다중 지역 이동 에이전트 컴퓨팅 환경에서의 공간 기반 복제 기법을 위한 이동 에이전트 위치 관리 프로토콜*

윤준원[O] 최성진[†] 양권우[‡] 손진곤 황종선[†]
고려대학교 컴퓨터학과 분산시스템 연구실, 공주교육대학 컴퓨터교육과[‡], 한국방송통신대학 컴퓨터과학과
{zebra[O], lotieye[†],hwang[†]}@disys.korea.ac.kr, kwyang@pro.gjue.ac.kr[‡], jgshon@mail.knou.ac.kr[‡]

Mobile Agents Location Management Scheme for Spatial Replication Based Approach in Multi-Region Mobile Agents Computing Environment

JunWeon Yoon[0], SungJin Choi, ChongSun Hwang
Distributed System Lab. Dept. of Computer Science & Engineering, Korea University
KweonWooYang
Dept. of Computer Science & Education,Gongju National University of Education
JinGon Shon
Dept. of Computer Science, Korea National Open University

요 약

다중 지역으로 구성된 이동 에이전트 컴퓨팅 환경에서 결함 포용을 위한 대표적인 기법인 공간 기반 복제 기법(SRBA : Spatial-Replication-Based approach)은 이동 에이전트가 수행 중 중간 연산에 대한 빠른 결과 산출이라는 장점을 가지는 대중화된 결함 포용기법이다. 그러나 공간 기반 복제 기법을 실제 이동 에이전트 기반 컴퓨팅 환경에 적용하는데 있어서 해결되어야 하는 중요한 선결 과제 중의 하나는 단계별로 복제된 이동 에이전트들에 대한 추가 위치 관리 비용을 감소시키는 것이다. 본 논문에서는 공간 기반 복제 기법에서 발생하는 불필요한 위치 관리 오버헤드를 줄이기 위해 이동 에이전트를 위한 새로운 위치 관리 기법인 SRLM(Spatial Replication Location Management) 프로토콜을 제안한다. 제안된 SRLM 프로토콜은 단계군 구성시 복제된 이동 에이전트를 위한 등록 비용 및 대표 작업자에게 메시지 전달을 보장하며 메시지 전달 비용을 감소시킨다. 또한, 제안된 프로토콜은 단계 군 내에서의 대표 작업자의 결함 발생시 새로운 대표 작업자의의 선출로 인한 위치 관리 문제를 해결한다.

1. 서 론

이동 에이전트는 사용자를 대신하여 호스트와 호스트를 자율적으로 이동하면서 연산을 수행하는 컴퓨터 프로그램이다[1][2]. 여러 지역으로 구성된 이동 에이전트 컴퓨팅 환경에서 이동 에이전트 수행 중 에이전트를 관리하기 위한 위치 관리 기법은 이동 에이전트 시스템을 개발하는데 있어서 중요한 고려사항이다[4].

이동 에이전트 컴퓨팅 환경에서 이동 에이전트 수행 중 통신고장이나 호스트의 고장으로 인해 연산이 부분적 또는 전체적으로 손실되어 연산 수행이 정지 되는 현상이 발생한다. 이동 에이전트에 대한 결함 포용적 연산 수행을 위해, 기존 연구에서는 이동 에이전트와 이동 에이전트의 실행장소를 복제하여 결함을 포용 하는 기법이 사용된다. 복제를 기반으로 한 결함 포용 기법으로는 크게 시간적 기반 복제 기법(TRBA : Temporal-Replication-Based approach)과 공간적 기반 복제 기법(SRBA : Spatial-Replication-Based approach)으로 구분된다[7]. TRBA는 이전 단계에서 현재 진행 중인 단계의 에이전트를 저장하여 현재 단계가 고장이 나더라도 이전 단계에서 미리 저장되어 있는 에이전트를 다시 보내어 연산을 수행하는 방법이다.[7] SRBA는 한 단계에서 여러 실행장소를 두어 결함 포용을 해결하는 방법으로 현재 작업을 진행 중인 실행장소가 고장이 나더라도 단계를 이루는 다른 실행장소들이 새로 작업자로 선출되어 작업을 진행하는 방법이다[7]. 또한 SRBA 기법은 이동 에이전트가 수행 중, 중간 연산에 때한 빠른 결과를 제공하기 때문에 대부분의 이동 에이전트 컴퓨팅 환경에서 사용되고 있다.

