# Web/Mobile 정보 서비스 시스템을 위한 멀티 에이전트 구현

이용준, 배석찬
군산대학교 컴퓨터정보과학과

# Implementation of Multi-Agents for Web/Mobile Information Service System

Yong Jun Lee, Seck Chan Bae
Dept. of Computer Information Science, Kunsan National University
E-mail : cyanlyz@naver.com

## 요 약

현재 사용되고 정보 서비스 기술의 형태는 다양화, 복잡화되고 있지만 대부분 웹 기반 서비스 기술을 사용하고 있다. 그러나 이러한 기술은 분산되어 있는 정보를 가공하는 과정이 복잡하고, 검색결과가 너무 많아 정보의 질이 저하된다. 또한, 한정적인 대역폭으로 인해 네트워크 상에 지연문제가 발생하여 효율적인 정보를 사용자에게 전달하지 못한다. 따라서 본 논문에서는 멀티 에이전트 기반의 Web/Mobile 환경에서의 능동적 정보 서비스를 위한 시스템을 구현한다. 제안된 멀티 에이전트 시스템에서는 기존의 웹 기반 정보 서비스가 가진 단점들을 해결하기 위해서 이동 에이전트의 작업 실행 모듈을 분리하는 CORBA 기반의 이동에이전트를 구현하여 분산된 다양한 정보를 효과적으로 수집하였고, 네이밍 에이전트의 호스트 이주를 위해 노드의 위치 투명성 및 최초 라우팅 스케줄을 지정하는 네이밍 에이전트를 구현하였다. 마지막으로 사용자에게 지속적인 정보 서비스를 제공함으로 네트워크 트래픽을 가중시키는 문제를 해결하였다.

## 1. 서 론

정보화 사회의 발전과 인터넷의 확산과 다양한 IT기술들의 변화로 사용자는 많은 양의 정보와 서비스를 이용할 수 있을 뿐만 아니라, 원하는 정보를 원하는 시간 안에 얻을 수 있게 되었다. 그러나 정보의 종류와 양이 많아짐으로 인해 원하는 정보만을 효율적으로 찾기가 점점 어려워지고 있으며 사용자들은 자신이 원하는 정보를 검색하고 서비스 받기 위해 많은 시간과 노력을 소모해야 한다. 그리고 하나의 에이전트가 해결할 수 있는 문제의 양이나 범위에는 한계가 있어 여러 개의 에이전트가 제한된 시간 내에 한정된 자원을 적절히 활용하고 배분해야 한다는 문제를 해결하기 위해 멀티 에이전트 시스템이 제안되고 있다. 본 연구에서는 Web/Mobile 환경에서의 능동적 정보 서비스를 위한 멀티 에이전트 시스템을 구현한다. 제안된 Web/Mobile 환경에서의 능동적인 정보 서비스를 위한 멀티 에이전트 시스템은 기존 웹 기반 정보 서비스가 가진 단점들을 해결하기 위해 멀티 에이전트를 이용하여 분산되어 있는 다양한 비즈니스 정보에 접근하여 정보를 수집하는 이동 에이전트와 이동 에이전트 이주 노드의 위치 투명성 및 초기 routing 스케줄을 지정하는 네이밍 에이전트, 사용자에게 능동적 정보를 제공하는 푸시 에이전트, 또한, 제공되는 시스템 자원과 에이전트들을 관리하는 시스템 모니터링 에이전트를 포함한다. 멀티 에이전트 시스템에서는 네이밍 에이전트, 이동 에이전트 및 푸시 에이전트와의 상호 협력을 통해 정보 검색 시간의 절감과 네트워크 Traffic 감소는 물론 안정적이고 정확한 정보 서비스와 사용자 주문형 정보 요구에 대한 능동적인 정보 서비스를 지원한다[1,2,6].

본 논문의 구성은 다음과 같다. 1장에서 연구의 필요성 및 목적을 정의하고, 2장에서는 관련 연구로서 서버기반 및 이동에이전트기반의 정보시스템에 대해 기술하였고, 3장에서는 Web/Mobile기반의 정보 서비스에서 멀티에이전트의 구조 및 기능에 대해

제시하고 구성에이전트에 대한 알고리즘을 제안한다. 마지막으로 4장에서는 결론 및 향후 연구 과제를 제시하였다.

