분산시스템에서 객체공유를 위한 상호협력모델

정진섭" · 윤인숙" · 이재완"

"군산대학교

A Cooperation Model for Object Sharing in Distributed Systems

Jin-Seob Jeong" · In-Suk Yun" · Jae-Wan Lee"

"Kunsan National University

E-mail : jwlee@knusun1.kunsan.ac.kr

요 약

이질적인 네트워크를 기반으로한 분산 객체지향 환경에서 대규모의 시스템 증가로 인한 분산객체들의 관리의 복잡성을 해결하기 위하여 분산객체들간의 효과적인 상호협력정책들이 필요하게 되었다. 따라서 본 논문에서는 트레이더의 설비, 목적과 목적의 가중치에 따른 세가지 다른 상호협력 모델(단순협력(light weight trader), 단순교섭(simple negotiation), 연합(federation))을 살펴보므로써 클라이언트에게 보다 나은 서비스 속도 제공 및 선택의 폭을 넓히기 위한 트레이딩 상호협력을 제안했다.

ABSTRACT

In distributed object oriented environment based upon wide heterogeneous network, effective cooperation policies between/among distributed objects are needed to resolve a complexity of management of distributed objects because of growing of a large scale of systems. Thus, in this paper, we propose three trading cooperation models between/among traders for supporting a high speed and a wide selection of trader service for clients, by considering three different cooperation models(light weight trader, simple negotiation and federation) depending upon their facilities, goals, and weights of goals.

Ⅰ. 서 론

오늘날 분산 컴퓨팅 환경의 분산 시스템을 구성하는 시스템들은 광범위한 네트워크를 기반으로한 분산처리 기술에 객체지향기술의 도입으로 인하여 분산 시스템의 크기가 거대해지고 응용 서비스 기능들이 보다 다양해지고 있으며, 이들간에 많은 상호협력이 요구되고 있다.

본 논문에서는 분산처리 기술에 객체지향기술을 도입한 OMG(Object Management Group)와 TINA(Telecommunications Information Networking Architecture)에서 제시한 트레이딩 서비스를 기반으로[1][2], 분산 객체지향 환경에서 분산 시스템을 이루는 어플리케이션들(서비스 객체)사이의 상호운용성을 제공하고 이기종 다중 객체 시스템들간의 접속에 대한 투명성을 제공하는 트레이딩 서비스를 확장하는데 초점을 맞추고 있다. 특히, 서로 다른 시스템의 규모가 커짐에 따라 분산된 객체의 수가 비례하여 발생하는 관리의 복잡성과 분산 객체간의 빈번한 상호작용으로 인한 네트워크상의 트래픽 문제점을 해결하는 트레이딩 서비스의 이점을 확장하여, 사용자들에게 투명성을 제공하고 확장 및 서비스의 재사용성 그리고 서비스의 질을 높이기 위해 각각의 영역에서 독립적으로 운용되는 트레이더들을 연결한 상호협력의 성능향상에 목적을 두고 있다.

따라서 본 논문에서는 서로 다른 영역에 위치한 독립적인 트레이더들의 서비스 타입, 서비스들의 수와 서비스 요청 빈도수에 따라 협력모델들을 분류하였으며[3][4][5], 이 협력모델들은 하나의 호스트 트레이더(Host Trader)와 여러 개의 로컬 트레이더(Local Trader)를 묶어 하나의 트레이딩 도메인을 형성하고 호스트 트레이더간에 협력이 이루어질 수 있도록 모델링(modelling)하였다. 제안된 협력모델에 따라 트레이더간의 접속 과정을 구현하므로서 분산시스템을 이루는 객체들의 접속을 효과적으로 할 수 있으며, 클라이언트에게 서비스 선택의 폭을 좀 더 넓힐 수 있도록 하였

다.