기존 이동 에이전트를 위한 위치 관리 기법은 TRBA기법에 적용 가능하다. 그러나 실행장소들을 공간적으로 구성하여 하나의 단계군으로 구성한 다음 결함을 포용하는 SRBA 기법에서는 적용 가능하지 않다. 특히, 기존의 위치 관리 기법을 SRBA에 적용했을 경우 높은 위치 관리 및 메시지 비용이 발생하며, 단계군 내에 작업자가 고장으로 새로운 작업자를 선출한 경우 메시지 전달을 보장하지 못하는 문제점이 발생한다.

이에 본 논문에서는 결함 포용을 위해 제안되었던 SRBA에 적용되는 SRLM(Spatial Replication Location Management) 프로토콜을 제안한다.

2. 관련 연구

이동 에이전트 컴퓨팅 환경에서 이동 에이전트에 대한 위치 관리 기법은 크게 홈 프락시(Home-Proxy), 추적 프락시(Follower-Proxy), 브로드캐스트(Broadcast)기법으로 분류된다[5]. 홈 프락시 기법[5]에서는 위치 정보 저장소를 이동 에이전트가 생성된 홈 호스트에 위치시키고 이동 에이전트 이주 시마다 그 위치 정보를 홈 호스트의 위치 정보 저장소에 저장한다. 위치 탐색 시 이 정보를 이용하여 이동 에이전트를 찾아 메시지를 전달하는 기법이다. 추적 프락시 기법[5]은 위치 정보를 방문한 호스트들에 저장한 다음 경로 프락시(Path proxies)를 따라 이동 에이전트를 찾거나, 메시지를 경로 프락시를 따라 계속해서 포워딩(Forwarding)하는 기법이다. 브로드캐스트 기법[5]은 정해진 지역이나 이동계획(Itinerary)의 모든 호스트들에게 메시지를 보내어 위치를 찾거나 메시지를 전달하는 기법이다. 위 세 가지의 기법은 단일 지역으로 구성된 이동 에이전트 컴퓨팅 환경에 기반하고 있다. 다중 지역으로 구성된 이동 에이전트에 대한 위치 관리 기법으로는 SPC[3] 프로토콜이 있다. 이 기법에서 이동 에이전트는 한 호스트에서 생성되고, 그 지역 서버는 지역내의 이동 에이전트의 정보를 가지고 있게 된다. 이동 에이전트는 여러 지역을 이주하게 되므로 많은 갱신 메시지 비용이 발생하게 되며, 위치 관리시 갱신 메시지의 분실로 인해 호스트에 저장된 이동 에이전트의 정보를 따라가는 높은 탐색 비용을 초래할 수 있다. 결함포용을 위한 SPC 프로토콜은 TRBA에 적용 가능하나, SRBA에 적용시 단계군 내의 모든 이동 에이전트들에 대한 위치 정보 메시지를 등록해야하는 문제점이 발생한다.

