CORBA를 기반으로 한 분산 다중 에이전트 서비스 계층 설계

최형은°, 김태형
한양대학교 컴퓨터공학과
{hechoi°, tkim}@cse.hanyang.ac.kr

Designing A Distributed Multi-Agent Service Layer Based on CORBA

Hyong-Eun Choi°, Tae-Hyung Kim
Dept. of Computer Science & Engineering, Hanyang University

요 약

유비쿼터스 컴퓨팅과 같이 서비스 지향적인 환경에서 에이전트 기술은 효과적인 시스템 구축 방법이다. 그러나 오늘날과 같이 이기종 분산 시스템에서 에이전트 시스템을 구축하기 위해선 분산 시스템의 투명성을 제공하는 분산 객체 기술이 필요하며 이에 CORBA를 기반으로 하는 에이전트 서비스 계층을 설계한다.

1. 서 론

현재와 같이 이기종 분산 시스템으로 이루어진 컴퓨팅 환경에서 CORBA와 같은 분산 객체 기술은 효과적인 시스템 구축 방법으로 여겨지고 있으며 다중 에이전트 시스템 역시 다양한 컴퓨팅 환경에서 에이전트간에 협업을 통해 문제를 해결하기 위한 방법을 제시하고 있다. 소프트웨어 엔지니어링 측면에서 본다면 에이전트는 객체의 확장된 형태로 볼 수 있으나 저수준의 메소드 호출을 통해 상태를 변화시키는 객체와 달리 독립적이고 능동적이라 할 수 있다. 또한 에이전트는 상호작용을 위해 에이전트 통신 언어라는 메시지를 주고 받는 유연한 통신 방식을 사용하여 시스템의 확장성을 높일 수 있다. 이에 본 논문에서는 분산 시스템의 투명성을 제공하는 CORBA를 기반 구조로 사용하면서 에이전트 통신 언어 (Agent Communication Language)를 이용한 에이전트간 통신을 지원하여 에이전트 기술이 갖는 장점을 향상 시킬 수 있는 분산 다중 에이전트용 서비스 계층을 설계한다.

2. 관련연구

2.1 에이전트

에이전트란 자율적인 작업 수행 능력을 지닌 컴퓨터 시스템[1]으로 변화된 상황에 능동적으로 대처하는 특성을 지닌다. 다중 에이전트란 하나의 에이전트로 해결하지 못하는 복잡한 문제의 해결을 위하여 여러 에이전트간의 협동을 통해 작업을 수행하는 에이전트를 말한다

2.2 CORBA

CORBA[2]는 The Common Object Request Broker Architecture의 약어로 OMG에서 제안한 분산 객체 기술이다. CORBA는 분산 이기종 컴퓨팅 환경하에서 객체간의 상호 운용성을 지원하는 미들웨어로서 이질적인 컴퓨팅 환경에 분산되어 있는 소프트웨어 컴포넌트를 객체화하여, 그 인터페이스를 ORB(Object Request Broker)에 등록하고 클라이언트는 원격 객체에 대해 마치 지역 객체처럼 접근하도록 하는 방법이다.

2.3 CORBA 서비스

2.3.1 Naming 서비스

객체 정보의 등록 및 검색과 관련된 기능을 제공하는 CORBA 서비스[3]로서 구현 객체들은 이름으로 자신의 정보를 등록하고 클라이언트는 자신이 필요로 하는 객체의 정보를 이름으로 검색하여 구현 객체의 정보를 얻는다. 취득한 구현객체의 정보를 이용하여 해당 객체에 접근할 수 있도록 해주는 서비스이다.

2.3.2 Trader 서비스

객체를 등록하고 검색할 수 있는 서비스로 Naming 서비스와 달리 객체에 대한 정보를 설명 형식으로 제공하고 클라이언트는 필요한 기능을 질의문으로 만들어 검색하는 기능을 제공한다.

2.3.3 Notification 서비스

Notification 서비스는 기존의 CORBA Event Service를 확장한 서비스로 이벤트 채널에 기반한 통신을 지원하며 다음과 같은 새로운 특성이 추가되었다.

