독립된 병렬작업으로 분할되어 웹상의 여러 노드들로 이동되어 실행이 되어질 수 있는 가능성을 제시하여 주고 있다. 그러나 자바 applet은 보안에 대한 모델상의 제약으로 인하여 제한된 범위 내에서만 실행이 가능하며 클라이언트가 applet을 포함하고 있는 호스트들에 접속을 해야 한다는 점에서 유연성이 부족하다 [3, 9].

이에 대한 대안으로 일본 동경 IBM연구소에서는 applet의 개념에 자치적으로 작업을 처리할 수 있는 기능을 추가하여 이동형 에이전트라 할 수 있는 Aglet(Agile applet)을 개발하였다 [2,4,5]. Aglet은 한 컴퓨터로부터 다른 컴퓨터로 프로그램 코드와 상태나 데이터를 옮길 수 있는 경량 자바 객체(lightweight Java object)라 할 수 있다. 따라서 Aglet과 같은 이동형 에이전트의 이용은 다음과 같은 장점을 가질 수 있다. 첫째 네트워크의 부하를 줄일 수 있다. 주어진 작업을 처리하기 위하여 지역적으로 분산된 웹 상의 노드들간에 상호작용을 필요로 하게 되며 이는 결국에는 통신프로토콜에 의존하게 되는데 이동형 에이전트를 이용하여 처리를 하면 네트워크상의 부하를 줄일 수 있다. 두 번째로는 이동형 에이전트를 이용한 접근방법은 실시간 응용이 가능하다. 예를 들어 공장의 제조과정에서는 실시간의 제어가 필요하게 되며 네트워크를 통한 제어는 상당한 잠복시간을 야기시키는 반면 이동형 에이전트는 실시간 제어가 가능할 수 있다. 세 번째로는 병렬 분산환경에서 데이터가 교환이 되기 위하여는 프로토콜을 구현하는 코드가 분산환경에 설치되어야 한다. 그러나 이러한 프로토콜은 효율성과 보안의 문제로 인하여 점진적 발전을 하게 되며 이를 유지 보수해야 하는 문제는 새로운 문제를 야기시킨다.

그러나 이동형 에이전트는 기존의 프로토콜 기반 위에 새로운 채널(channel)을 설정하는 것이 가능하여 프로토콜을 포장화(encapsulation)함으로써 이러한 문제를 해결할 수 있다. 또한 이동형 에이전트는 비동기적이며 자치적으로 수행이 가능하며 동적으로 적응이 가능하기 때문에 applet과 비교할 때 더 유연적이며 더 많은 응용이 가능하다. 또한 웹상에서의 컴퓨팅노드는 본질적으로 하드웨어적으로나 소프트웨어적으로 이기종이 전제가 되며 이러한 환경에서 이동형 에이전트는 컴퓨터나 transport계층에는 독립적이며 단지 실행 환경에만 종속적이기 때문에 이동형 에이전트는 이음새 없는 유연한 최적의 조건을 제공할 수 있다.

본 논문에서는 Aglet의 개념과 구조, 이를 이용하여 웹 기반 병렬 컴퓨팅환경에 대한 개념적 설계와 문제점이 논의되고 간단한 시뮬레이션과 이에 따른 분석이 행하여진다. 본 논문의 구성은 다음과 같다. 2장에서는 Aglet의 개념과 구조가 논의되며 3장에서는 웹 기반 병렬컴퓨팅 환경에 대한 기술적 이슈와 Aglet을 이용하여 병렬 컴퓨팅 환경에 대한 개념적 설계가 이루어진다. 4장에서는 간단한 시뮬레이션과 분석이 applet기반의 방식과 비교하여 이루어진다. 마지막으로 5장에서는 결론과 향후 연구에 대하여 언급한다.

2. Aglet의 구조

Aglet은 인터넷상에서 한 호스트로부터 다른 호스트로 이동할 수 있는 자바 객체라 할 수 있다. 즉 한 호스트에서 실행을 하다가 갑자기 중단되어 원격지의 다른 호스트로 보내져서 실행을 계속할 수 있다. Aglet이 이동될 때 프로그램의 코드와 데이터의 상태가 이동될 수 있으며 신뢰 되지않는 호스트에 대하여는 내재된 보안 메커니즘을 통하여 안전성을 보장하게 된다. Aglet을 이용하여 이동형 에이전트를 구현하기 위한 자바기반의 framework으로는 IBM에서 개발된 Aglets Software Development Kit(ASDK)이 있으며 객체지향 프로그래밍 인터페이스와 한 호스트로부터 다른 호스트로 코드와 데이터 상태정보를 이동할 수 있는 메커니즘과 플랫폼에 독립적인 개발과 실행환경 및 보안 메커니즘을 제공한다. Aglet을 이용한 이동형 에이전트 개발에 필요한 인터페이스는 표준 API(Application Programming Interface)로 공개하고 있으며 이를 Java Aglets API(JAAPI)라 한다.