3. 이동 에이전트 컴퓨팅 환경

본 논문에서 가정하는 이동 에이전트 컴퓨팅 환경은 이동 에이전트 a_i, 실행장소 P_i, 단계군 S_i로 구성된다. 이동 에이전트 a_i는 각 호스트의 실행 결과에 따라 다음 호스트를 동적으로 선택하여 이동하거나, 이동계획(itinerary)에 따라 호스트와 호스트사이를 옮겨 다니며 작업을 수행한다. 이때 호스트에서는 이동 에이전트가 수행할 수 있는 실행장소 P_i를 제공한다[2]. 실행장소는 이동 에이전트에게 특정 서비스를 제공한다. 이동 에이전트 시스템들 중에서 같은 권한을 가진 이동 에이전트 시스템들의 집합을 지역 R_i(Region)이라고 한다[2]. 이동 에이전트의 결함 포용을 위해 실행장소들을 모아 단계군 $\{S_0, S_1, \ldots, S_{i-1}\}$을 형성하며, 출발지(Agent Source:S_0)와 도착지(Agent Destination:S_{i-1})까지 동적으로 이주하며 작업을 수행하게 된다. 전 단계에서 실행된 이동 에이전트는 결함 포용을 위해 다음 단계군 내의 모든 실행장소에 복제되고, 단지 하나의 실행장소에서만 에이전트가 수행 되어진다. 만약, 에이전트 수행 중 고장이 생기면 단계군내의 다른 실행장소(place)들은 고장을 탐지한 후 새로운 작업자를 선출하여 작업을 진행하게 된다[7].

단계군을 구성할 때 이동 에이전트의 이동 계획과 단계군을 구성하는 실행장소의 수에 따라 다른 단계군이 구성된다[8]. 이때 단계군을 이루는 실행장소는 서로 같은 지역(region)에 있을 수도 있고 서로 다른 지역에 있을 수도 있다. 본 논문에서는 이동 에이전트가 여러 지역으로 구성된 다중 지역 이동 에이전트 컴퓨팅 환경을 이동하면서 작업을 수행

* 본 연구는 한국과학재단 목적기초연구(R01-2002-000-00235-0)지원으로 수행되었음.

하는 것으로 가정한다. 그림 1은 본 논문에서 가정하는 이동 에이전트 컴퓨팅 환경을 나타내고 있다.

본 논문에서는 호스트 또는 실행장소의 고장을 간주하며, 결함 발생 시 다른 구성요소에 영향을 끼치지 않고 자신의 내부 수행 상태를 상실하는 고장 정지(failure-stop)모델을 전제하기로 한다.

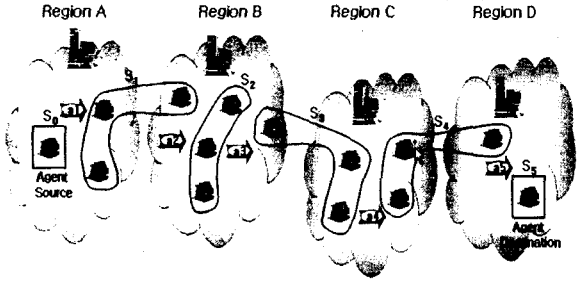

그림 1. 이동 에이전트 컴퓨팅 환경

4. 제안 기법

다중 지역 이동 에이전트 컴퓨팅 환경에서 단계군으로 구성된 실행장소들은 위치 관리시 에이전트의 실행을 시작한 실행장소만을 등록함으로써 위치 등록 비용 및 메시지 전달 비용을 감소시킨다.

SRLM 프로토콜의 위치 관리는 다음과 같이 크게 에이전트 네이밍&생성, 이주, 결함 포용적 위치 등록, 메시지 전달 과정으로 구분된다.

- **생성 & 네이밍 과정**
  이동 에이전트가 생성되면 위치 관리를 위해 그 정보를 등록하게 되는데, 우선 생성된 지역 서버(RS_{Home} : Home Region Server)에 이동 에이전트의 식별자와 생성된 호스트의 주소 ({AgentID, HostLocation})를 등록하게 된다.
- **이주 과정**
  다중 지역에서 이주 과정은 크게 같은 지역으로 이주하는 지역내 이주(Intra Migration)와 다른 지역으로 이주하는 지역외 이주(Inter Migration)로 나뉘어 진다. 지역내 이주시 갱신정보는 그 지역의 지역서버와 이전 호스트에게 알리게 되고, 다른 지역으로 이주하는 경우, 이동 에이전트는 생성된 지역서버(RS_{Home}), 이주 전 지역서버(RS_{before}), 이주 후 현재 지역서버($RS_{current}$) 그리고 이전 실행장소에 갱신 메시지를 전달하게 된다. 표1과 그림2는 이동 에이전트의 메시지 갱신 내용과 위치 등록 과정을 나타내고 있다.