본 논문의 구성은 다음과 같다. 관련연구에서는 CORBA(Common Object Request Broker Architecture)와 트레이딩 서비스에 대해서 알아보고 3장에서는 다른 도메인 상의 트레이더들을 연결시켜주는 기능을 갖는 호스트 트레이더의 구조와 기능을 살펴본다. 4장에서는 트레이더간의 상호협력의 필요성과 트레이딩 상호협력 모델 그리고 호스트 트레이더를 통한 서비스 수행을 설명한다. 끝으로 5장에서는 결론으로서 앞으로 연구해야 할 내용에 대해서 기술한다.

1. 관련연구

1.1 CORBA(Common Object Request Broker Architecture)

이종의 분산환경에서 이질성과 언어의 종속성이라는 문제를 해결하기 위하여 OMG에서 제안한 CORBA는 이기종간 분산 객체 컴퓨팅을 위한 표준안이며 서비스를 제공하는 객체와 서비스를 받고자 하는 클라이언트 사이의 중개자로서 ORB(Object Request Broker)를 이용하는 시스템으로 클라이언트는 원하는 서비스에 관한 요청(request)를 생성하고 이것을 ORB가 객체구현(Object Implementation)에 전달한다. 요청을 생성하는 방법에는 IDL 컴파일러로부터 생성된 클라이언트 스텁(stub)을 이용하는 정적인 방법과 인터페이스 저장소(Interface Repository)를 이용하는 동적인 방법이 있다.

CORBA는 객체 기술을 가지고 이종 환경에서의 미들웨어(middleware)로서 기능을 하며 또한, CORBA의 분산객체(Distributed Object)는 네트워크상의 어느 곳에서나 존재할 수 있으며 리모트 클라이언트에 의해 액세스 될 수 있고 분산 서버 객체를 만든 언어와 컴파일러는 클라이언트에 대해 완전히 투명하며, 클라이언트는 분산 객체가 어디 있고 어떤 운영체제에서 돌아가는지 알 필요없이 ORB를 통해 원하는 객체에 링크할 수 있는데 이러한 특징들이 트레이더의 구현을 직접적으로 도와줄 수 있기 때문에 CORBA 위에서 트레이더를 구현하였다.

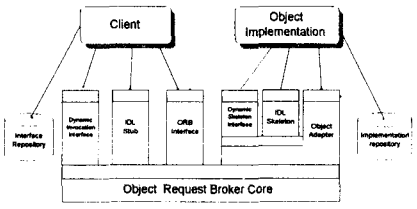

그림 1. CORBA의 기본 구성도

1.2 트레이딩 서비스

트레이딩 서비스는 분산 환경에서 클라이언트가 어떤 종류의 서비스를 필요로 하는지는 알고 있지만 어떠한 서버가 클라이언트의 요구사항에 적합한 서비스를 제공하는지 모를 경우, 적절한 서버를 이름이 아닌 속성(attribute)을 가지고 선택할 수 있는 서비스이다[3][4][5][6].

분산처리 환경에서 분산 시스템을 이루는 객체들의 상호 접속을 효율적으로 하기 위해 트레이딩 서비스를 이용하지만 단일 트레이더가 관리해야 할 정보의 양이 증가하게 되면 트레이더의 성능 저하의 문제점이 발생하게 된다[7][8]. 그래서 각각의 영역내에 트레이더를 두고 이들간의 상호협력이 필요로 하게 되었다. 이로 인하여, 트레이더는 자신의 영역내에서 관리하는 정보 이외에 분산된 트레이딩 정보를 제공할 수 있게 되고, 클라이언트(임포터)들에게 서비스 선택의 폭을 넓혀줄 뿐만 아니라 보다 나은 서비스를 제공할 수 있다.

다음 그림 2는 트레이더 상호협력(trader cooperation) 과정을 나타낸다.