Aglet의 생명주기는 다음과 같다.

(1) Created : 새로운 Aglet이 생성된다. 이 상태는 초기화되고, Main이 되는 부분이 수행되기 시작하는 단계이다.

(2) Cloned : 현재 상태를 갖는 복제 본을 생성한다. 즉 동일한 Aglet을 하나 더 만드는 것이다.

(3) Dispatched : 현재의 상태를 유지하면서 새로운 호스트로 이동하게 된다.

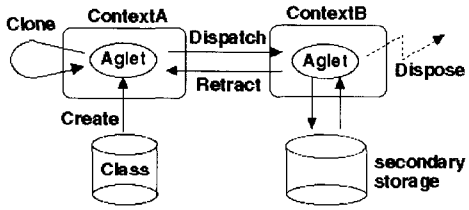
(4) Retracted : 현재의 상태를 가지고, 원격호스트로부터 바로 이전에 Dispatched된 호스트로 돌아가게 된다.

(5) Deactivated : 디스크 등의 보조기억장치에 상태 정보 등을 기록하고 Sleep상태로 들어간다

(6) Activated : 디스크 등의 보조기억장치에서 복원되어 다시 활동을 재개하게 된다.

(7) Disposed of : Aglet이 소멸되는 최종단계이며 상태정보 등이 모두 없어지게 된다.

이와 같은 Aglet의 생명주기를 도식화 하면 다음과 같다.



[그림 1] Aglet의 생명주기(Life Cycle)

[Fig. 1] Life Cycle of Aglet

3. Aglet을 이용한 웹 기반 병렬 컴퓨팅 환경의 설계

3.1 웹 기반 병렬컴퓨팅

웹은 클라이언트/서버모델로 해석되어 질 수 있으며 이를 기반으로 한 병렬 컴퓨팅 환경 또한 기본적으로 클라이언트와 서버모델로 설계 구현될 수 있다. 클라이언트는 추가적인 계산자원을 필요로 하는 사용자 혹은 계산으로 생각되어질 수 있으며 서버는 이러한 클라이언트에게 컴퓨터의 자원을 제공하여 줄 수 있는 웹상의 연결되어 있는 호스트들이 될 수 있다. 또한 이러한 클라이언트와 서버사이에서 수요와 공급을 중재하여 줄 수 있는 일종의 중재자(Broker) 시스템 혹은 메타서버(Metaserver)가 필요로 하게 된다. 성공적인 웹 기반 병렬 컴퓨팅 환경을 구축하기 위한 조건과 이를 지원할 수 있는 기술들은 다음과 같다.

• *클라이언트 측면*

웹이 오늘날과 같이 폭발적인 성장을 하게 된 배경에는 어디서나 쉽게 접근할 수 있는 접근의 용이성과 브라우저라는 단일 인터페이스를 통한 사용의 편리성 때문이다. 그러나 웹의 기본이 되는 기반기술은 단순한 정보의 전달이 이루어지는 분산 파일 시스템에 근간을 두고 있으며 진정한 병렬/분산 시스템이 되기 위하여는 병렬/분산 컴퓨팅 시스템으로 확장 발전되어야 할 필요성이 존재하게 된다. 따라서 일반 사용자에게는 웹의 장점을 모두 유지하며 웹을 컴퓨팅 자원이 연결되어 있는 광역 컴퓨팅을 위한 기반구조로 확장 발전시키기 위하여는 웹 기반 프로그래밍 환경지원 시스템이 필요하다.