## II.관련 연구

### 2.1 서버 기반 정보 서비스 시스템

서버 기반 정보 서비스 시스템은 비즈니스 Logic을 서버가 가지고 있는 시스템이다. 서버에 검색엔진, 데이터베이스 등이 위치하고 단순히 클라이언트 쪽은 브라우저 기능을 위주로 하는 경우이다. 즉, 수많은 클라이언트의 요구에 단 방향으로 응답하는 요구-응답 방식을 가지고 있다. 이러한 서버 기반 시스템의 문제점은 첫 번째, http의 빈번한 연결 설정과 해제 과정으로 많은 페이지를 대기 상태로 두므로 서버에 부담을 가중시킬 수 있다. 두 번째, 작업이 서버에게 집중되므로 클라이언트의 증가에 따른 서버의 부담이 너무 크고, 많은 수행들이 서버에서 이루어지므로 상대적으로 사용자의 요구에 대한 처리속도가 늦어질 수 있다. 세 번째, 네트워크 트래픽을 줄일 방법이 없다. 네 번째, 분산되어 있는 방대한 양의 정보를 가공 과정이 복잡하다. 즉, 다수의 문서들을 위하여 여러 번의 연결 설립, 동작, 해제의 단순 동작이 반복된다, 이러한 잦은 연결은 설립과 여러 개의 연결을 설립해야 하는 동작 구조로 잦은 링크에서 부하를 가중시키고 웹 사용자에게 만족스럽지 못한 성능을 제공한다[3,7].

### 2.2 이동 에이전트 정보 서비스 시스템

이동 에이전트 시스템은 에이전트를 인터넷상에 보내 최적의 경로를 통하여 검색코드 자체가 이동하여 검색을 수행하므로 기존의 정적인 웹 로봇에 비하여 고정된 네트워크 거리를 극복할 수 있고 단축된 원격 서버와의 네트워크 거리만큼 검색시간을 단축시킬 수 있다. 에이전트가 목적하는 노드로 이동이 이루어지고 나면, 응답 전까지 대부분의 작업이 해당 노드에서 로컬하게 수행되므로, 네트워크 비 접속 상태에서도 작업 수행이 가능하다. 그러므로 이동 에이전트는 접속이 불안정하거나 부하가 높은 네트워크 환경에서 트래픽을 줄이고 채널을 효율적으로 사용할 수 있다. 그러나 이동 에이전트 시스템은 해당 서버에 실행 환경인 에이전트 플랫폼이 존재해야 하며 이주 정책에 따라 성능의 차이가 있으며 보안 문제를 고려해야 한다[4,5,8].

## III. 멀티에이전트 구현

### 3.1 시스템 구조

멀티 에이전트의 구성은 능동적인 컨텐츠 전달 방식을 제공하는 푸시 에이전트, 노드 이주 및 임무 수행을 담당하는 이동 에이전트, 분산 객체의 투명성 및 정확한 정보 서비스를 지원하는 네이밍 에이전트, 시스템 자원을 관리하는 시스템 모니터링 에이전트로 구성된다. [그림 1]은 각 기능과 역할이 상이한 다양한 에이전트로 구성된 멀티 에이전트 시스템의 구조를 나타낸다.

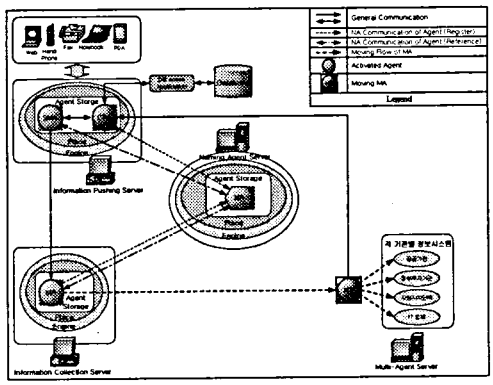


그림1. 멀티 에이전트 구조

### 3.2 이동 에이전트 구현

이동 에이전트는 사용자의 요청된 작업을 실행하고 다음 대상 목적 호스트로 이주하는 이동성 객체이다. 이동 에이전트는 에이전트 실행 호출 모듈(agent execution call module)을 통해 서버 에이전트 시스템의 작업 실행 모듈을 호출한다. 그리고 이동 에이전트 시스템의 에이전트 생성기에 의해 생성된 이동 에이전트는 네이밍 에이전트에 등록된 객체 참조자 리스트를 획득하여 이에 따라 생성된 원본 에이전트 객체를 기반으로 복제된다.