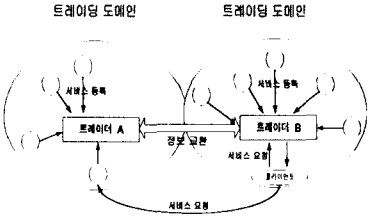

그림 2. 트레이더 상호협력

II. 본 론

1. 호스트 트레이더

1.1 호스트 트레이더의 구조

호스트 트레이더는 트레이더간의 상호협력을 위한 중재자로서 각각 다른 트레이딩 도메인 상에서 독립적으로 운영된다. 호스트 트레이더를 이용한 상호협력은 각각의 트레이딩 도메인 상에 로컬 트레이더와 호스트 트레이더를 두고 있기 때문에 임의의 지점에서 트레이딩이 요청될지라도 동등한 트레이더간의 정보교환이 가능하며 대규모 분산시스템에서 효과적이다. 호스트 트레이더는 다른 도메인상의 호스트 트레이더와의 상호작용시에는 자신의 트레이딩 도메인의 대표자로서 역할을 수행하며 동시에 하나의 로컬 트레이더로서의 역할을 수행한다.

다른 도메인 상의 트레이더들을 연결시켜주는 호스트 트레이더의 구조는 다음과 같다.

■관리자 객체(MO : Management Object)

로컬 트레이더의 트레이딩 서비스 요청을 받아 먼저 타입관리자 객체에게 서비스 타입을 검색하게 하고, 다른 도메인 상에 있는 호스트 트레이더

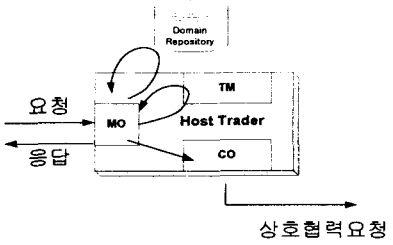

그림 3. 호스트 트레이더의 구조

에 대한 위치정보를 얻기위해 정보 저장소 객체(I/RO)에게 요청한 후, 얻은 정보를 가지고 상호협력 객체(CO)를 통해 다른 도메인 상의 호스트 트레이더에게 서비스 요청을 한다. 또한 트레이더가 처리하여 반환된 레퍼런스를 요청한 로컬트레이더에게 전달한다.

■도메인 저장소 객체(Domain Repository Object)

관리자 객체(MO)에게 연결되어 있는 로컬 트레이더 또는 호스트 트레이더의 정보를 관리하여 그 위치정보를 관리자 객체(MO)에게 반환한다.

■상호협력 객체(CO : Cooperation Object)

다른 도메인 상의 호스트 트레이더와의 연결을 위해 로컬 트레이더 또는 호스트 트레이더의 위치정보를 추가, 삭제, 그리고 수정하여 다른 트레이더에게 서비스를 요청한다.

■타입관리자 객체(TM : Type Manager Object)

도메인 상에 존재하는 로컬트레이더에 대한 서비스 타입을 관리하며 그 정보를 관리자 객체에게 반환한다.

1.2. 호스트 트레이더의 기능

각각의 다른 도메인내의 호스트 트레이더는 어느 트레이더가 어떠한 서비스를 제공해 줄 수 있는지에 대한 서비스 제공 목록을 가지고 있는 인터페이스 레포지토리(Interface Repository) 역할을 담당하고, 상호협력을 위해서 다른 도메인 상에 존재하는 호스트 트레이더의 설비와 목적, 위치정보 등에 대한 리스트를 가지고 있다.

· 트레이딩 서비스 요청 기능 : 트레이딩 서비스를 받고자하는 객체(임포터, 익스포터)들의 요청에 대한 오퍼레이션을 트레이더에게 전해주는 기능

· 연결에 대한 정보 관리 기능 : 다른 트레이더와 상호협력을 위해 연결된 호스트 트레이더나 로컬트레이더에 대한 정보를 관리할 수 있는 기능

· 상호협력 기능 : 다른 트레이더에게 임포트 요청을 전달하고 얻어진 결과를 처리할 수 있는 기능

· 서비스 타입에 대한 정보관리 기능 : 다른 트레이더와 상호협력을 위해 연결된 호스트 트레이더나 로컬트레이더에 대한 서비스 타입을 관리할 수 있는 기능