• *서버 측면*

서버는 허가 받은 클라이언트에게 필요한 컴퓨팅 자원을 제공하여 주는 시스템으로서 웹 기반 컴퓨팅 환경의 안정적인 운영을 위해서는 보안과 결함포용 기술이 필요하다. 여기서의 보안이란 병렬프로그램이 컴퓨팅 노드들을 이용하여 실행하는 중에 컴퓨팅 노드에 있는 자료에 허가 받지 않고 접근하거나, 병렬 프로그램의 내용이나 수행결과가 병렬 컴퓨팅에 참여하지 않는 외부 노드에 의해 접근하는 것을 방지하는 것이다. 결함포용은 병렬컴퓨팅에 참여하는 컴퓨팅 노드나 서버에 결함이 발생하더라도 수행중인 병렬프로그램이 완료될 수 있도록 하기 위해 필요하게 된다. 결함은 크게 두 가지로 나누어 질 수 있는데 하나는 서버나 컴퓨팅 노드의 수행을 중단시키는 결함이고 또 다른 하나는 컴퓨팅 노드가 여러 가지 이유로 인하여 수행을 지연시켜 결과를 늦게 보내거나 잘못된 결과를 보내는 경우이다

• *중재자 시스템 측면*

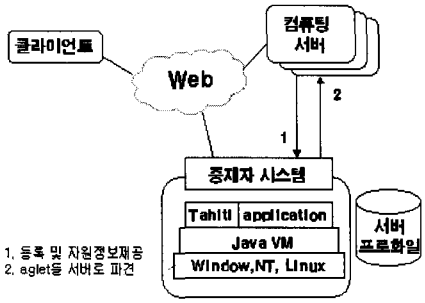
성공적인 웹 기반 병렬 컴퓨팅환경을 위하여는 효율적인 중재자 시스템의 설계가 필수적이다. 중재자 시스템은 병렬 계산에 참여하는 웹상의 컴퓨팅 노드에 대한 등록, 실행 프로파일, 처리능력에 대한 정보 등을 유지관리하며 이를 기반으로 병렬프로그램이 컴퓨팅 노드들에서 효율적으로 실행되기 위하여는 등록된 각각의 컴퓨터의 처리능력에 맞게 작업이 균등하게 분할되고 분할된 작업이 효율적으로 할당되게끔 하는 역할을 한다. 이러한 작업의 분할은 분할 입자크기가 작을 경우에는 병렬성을 충분히 이용할 수 있으므로 실행시간의 향상을 가져올 수 있으나 부수적으로 오버헤드가 커짐으로써 지연될 수 있다. 반면에 분할의 입자크기가 클 경우에는 내재되어있는 병렬성의 이용을 손실할 수 있으므로 실행효율이 떨어질 수도 있다. 최적의 작업분할과 작업할당문제는 NP-문제로 알려져 있으며 따라서 이는 실행 프로그램에 대한 정보와 참여한 컴퓨팅 노드들에 대한 정보를 이용하여 작업의 분할 및 스케줄링을 하여야 함을 의미하며 이를 위한 다양한 휴리스틱 스케줄링 알고리즘이 제공되어야 한다. 중재자 시스템은 서버의 신뢰성을 높이기 위하여 수행이 중단되는 결함 뿐만 아니라 잘못된 결과를 보내주는 결함에 대해서도 대비가 되어 있어야 한다.

중재자 시스템을 개발하는데 있어서 필요한 주요 기술들은 다음과 같다.

먼저 중재자 시스템을 구현하는데 있어서 등록된 컴퓨팅 노드들에 대한 정보를 유지관리하기 위하여는 LDAP(Lightweight Directory Access Protocol)에 기반한 API의 이용과 이를 이용한 구현기술이 필요하다. 또한 중재자 시스템과 컴퓨팅 서버들간의 효과적인 통신과 병렬계산을 위하여 효율적인 통신기술과 분산 객체기술이 필요하게 된다. 병렬프로그램이 컴퓨팅 노드들을 이용하여 실행하는 중에 컴퓨팅 노드에 있는 자료에 허가 받지 않고 접근하거나, 병렬프로그램의 내용이나 수행결과가 병렬 컴퓨팅에 참여하지 않는 외부 노드에 의해 접근하는 것을 방지하기 위하여 다양한 보안기술이 적용되어야 한다. 또한 네트워상의 결함포용기술이 필요하게 되는데 이는 병렬컴퓨팅에 참여하는 컴퓨팅 노드나 서버에 결함이 발생하더라도 수행중인 병렬컴퓨팅이 완료될 수 있도록 하기 위해 필요한 기술이다. 여기서 결함은 크게 두 가지로 나누어 볼 수 있다. 하나는 서버나 컴퓨팅 노드의 수행을 중단시키는 결함이고 또 다른 하나는 컴퓨팅 노드가 수행을 지연시켜 결과를 늦게 보내거나 잘못된 결과를 보내는 경우이다. 중재자 시스템이 서버의 신뢰성을 높이기 위하여 수행이 중단되는 결함 뿐만 아니라 잘못된 결과를 보내주는 결함에 대해서도 대비가 되어 있어야 한다. 이를 위하여 소프트웨어기반 결함고립기술이나 결함포용기술이 필요하게 된다.