이렇게 복제된 에이전트 객체들은 객체 참조자 리스트에 포함된 정보에 따라 각 호스트의 위치로 이주를 수행한다. 만약, 각 호스트로 이주한 이동 에이전트가 호스트 장애로 인해 무한대기에 빠졌거나 파괴되었을 경우에는 Timestamp가 적용되어 소멸하게 된다. 그림2는 이동 에이전트 생성 및 실행알고리즘이다.

```
// Run : 에이전트 실행
  ServerObject_Instance 생성
  // Search Keyword 전달 : ServerObject
     작업 모듈 실행
     ServerObject_Instance.aggregate
     SearchResults 획득
     AgentRetract 요청
// Run End

// onLeave: 현 이동 에이전트 시스템 떠 나기 전 수행
   Agent Runner 생성
  // Agent Register Storage에 저장된 에이전트 객체 삭제
  AgentRunner에 AgentDelete 전달
// onLeave End

// Agent Server Object의 위치 정보 저장 변수

  Initialize ServerObject_Location

// Agent 작업 처리 결과 저장 변수
Initialize SearchResults

// onArrive:에이전트 시스템 도착 수행
     NamingAgent 생성

  // Naming Agent로부터 Server Object의 위치 정보 획득
  ServerObject_Location
= NamingAgent.lookup_agent()
// onArrive End

// getAgentProfile : 에이전트 상태 정보 획득
  AgentRunner 생성
  AgentRunner로부터 AgentProfile 획득
// getAgentProfile End
```

그림2. 이동에이전트 생성

### 3.3 네이밍 에이전트 구현

그림3은 네이밍 서비스에 서버 객체의 등록, 키워드 저장 및 객체 참조자 정보의 요청·반환의 수행과정을 나타낸다.

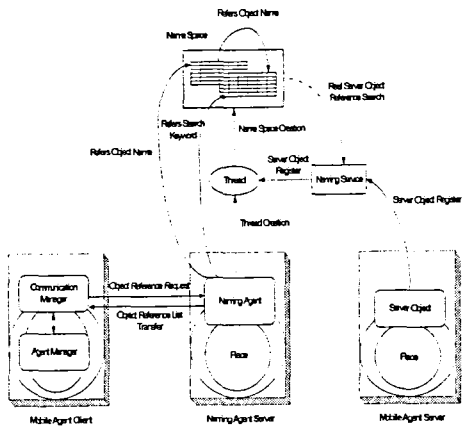


그림3. 네이밍 에이전트의 수행과정

네이밍 에이전트는 이동 에이전트의 이주 호스트의 위치 투명성을 보장하기 위해 각 에이전트 시스템 관련 객체들과 사용자가 요구하는 검색 키워드를 관리하는 데몬(Daemon) 객체이다. 네이밍 에이전트는 각 객체 정보들을 관리하기 위해 네이밍 서비스별로 스레드를 할당하여 각 스레드별 네임 스페이스를 생성한 후, 네임 스페이스 내의 메타데이터 테이블에 서버 객체, 플레이스, 이동 에이전트 시스템 등의 객체를 등록한다.

```
// ServerObject, Place, AgentSystem 등록 해쉬 테이블
HashTable ServerObjectHash, PlaceHash,
AgentSystemHash

// ServerObject에 대한 메타데이터 및 테이블
//ServerObject_ID,SearchKeword,
Total_Doc_Count, Hit_Count,
   Hit_Rate
MetaData ServerObject_MetaData
MetaTable ServerObject_MetaTable

// Registry 등록하여 Naming Agent 실행
Registry                reg
LocateRegistry.createRegistr(port);
reg.rebind(name, this);

NamingAgentManager 생성
NamingAgentManager에 의해 Register_쏙
ad 생성

ServerObject_MetaTable 생성

// Message Transmission
//    ServerObject    Register,    ServerObject
Unregister, ServerObject, Lookup, Search Info

SWITCH (Transmitted Message)
  {
   // Register Message
   //        ServerObjectInfo,        PlaceInfo,
AgentSystemInfo

CASE : Register Message
CASE : Unregister Message
CASE : Lookup Message
CASE : SearchKeyword Message
  }
```

그림4. 네이밍에이전트 생성

그림4의 네이밍 에이전트 관리자는 서버 객체의 등록 요청에 따라 네이밍 에이전트 스레드를 생성하여 해당 네이밍 서비스와 연결을 통해 각 멀티 에이전트 서버 객체의 이름과 객체 참조자를 rebind() 메서드에 의해 메타 테이블에 등록하고, 네이밍 에이전트 스레드의 네임 스페이스를 생성하여 등록된 서버 객체에 대한 메타 테이블을 생성한다.