1.3. 호스트 트레이더와 트레이더간의 접속

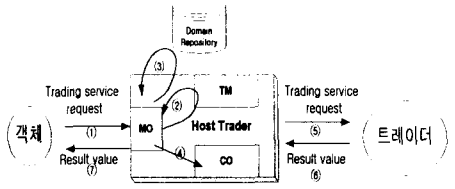

그림 4.호스트 트레이더와 트레이더간의 접속

임포터 또는 익스포터 객체가 트레이더에게 트레이딩 서비스 요청을 하고자 호스트 트레이더의 관리자 객체에게 요청하면, 관리자 객체는 먼저 타입관리자 객체(TM)에게 서비스 타입에 대한 검색을 요청하고 도메인 레포지토리 객체에게 트레이더 위치에 관련된 정보에 대하여 검색한다. 관리자 객체는 얻은 레퍼런스를 가지고 해당 트레이더에게 트레이딩 서비스를 요청하기 위해 상호협력 객체에게 메시지를 전달한다. 상호협력 객체는 해당 트레이더에게 트레이딩 서비스 요청을 전달하고 트레이더는 트레이딩 서비스 요청에 대하여 처리한 결과 값을 반환하면 호스트 트레이더는 트레이더로부터 얻어진 결과 값을 요청한 객체에게 반환한다.

2. 상호협력

분산처리 환경에서 분산 시스템을 이루는 객체들의 상호접속을 효율적으로 하기위해 트레이딩 서비스를 이용하지만 단일 트레이더가 관리해야 할 정보의 양이 증가하게 되면 트레이더의 성능저하 문제점이 발생하게 된다. 이로 인하여 각각의 영역내에 호스트 트레이더를 두고 이들간의 상호협력이 이루어지로록 하므로서 트레이더는 자신의 영역 내에서 관리하는 정보 이외에 분산된 트레이딩 정보를 제공할 수 있게 되고, 클라이언트(임포터)들에게 보다 다수의 서비스 선택의 폭을 넓혀줄 뿐만 아니라 보다 나은 서비스를 제공할 수 있다. 그래서, 클라이언트(임포터)가 많은 개별적인 트레이더들로부터 해당 서버들에 대한 정보를 얻어올 수 있지만, 다른 위치의 트레이더와의 상호협력을 통한 상호작용을 하므로서 더욱 편리함을 가져온다.

트레이더의 상호협력은 각각의 트레이딩 도메인 상의 트레이더가 갖는 정책과 객체들의 상호접속을 위한 제어에 상관없이 각 객체가 가지고 있는 서비스의 공유가 이루어지므로 인해, 다른

도메인에 있는 시스템과 연동을 가능하게 한다. 그로 인해, 한 도메인은 상호협력된 다른 도메인과 정보를 공유할 수 있고, 적당한 서비스 오퍼를 찾도록 할 수 있다. 따라서, 클라이언트는 다른 트레이딩 도메인 상에서 관리되는 객체에 대한 레퍼런스를 트레이더로부터 획득하여 서비스 요청을 할 수 있다.

2.1. 상호협력 주체

분산 시스템에서 클라이언트들에게 좀 더 빠른 서비스 속도와 보다 넓은 서비스 폭을 제공하기 위해 상호협력하는 주체(서브젝트)로서 트레이더들이 그들의 설비와 목적에 따라 상호협력한다. 상호협력하는 방법은 그들의 설비와 목적에 의해 결정된다. 따라서 상호협력에 관한 서브젝트(주체)들의 활동은 다음과 같이 정의된다.

Subject = ((facilities, cost), (goals, weights))

이 모델에서 설비들은 다음과 같이 세가지 구성요소를 가진다.

· 동작(Actions) : 서브젝트가 수행할 수 있는 동작들

· 자원(Resources) : 동작을 수행하는 데 필요한 자원들

· 정보(Knowledge) : 자신 뿐만아니라 다른 서브젝트들의 환경에 대한 정보

2.1.1. 설비(Facilities)

동작(Actions)은 클라이언트 요청들과 서버 오퍼들의 처리에 관계한다. 두가지 경우 예를 들어, 클라이언트 요청에서의 탐색범위와 선택기준이나 제한된 오퍼들에 대한 기준과 같은 많은 옵션들이 포함된다. 각각의 옵션이 다른 트레이더 동작을 요구하지만 트레이더는 모든 동작들을 수행할 수 없기 때문에 트레이더 간의 상호협력이 더욱 필요하다.