- Structured Events
- Event Filtering
- Subscriptions
- Batched Events
- Quality of Service Properties

3. 에이전트 서비스 계층 설계

에이전트 서비스 계층(Agent Service Layer)은 CORBA를 기반으로 하고 있으며 ASL상에서의 에이전트는 CORBA 객체로 표현할 수 있다. 이에 에이전트 협업에 필요한 요소들을 살펴보고 이를 CORBA에서 사용 가능한 방법으로 맵핑하여 에이전트 어플리케이션에 분산 시스템의 투명성을 제공하는 방법을 제시한다.

3.1 에이전트 통신 언어

에이전트 시스템에서는 에이전트간의 협업을 위해 의미를 정의한 메시지를 교환하는데, 이를 에이전트 통신 언어(Agent Communication Language)라고 한다. 현재 ACL에는 FIPA-ACL[4], KQML등의 언어들이 사용되고 있으며 기본적인 메시지 구조에는 큰 차이가 없다. 본 논문에서는 FIPA-ACL Parameter를 사용하며 그 구조는 다음과 같다.

Parameter	Category of Parameters
performative	Type of communicative acts
sender	Participant in communication
receiver	Participant in communication
reply-to	Participant in communication
content	Content of message
language	Description of content
encoding	Description of content
ontology	Description of content
protocol	Control of conversation
conversation-id	Control of conversation
reply-with	Control of conversation
in-reply-to	Control of conversation
Reply-by	Control of conversation

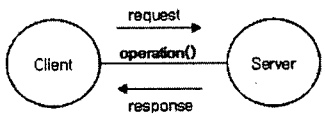
<표 1. FIPA-ACL Message Parameters>

각 인자는 FIPA-ACL에서 명세한 값을 이용하여 메시지를 만들 수 있고 이를 이용하여 에이전트 간에 서비스를 요청하거나 회신 받을 수 있다. 메시지 전송을 위해서는 ACL을 IDL로 맵핑하는 과정이 필요하며 다음과 같이 정의할 수 있다.

```
module AgentServiceLayer {
   struct ACLmsg {
      string performative;
      string sender;
      string receiver;
      ……
   };
   Typedef sequence<ACLmsg> ACLmsg_sequence;
};
```

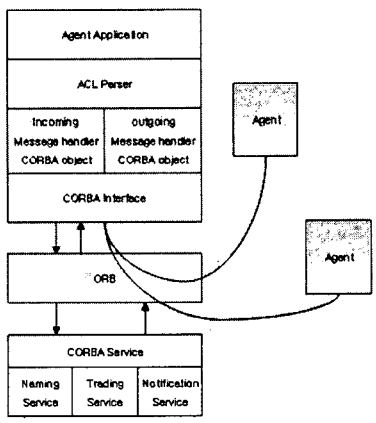
3.2 메시지 전송 방법

CORBA에서의 메시지 전송은 메소드를 호출한다는 의미하며 일반적인 호출 방식은 일반적으로 <그림1>과 같은 동기적인 모델을 사용하고 있다.



<그림 1. 요청/응답 모델>

하지만 에이전트 간의 메시지 전송은 비동기적인 특성을 갖고 있으며 이는 오늘날의 서비스 지향적인 시스템에서 사용되는 방식으로 메시지 전송에 유연성을 제공한다. 이를 위해 CORBA에서는 비동기적인 메소드 호출방식[5]과 이벤트 채널을 통한 통신 방법을 제공하고 있으며 에이전트 시스템의 특성을 고려한다면 객체간의 분리된 통신기법을 제공하는 이벤트 채널 통신이 적합하다. CORBA에서는 이벤트 서비스의 확장된 형태로 Notification 서비스가 제공되고 있으며 ACL 메시지를 구조화된 이벤트 객체로 만든 뒤 Notification 서비스가 제공하는 이벤트 채널을 통해 메시지를 전송할 수 있다.