지역내 이주	$RS_{current}$	{AgentID, Location of P_i^{prim}}
지역외 이주	RS_{Home}	{AgentID, Location of $RS_{current}$}
	$RS_{current}$	{AgentID, Location of P_i^{prim}}
	RS_{before}	{AgentID, Location of $RS_{current}$}

표 1. 이동 에이전트 이주시 메시지 갱신 정보

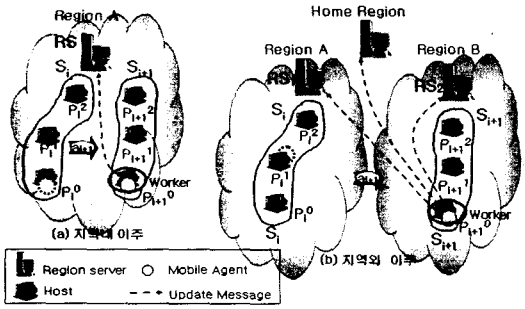

그림 2 이동 에이전트 이주 과정

- **결함 포용적 위치 등록 과정**
  SRLM 프로토콜에서 에이전트 수행 중 고장이 생기면 단계군내의 다른 실행장소(place)들은 고장을 탐지한 후 새로운 대표 작업자(worker)를 선출하여 작업을 진행하게 되고[7], 이동 에이전트의 위치 정보는 단계군 내에서 새로 선출된 실행장소의 위치에 따라 표1과 같이 지역내 이주 또는 지역외 이주 방법에 의해 지역서버에 위치 메시지를 갱신하게 된다.

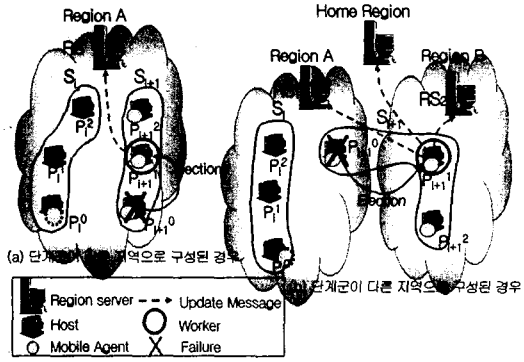

그림 3. 결함 포용적 위치등록 과정

그림3과 같이 SRLM 프로토콜에서 고장이 발생하여 새로운 대표 작업자의 위치 정보를 갱신할 때, 이전 단계군의 실행장소가 위치한 지역과 상관없이 이전 작업자의 위치 정보를 따르게 된다. 즉, 단계군에서 고장이 발생한 후, 새로 선출된 작업자의 위치가 같은 지역에 있다면 이전 단계군의 실행장소와 상관없이 새로운 작업자에 대한 위치 갱신 메시지는 지역내 이주 방법을 따르게 된다. 그림 4는 이동 에이전트의 이주과정에 대한 알고리즘이다.

```
// -- 이주 과정 --
SendAgent(a_i);                    // 에이전트 이주
UpdateMsg(RS_current, Loc of P_i^prim);  // 이주 지역의 RS 등록(지역내이주)
if compare(Loc of P_i-1^prim, Loc of P_i^prim) = 0 then  // 지역외 이주시 추가 등록
      UpdateMsg(RS_before, Loc of RS_current);     // 이주전 지역의 RS 등록
      UpdateMsg(RS_Home, Loc of RS_current);       // 홈 RS 등록
fi;
if(P_i^prim=P_destination^prim) then  // 에이전트 연산 종료
      Termination();
fi;

// -- 결함 포용적 위치 등록 과정 --
if P_i^prim = failure then  // 단계군(stage M_i)내에 작업자가 고장난 경우
      Election();           // 새로운 작업자 선출
      UpdateMsg(RS_current, Loc of P_i^NewWorker);//새 작업자 현재 위치 등록
         if compare(Loc of P_i^prim, Loc of P_i^NewWorker) = 0 then
         //이전 작업자와 새 작업자가 서로 다른 지역에 있는 경우 추가 등록
            UpdateMsg(RS_before, Loc of RS_current);//이주전 위치갱신메시지 전달
            UpdateMsg(RS_Home, Loc of RS_current);//홈RS 위치갱신메시지 전달
         fi;
      prim = NewWorker;
fi;
```