3.2 Aglet을 이용한 웹 기반 병렬 컴퓨팅 환경의 설계

Aglet을 이용한 웹 기반 병렬 컴퓨팅 환경의 개념적 구성도는 다음과 같다.



[그림 2] Aglet을 이용한 웹 기반 병렬 컴퓨팅 환경의 개념적 구성도

[Fig. 2] Conceptual Configuration of Web-based Parallel Computing Environment using Aglet.

이를 구현하기 위하여 필요한 주요 객체들과 그에 따른 설명은 다음과 같다.

class ServerRecord /* 등록된 서버들의 정보를 포함하는 구조체

String ServerID
String name
String URL
int CPUPower
int Storage

class ServerProperty /* 등록된 서버목록을 저장하고, 로드할 수 있도록 관리하는 클래스

Method
public String getInfo()
public boolean setInfo()
public String[ ] getServer()

class MobileCoordinator /* 에이전트들을 조정하며 서버들에게 파견될 작업을 분할 한다.

class TaskHandler /*현재의 상태를 인식하며 Runnable interface로 구현된다.

class BrokerServer /* 이용가능한 Tihiti 서버를 관리하고 필요한 정보를 유지 관리한다.

class Hcinterface /* 사용자 인터페이스를 구현한 클래스

class WorkingAgglet /* 실제로 작업이 수행될 에이전트이며 기본 계산단위를 가지고 가서 작업을 수행한다.

4. 시뮬레이션

본 장에서는 aglet을 이용한 병렬 컴퓨팅 환경에서 간단한 시뮬레이션과 시뮬레이션 결과에 따른 분석이 행하여진다. 시뮬레이션은 웹상에 연결되어있는 4대의 컴퓨터에서 이루어졌으며 이들 컴퓨터의 상세는 <표 1>과 같다.

<표 1> 시뮬레이션에 사용된 컴퓨터 상세

<Table 1> Computer Spec. on Simulation

Computer	CPU	Main memory	HDD
I	Celeron 700	256M	20G
II	Pentium III 800	512M	30G
III	Pentium III 800	256M	30G
IV	Pentium III 800	256M	30G

시뮬레이션에 사용된 벤치마크 프로그램은 Lawrence Livermore Loop중에서 비교적 간단한 형태의 LLL1프로그램을 채택하여 루프의 반복회수를 400에서 11000으로 변형하였다.

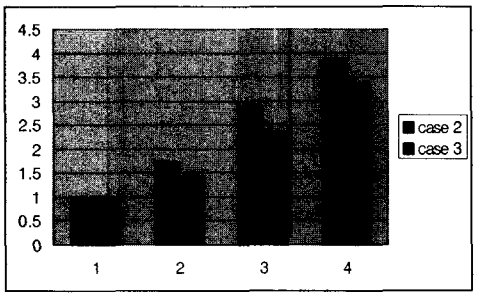
실험결과는 세가지 경우에 대하여 각각 이루어졌으며 각 경우에 대한 설명은 다음과 같다.

case 1)
병렬컴퓨팅이 아닌 한 컴퓨터에서 변형된 LLL1을 실행하는 경우
case 2)
변형된 LLL1을 n개의 applet으로 분할하여 n개의 서버에서 웹을 이용한 병렬컴퓨팅을 할 경우.

case 3)
변형된 LLL1을 n개의 aglet으로 분할하여 n개의 서버에서 웹을 이용한 병렬컴퓨팅을 할 경우.

[그림 3] 은 case 1에 대한 상대적인 case 2와 case 3의 speed-up 을 보여주고 있다.