자원(Resource)은 컴퓨팅과 통신자원 특히 디스크 공간 뿐만아니라 트레이더 동작을 수행하는데 이용되는 알고리즘으로 구성된다. 어떤 자원들은 한정된 용량을 가지고 있어서 설비들의 양을 제한한다. 이것은 특히 트레이더의 응답시간과 저장될 수 있는 서비스 오퍼들의 수에 대해서 중요한 역할을 한다.

정보(Knowledge)는 클라이언트에 의한 서비스 이용내력, 저장된 서버들의 특성들과 다른 트레이더들의 특성에 관계한다. 한 트레이더는 한 시스템 내에 있는 모든 트레이더들이나 공간적으로 이웃한 트레이더들의 부집합에 대한 정보를 안다. 두 번째 경우에서 트레이더들의 고립을 피하기 위해서는 모든 트레이더들의 부집합 연합은 상호협력 하고자 하는 모든 트레이더들의 집합과 동일해야 한다.

따라서 여러 가지 설비들을 가진 다른 트레이더들이 존재할 수 있고 다른 트레이더들에게 자신의 설비를 제공할 수 있고 익스포팅하는 양은 목적의 가중치에 달려있다.

2.1.2. 목적(Goals)

상호협력하려는 트레이더들간의 목적은 클라이언트와 서버 입장에서의 목적과 트레이더의 특정한 목적으로 나뉘어질 수 있고 부분적으로 기능과 질 면에서 다시 세분할 수 있다. 클라이언트 입장에서의 목적은 각각의 클라이언트 요청에 대해서 응답하는 것과 클라이언트 기준에 따라 클라이언트와 서버 사이에 서버 응답시간을 최소화하는 등으로 할당을 최적화하는 것이다. 서버 입장에서의 목적은 클라이언트 요청에 대한 응답 가능성을 증가시키기 위한 포텐셜 클라이언트 집합을 제공하고 서버 기준에 따라 클라이언트와 서버 사이에 최적화 할당을 하는 것이다. 그리고 트레이더 특정한 목적은 높은 트레이더 서비스 질을 제공하고 필요하다면 다른 도메인 상의 트레이더와도 상호협력을 할 수 있도록 하는 것이다.

2.2 상호협력 모델

본 논문에서는 현재 상태를 고려하여 합리적인 선택을 하기위해 다음 기준에 따라 분류된 세가지 유형의 상호협력 형태를 이용한다.

· 상호협력에 참여하는 트레이더들의 서비스 타입

· 익스포팅되거나 임포트하려는 서비스들의 수

· 서비스 이용의 빈도수

위에 나타낸 기준에 따라 분류된 세 가지 유형의 협력 형태는 다음과 같다.

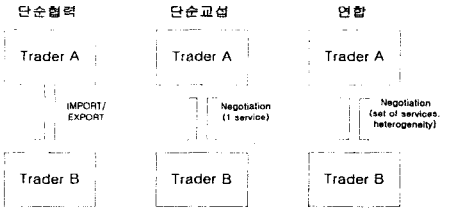

그림 5. 트레이더의 상호협력 형태

다른 트레이더에 대해서 클라이언트로서 동작하여 서비스를 수행하는 단순협력(Light Weight Cooperation), 다른 트레이더들과 한 서비스에 대해서 상호협력을 교섭하는 단순교섭(Simple Negotiation)과 트레이더들간의 이종성 문제를 해결하고 퍼실러티들의 공유양을 교섭하는 연합(Federation)이 있다. 이러한 상호협력의 형태들은 트레이더들의 능력면에서 뿐만아니라 자원소비양에서 다르다.