그림 4. 이주 및 결함 포용적 위치 등록 과정

- **메시지 전달 과정**
  그림 5는 메시지 전달과정에 대한 알고리즘이다. 본 논문의 기법은 작업자가 고장이 난 경우에도 메시지 전달을 보장한다.

① 메시지를 전달하고자 하는 통신자는 이동 에이전트가 성성된 지역서버(RS_{Home})에 질의를 한다.
② RS_{Home}는 이동 에이전트가 있는 지역서버를 찾는다. 이동 에이전트가 RS_{Home}에 있는 경우, 해당 호스트로 메시지를 전달하게 되고, 그렇지 않은 경우 이동 에이전트가 있는 지역서버로 메시지를 보낸다.
③ 지역서버에 있는 이동 에이전트의 위치 정보를 이용하여 이동 에이전트가 있는 호스트로 메시지를 전달한다.
④ 이동 에이전트가 고장이 난 경우, 지역서버는 고장난 이동 에이전트에게 보냈던 전달되지 않은 메시지를 버퍼에 저장한다.
⑤ 새로운 작업자가 선출되면, 그 실행장소는 지역서버에 위치 갱신 메시지를 등록하고, 현재 지역서버 버퍼에 저장되어 있는 메시지를 가져온다.
⑥ 실행장소에게 메시지가 전달된 후, 단계군내의 작업이 끝나면 버퍼에 있는 메시지를 삭제한다.

```
CN.SendMsg(Msg, α, RS_Home); // 통신자는 홈 지역서버에 메시지 전달
RSAddr = FindAddr(α);        // 에이전트가 위치한 지역서버 주소를 가져온다.
if compare(Addr of RS_Home , RSAddr) = 1 //홈 지역서버(RS_Home)에 있는 경우
    SendMsg(Msg, α, HostAddr);          //해당 호스트에 메시지 전달
    if HostAddr = failure then          //해당 호스트가 고장난 경우
        SendMsgFailure(α);              //메시지가 전달되지 않은 경우
    fi;
else                                    //에이전트가 다른 지역으로 이주 한 경우
    SendMsg(Msg, α, RSAddr);            //해당 지역서버로 메시지 전달
    SendMsg(Msg, α, Hostaddr);          //해당 호스트에 메시지 전달
    if HostAddr = failure then          //해당 호스트가 고장난 경우
        SendMsgFailure(α);              //메시지가 전달되지 않은 경우
    fi;
fi;
Fucnction SendMsgFailure(α)             // 이동 에이전트가 고장난 경우
    Buf.StoreMsg(Msg, α, RSAddr) // 메시지를 지역서버의 버퍼에 저장
    RS.receive(NewHostAddr);     // RS에 새 작업자의 주소를 받는다.
    if compare(Addr of NewHost, RSAddr); // 새 작업자가 다른 지역
        SendMsg(Msg, α, RSAddr);         // 먼저 해당 RS 메시지 전달
    fi;
    SendMsg(Msg, α, NewHostAddr);        // 새 작업자에게 메시지 전달
    StageCommit(S);
End Function;
```

그림 5. 이동 에이전트 이동 에이전트 메시지 전달 과정

4. 분석 및 토론

이주 과정에 따른 메시지 갱신 비용과 전달 비용을 살펴보자. 기존 다중 지역을 다루었던 대표적인 기법인 SPC 프로토콜에 SRBA를 적용했을 경우와 본 논문의 제안 기법인 SRLM 프로토콜을 비교하였다. SPC 프로토콜에서 SRBA가 적용된 경우에는 이동 에이전트가 이주한 단계군내의 모든 실행장소에 대해 위치 갱신 메시지를 전달하게 된다. SRLM 프로토콜에서는 단계군 내의 대표 작업자에 대해서만 위치 갱신 메시지를 전달하게 된다.