[그림 3]에서 보듯이 계산에 참여하는 컴퓨터 노드들이 증가함에 따라 speed-up factor 또한 증가를 하고 있으며, 계산에 참여하는 노드들이 증가할수록 speed-up factor 또한 더 증가하였다. 이러한 이유는 분산에 따른 여러 오버헤드들이 계산에 따른 컴퓨터의 노드들이 증가하면서 상대적인 비율이 감소하고 또한 병렬계산을 통한 장점이 상대적으로 증가함을 의미하고 있다. Case 2의 applet와 비교할 때 case 3의 aglet의 speed-up은 상대적으로 작으나 aglet이 더 많은 응용이 가능하며, 더 많은 잠재적 가능성을 가지고 있다. 예를 들면 상호 신뢰도가 어느 정도 보장된 인트라넷환경에서 aglet은 데이터베이스의 갱신(update)연산 등이 가능하며, 하나의 aglet이 복제되어 n개의 서버들 사이로 이동이 가능하나, applet은 의미론상 그와 같은 일들이 불가능하거나 제약조건을 수반하게 된다.



[그림 3] 컴퓨팅 노드 증가에 따른 speed-up factor
[Fig. 3] Speed-up Factor as the computing nodes increase

5. 결론

Aglet기술은 한 컴퓨터로부터 다른 컴퓨터로 프로그램 코드와 상태나 데이터를 옮길 수 있는 경량 자바 객체(lightweight Java object)인 동시에 서로 합의한 컴퓨터들 사이에는 자바 applet보다 더 광범위한 작업을 할 수 있다. 따라서 주어진 작업을 병렬적으로 처리하기 위하여 지역적으로 분산된 웹 상의 노드들간에 상호작용 시에 이동형 에이전트인 Aglet을 이용하면 많은 장점을 얻을 수 있다. 본 논문에서는 Aglet의 개념과 구조, 이를 이용하여 웹 기반 병렬컴퓨팅환경에 대한 개념적 설계와 문제점이 논의되고 간단한 시뮬레이션과 이에 따른 분석이 행하여졌다. 향후 연구로서는 웹 상에서의 많은 컴퓨팅 노드들에서 aglet을 실행하게 될 때 다양한 실행모델과 실행 시나리오의 개발이 필요하다. 또한 이에 따른 다양한 벤치마크 프로그램들의 성능분석이 향후 연구로서 남아 있다.

※ 참고문헌

[1] P.H.Jr. Enslow, "Multiprocessor Organization - A Survey", ACM Computing Surveys Vol.9 No.1 1977 pp. 103-125.

[2] http://www.trl.ibm.co.jp/aglets/index.html

[3] http://www.javaworld.com/javaworld/jw-05-1997/jw-05-hood.html

[4] http://luckyspc.lboro.ac.uk/Aglets/

[5] Java Aglet API WhitepaperAglets

[6] J. M. Andreoli, F. Pacull and R. Pareshi, ''XPect: A framework for electronic commerce,'' IEEE Internet Computing, vol. 1, no. 4 (July-August 1997), pp.40-48.

[7] AuctionBot URL: http://auction.eecs.umich.edu.

[8] Firefly URL: http://www.firefly.com.

[9] Albert D. Alexandrov, Maxilian Ibel, Klaus E.Schauser "Superweb: Towards a Global Web based Parallel Computing Infrastructue"

[10] R. Guttman and P. Maes, ''Agent-mediated integrative negotiation for retail electronic commerce,'' Mediated Electronic Trading Workshop, Minneapolis, MN (May 1998),

[11] D. B. Lange, M. Oshima, G. Karjoth and K. Kosaka, ''Aglets: Programming mobile agents in Java,'' Proceedings of the International Conference on Worldwide Computing and Its Applications, Tsukuba, Japan (March 1997), Lecture Notes in Computer Science 1274, Springer-Verlag, Berlin, Germany, pp. 253-266.

[12] Karjoth G, Lange, et al. " A Security Mode l for Aglets, IEEE Internet Computing1, 4, 1997 pp 68-77

[13] White, J. Mobile Agent, In Software Agents, Bradshaw, MIT Press 1997

[14] P.Evripidou, G.Samaras, E. Pitoura and C. Panayiotou, " PacMan : Parallel Computing Using Java Mobile Agents" , 13th ACM International Conference on Supercomputing(ICS), Workshop on "Java for High Performance Computing", Rhodes, Greece, June 1999

김 윤 호

1985년 서울대학교
계산통계학과 졸업(이학사)
1987년 서울대학교
계산통계학과 대학원
전산과학전공(이학석사)
1996년 서울대학교
계산통계학과 대학원
전산과학전공(이학박사)
현재 상명대학교
소프트웨어학부 조교수