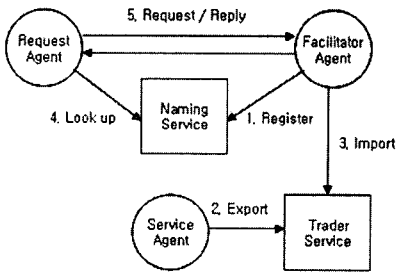
<그림 2. 에이전트 서비스 계층 구조>

<그림 2>는 CORBA를 기반으로 한 에이전트 서비스 계층 구조를 나타낸 것으로 ACL 메시지를 처리하는 ACL Parser와 이벤트 채널을 관리하는 객체들로 구성되어 있다. 이 모든 동작은 CORBA 인터페이스를 통해 ORB와 CORBA Service 기능을 이용하여 다른 에이전트들과의 상호작용을 가능하게 한다.

3.3 에이전트 탐색을 위한 Facilitator 에이전트

에이전트 시스템에서 서비스를 요청하는 에이전트는 자신에게 적합한 서비스를 제공하는 에이전트를 찾기 위해

Facilitator 에이전트[1]와 같은 시스템 에이전트에게 질의를 하는데 CORBA에서도 동적으로 객체의 참조를 얻기 위한 서비스를 Naming과 Trader 서비스를 통해 제공하고 있다. CORBA 서비스는 에이전트 통신 언어를 이용한 질의를 인식할 수 없기 때문에 이를 처리하고 적절한 서비스를 호출해 줄 Facilitator 에이전트가 필요하다. CORBA 서비스를 이용한 Facilitator 에이전트의 역할은 <그림3>에 나타나 있으며 동작 시나리오는 아래와 같다.



<그림 3. CORBA서비스를 이용한 Facilitator 에이전트>

(1) Facilitator 에이전트는 시스템 에이전트로 자신의 객체참조를 제공하기 위해 Naming 서비스에 등록을 한다.
(2) 서비스 제공 에이전트는 Trader 서비스에 에이전트 이름, 제공하는 서비스에 대한 설명, 통신을 위한 이벤트 채널 이름 등의 내용을 익스포트(export)한다.
(3) Facilitator 에이전트는 서비스 제공 에이전트가 익스포트(export)한 내용을 주기적으로 임포트(import)하여 서비스 제공 목록을 구성한다.
(4) 서비스 요청 에이전트는 Naming 서비스를 이용하여 Facilitator 에이전트의 객체를 참조한다.
(5) 서비스 요청 에이전트는 자신에게 필요한 서비스 목록을 질의하고 회신을 받는다.

이와 같은 Facilitator 에이전트를 위한 IDL은 다음과 같이 정의할 수 있다.

```
Module AgentServiceLayer {
  struct ServiceAgent {
    string AgentName;
    string ServiceDescription;
    string EventChannel;
    ……
  };
  Typedef sequence<ServiceAgent>ServiceAgent_seq;
  Interface FacilitatorAgent {
    Any getServiceAgentList();
  };
};
```

4. 결론 및 향후과제

다양한 컴퓨팅 환경에서 문제 해결을 위해 다른 기술과의 연동에 대한 필요성은 점차 높아지고 있다. 분산 객체 기술 또한 효과적인 에이전트 시스템 구축에 보완적 역할을 할 수 있고 이에 본 논문에서 CORBA를 기반으로 한 에이전트 서비스 계층을 설계하였다. 지금까지는 에이전트간의 협업을 위한 메시지 전송 방법의 기본구조 구축에 중점을 두었으나 향후 과제로서 실시간적 서비스 제공 기법이나 다양한 지식 제공을 위한 온톨로지 정의 등 에이전트 시스템에서 다루어야 될 요소들을 추가적으로 보완하는 연구를 진행중이다.

5. 참고문헌

[1] Agent Platform Special Interest Group, " Agent Technology Green Paper " - http://www.objs.com/agent/agents_Green_Paper_v100.doc
[2] OMG, " Common Object Request Broker Architecture: Core Specification " - http://www.omg.org/docs/formal/04-03-01.pdf
[3] OMG, " CORBAServices: Common Object Services Specification" - http://www.omg.org/cgi-bin/doc?formal/2004-10-11.pdf
[4] FIPA, " FIPA ACL Message Structure Specification " - http://www.fipa.org/specs/fipa00061/SC00061G.pdf
[5] Steve Vinoski, " Invocation Styles " - IEEE INTERNET COMPUTING 1089-7801/03/