이주 과정에 대한 위치 갱신 메시지 비용을 계산해 보면 SPC 프로토콜에서 SRBA가 적용된 경우에 메시지 비용은 이전 단계군의 작업자와 다음 단계군의 실행장소들이 같은 지역 또는 다른 지역에 존재하는지에 따라 달라진다. SRLM 프로토콜에서 메시지 비용은 이전 작업자와 새로운 작업자가 같은 지역으로 구성된 단계군인지에 따라 결정된다. 단계군의 개수가 m개, 단계군의 크기는 n개(일반적으로 3, 5, 7개로 구성)가 있고, 이동 에이전트가 단계군을 이동할 때, 지역내 이주 수를 a, 지역외 이주 수를 b라 하자. 각 단계군 내 실행장소의 고장 확률을 $k(0<k\leq n)$라 할 때, SPC 프로토콜의 위치 메시지 비용과 SRLM프로토콜에서 이주 과정 및 결합 포용적인 과정을 포함한 위치 갱신 메시지 비용은 다음과 같다.

본 모의 실험에서는 각 단계군의 구성이 같은 지역에 구성되어 있는 경우를 가정하고 메시지 비용을 측정하였다. 단계군이 다른 지역으로 구성된 경우 SPC 프로토콜 보다 SRLM 프로토콜 메시지 비용이 훨씬 줄어드는 것은 자명하기 때문이다.

구분	SPC 프로토콜	SRLM 프로토콜
위치 갱신 메시지 비용	n(2a+4b)	a+3b+k

표 2. SPC와 SRLM 프로토콜에서 위치 갱신 메시지 비용

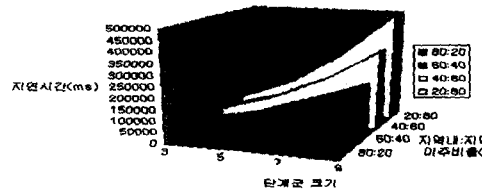

그림 6. SPC에서 SRBA 적용시 위치 갱신 메시지 비용

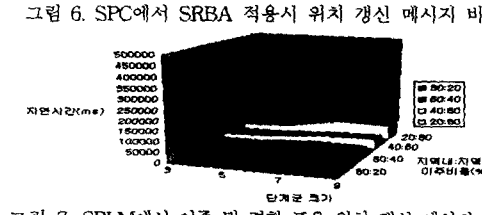

그림 7. SRLM에서 이주 및 결합 포용 위치 갱신 메시지 비용

그림 6, 그림 7은 SPC에서 SRBA를 적용한 경우와 SRLM에서 이주 및 결합 포용 위치 등록에 대한 메시지 갱신 비용을 비교하였다. SPC프로토콜에서는 단계군내의 모든 실행장소에 대해 위치 갱신 메시지를 보내게 되므로 SRLM 프로토콜보다 높은 갱신 메시지 비용이 발생하게 된다.

결합 포용적 위치 등록 과정에 대한 메시지 비용을 계산해 보자. 이전 기법에서는 작업자의 고장이 난 경우 새로운 작업자에 대한 위치 관리 기법이 없었다. 본 SRLM 프로토콜에서는 대표 작업자가 고장이 난 경우 새로운 작업자에 대한 위치 정보를 등록함으로써 메시지 전달을 보장한다.