등록된 객체들을 제거하기 위해 Unregister Message를 사용한다. Register Message는 네이밍 에이전트는 서버 객체와 플레이스, 그리고 이동 에이전트 시스템의 동적인 이름과 위치 정보를 유지하기 위한 메서드를 제공한다. 그리고 네이밍 에이전트는 네이밍 서비스와의 연결을 통해 각 네이밍 서비스에 멀티 에이전트 서버 객체들의 정보를

등록 및 관리하고 정보 수집 서버의 요청에 대해 멀티 에이전트 서버 정보를 반환함으로써 통합된 네이밍 서비스의 기능을 제공한다. 네이밍 에이전트는 네이밍 서비스에 호스트 정보와 멀티 에이전트 서버 객체의 이름을 등록하고, 추가로 검색에 사용되는 키워드들을 저장하기 위한 새로운 네임 스페이스를 생성하여 분산된 멀티 에이전트 서버의 객체 참조자와 계층적 검색 키워드들을 저장한다. 여기서, 계층적 검색 키워드는 데이터베이스의 테이블명과 주요 필드명, 테이블 관계 정보 등에 대한 데이터이다.

## Ⅳ. 결론

본 연구에서는 Web/Mobile 환경에서의 능동적 정보 서비스를 위한 멀티 에이전트 시스템에서는 기존의 웹 기반 정보 서비스가 가진 단점들을 해결하고 다양한 비즈니스 자료에 접근하여 정보를 수집하기 위해서 멀티 에이전트를 구현하였다. 이동 에이전트는 CORBA 기반의 이동 객체 형태로 에이전트 호출 모듈만을 포함한 에이전트 객체와 작업 실행 모듈을 가진 또 다른 객체로 분리하여 모듈크기에 따른 네트워크 부하를 감소시킬 수 있도록 하였다. 네트워크 트래픽 발생 시점에서 에이전트 이주시 기존 사용자에 의한 수동적 라우팅 스케줄(Routing Schedule)지정 방식에서 탈피하여 목적 노드까지의 최적 경로를 통해 물리적인 장애 발생 시 이주를 보장하였다.

또한, 이주 노드의 위치 투명성 및 초기 라우팅 스케줄을 지정하여 분산 객체의 위치 정보를 통합 및 관리하고 객체 참조자 정보를 추출하며 에이전트 등록에 의한 메타데이터 생성, 멀티 에이전트의 각 에이전트간의 협력, 메타데이터의 정보에 의한 노드 이주 및 노드 이주에 따른 메타데이터의 갱신 과정을 담당하는 네이밍 에이전트를 설계 및 구현 하였다.

향후 연구과제로는 멀티에이전트 구현시 고려하지 않은 네트웍 지연, 노드 처리 시간, 호스트 프로세싱 시간등 여러 메타데이터 정보를 조합하여 최적의 노드 이주 정책을 세우고 네이밍 에이전트에 캐시 기법을 적용하는 것이다.

## 참고문헌

[1] 고현, 김광종, 이연식, "분산 정보 서비슬 위한 Corba기반의 멀티 에이전트 모델", 한국정보처리학회 춘계학술발표 논문집, 제9권, 제1호, pp.327~330, 2002

[2] 김광명, 이연식, "메타 데이터를 이용한 네이밍 에이전트 설계", 한국멀티미디어학회 춘계학술발표논문집(하), 제5권, 제1호

[3] 김항찬, "멀티 에이전트를 웹 서비스로 활용하기 위한 멀티 에이전트 플랫폼", 경북대학교 석사학위논문, 2003

[4] 양선옥,이종희 "웹 기반 코스 스케줄링을 위한 멀티 에이전트 시스템 (A Multi-agent System for Web-based Course Scheduling)", 멀티 미디어학회 논문지, Vol.6 No.6, 2003

[5] 오양훈, "Mobile Agent 기술을 이용한 이동 에이전트 시스템 구현", 광운대학교 대학원, 석사학위논문, 2000

[6] 전병국, "효율적인 이동 에이전트 시스템", 광운대학교 대학원 박사 학위논문, February, 2000

[7] Amjad Umar. Ph. D, [Distribute Computing and Client-Server System], Book Plus, 1996

[8] B. W. Baek, G.T. kim and H. Y. Yeom, "Timed Mobile Agent Planning for Distributed Information", ACM agents, 2001

[9] Tarlas, "A multiagent framework for a diagnostic and prognostic system", Georgia Institute of Technology, 2003