구 분	메시지 전달 비용
SPC 프로토콜 SRBA 적용시	3n
SRLM 프로토콜	3k

표 3. SPC와 SRLM 메시지 전달 비용

표 3의 메시지 전달 비용을 살펴보면 SPC 프로토콜은 단계군내의 모든 실행장소(n개)에 메시지를 전달하게 되고, SRLM 프로토콜에서는 단계군 내에 실행장소가 고장난 경우에 다시 메시지를 전달하게 된다. 지역 서버는 메시지가 전달된 후 단계군의 작업이 끝날 때까지 메시지를 버퍼에 저장한다. 새로운 작업자는 지역서버에 위치 갱신 메시지를 등록하고, 현재 지역서버 버퍼에 저장되어 있는 메시지를 가져온다.

기존 위치 관리 기법에서는 메시지 전달시 대표 작업자가 고장이 난 경우 새로운 작업자에 대한 위치 관리가 없어 메시지 전달을 보장하지 못한다.

본 논문의 기법에서는 대표 작업자가 고장난 경우 작업자를 선출한 후 새로운 실행장소의 위치 정보를 갱신함으로써 이전 위치 관리 기법에서 다루지 못한 고장난 호스트들에 대한 메시지 전달을 보장한다.

6. 결론 및 향후 연구

본 논문은 다중 지역 이동 에이전트 컴퓨팅 환경에서 작업자가 고장난 경우, 메시지 전달을 보장하기 위한 위치 관리 기법을 제안하였다. 본 기법은 다중 지역에서 실행장소를 단계군으로 구성하여 고장이 발생시 새로운 작업자를 선출하고, 그 실행장소만을 등록함으로써 위치 등록 비용 및 메시지 전달 비용을 감소시킨다. 그러나, 단계군 구성기법은 실행장소(place)를 공간적으로 복제하기 때문에 그로 인한 각 실행장소로 에이전트를 전송하는 부하와 단계군에서 행할 작업들의 감시(monitoring), 투표(voting), 종료(termination)에 대한 부하들이 생기게 된다.

향후 연구로써는 단계군 내에 실행장소들을 재구성하여 이동 에이전트의 컴퓨팅 환경에 부하를 감소시킬 수 있는 위치관리 기법이 요구된다.

참 고 문 헌

[1] Neeran Karnik and Anand Tripathi, "Design issues in Mobile Agent Programing Systems", IEEE Concurrency, pp 52-61, July-Sep 1998.

[2] Dejan Milojicic, Markus Breugst, Ingo Busse, John Campbell, Stefan Covaci, Barry Friedman, Kazuya Kosaka, Danny Lange, Kouichi Ono, Mitsuru Oshima, Cynthia Tham, Sankar Virdhagriswaran1 and Jim White, "MASIF : The OMG Mobile Agent System Interoperability Facility", In Proceedings of the Second International Workshop on Mobile Agents (MA'98), LNCS 1477, pp 14-15. springer Verlag, September 1998.

[3] Antonella Di Stefano, A. Lo Bello, L; Santoro. C.; "Naming and Locating Mobile Agents in an Internet Environment", Enterprise Distributed Object Computing Conference, 1999. EDOC '99. Proceedings. Third International , 27-30, Page(s): 153 -161 9. 1999

[4] Antonella Di Stefano and Corrado Santoro, "Locating Mobile Agents in a Wide Distributed Environment", IEEE Transactions on Parallel and Distributed Systems, Vol. 13, No. 8, 2002.

[5] Dwight Deugo, "Mobile Agent Messaging Models", In Proc. 5th International Symposium on Autonomous Decentralized Systems, pp. 278 -286, 2001.

[6] D.B. Lange and M. Oshima, "Programming Mobile Agents in Java - with hte Java Aglet API". Addison-Wesley, 1998.

[7] S. Peisch and A. Schiper, "Approaches to Fault-Tolerant Mobile Agent Executions", IBM Research Report 3333, 2001.

[8] Markus Straßer, Kurt Rothermel, "Reliability Concepts For Mobile Agents", International Journal of Cooperative Information Systems(IJCIS), Vol 7, No 4 . pp. 355-382, 1998.