세 가지 상호협력 형태중에서 가장 적절한 형태를 선택하려는 알고리즘은 자원소비가 가장 낮

은 것을 이용한다. 단순협력과 단순교섭은 상호협력에 참여하는 트레이더들의 서비스 타입이 동일해야만 하고 더욱이 한 서비스 타입만을 허용한다. 그러나 단순협력은 서비스 요청 빈도수가 적을 때 이용되고 정보열람과 같은 클라이언트의 기능만을 가지게 되며 반면에 단순교섭은 서비스 요청 빈도수가 많을 때 이용되며 클라이언트 뿐만아니라 정보를 제공해주는 서버의 역할도 동시에 한다. 연합은 상호협력 형태중에서 가장 자원소비가 많고 복잡하기 때문에 필요한 경우에만 이용한다. 이 상호협력 형태는 서비스 타입이 서로 다른 경우에 적용되고 많은 서버들이 포함된다.

2.3. 호스트 트레이더와의 상호작용

호스트 트레이더가 제공하는 인터페이스는 그림 6에서와 같이 크게 세가지로 나뉠 수 있다. 로컬 트레이더가 호스트 트레이더에게 임포터하거나 익스포트 할 때 사용하는 인터페이스인 ① Local_Tr_Interface, 호스트 트레이더가 트레이딩 도메인 관리, 컨텍스트 관리 및 다른 도메인 상에 존재하는 다른 호스트 트레이더의 정보관리를 위해서 사용하는 인터페이스인 ② HostMgmt_Interface, 그리고 트레이더간의 상호협력을 위한 것으로 다른 로컬 트레이더나 다른 도메인 상에 존재하는 호스트 트레이더에게 상호협력을 요청하고자 할 때 사용되는 인터페이스인 ③Coop_Interface가 있다.

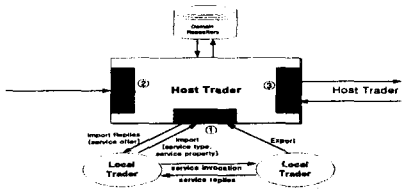

그림 6 호스트 트레이더와의 상호작용

2.4. 상호협력모델과 서비스 수행

본 논문에서 제안한 트레이더 상호협력은 각각의 트레이딩 도메인 상에 하나의 호스트 트레이더(Host Trader)와 여러개의 로컬 트레이더(Local Trader)를 묶어 하나의 트레이딩 도메인을 형성하고, 호스트 트레이더를 통한 트레이딩 상호협력에 기초한다.

(1) 서비스 타입이 동일한 경우

A. 단순협력(Light Weight Cooperation)

하나의 호스트 트레이더와 여러 개의 로컬트레이더로 묶여진 한 도메인 상에서 로컬트레이더 A가 호스트 트레이더에게 정보열람을 요청한 경우 발생되는 상호협력 형태로서 이 때 로컬트레이더는 서비스를 요청하는 클라이언트로서 동작하게 된다.

B. 단순교섭(Simple Negotiation)

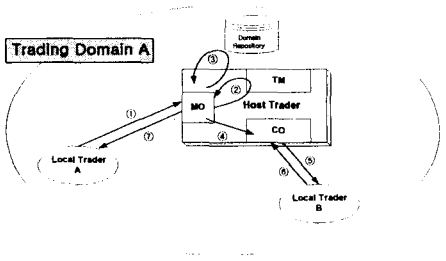

그림 7. 단순협력과 단순교섭 서비스 수행

한 도메인내에서 하나의 호스트 트레이더와 여러 개의 로컬트레이더간의 상호협력을 돕는 형태로 앞의 단순협력 모델과 같이 서비스가 수행되나 로컬트레이더 A는 이 때 단순히 정보열람의 클라이언트 역할 뿐아니라 정보를 제공해주는 서버의 역할도 동시에 가능하다.

임포터 또는 익스포터의 서비스 요청을 받은 로컬트레이더 A가 호스트 트레이더에게 트레이딩 서비스 요청을 하여 서비스 타입을 확인한 후 서비스 타입이 동일하고 로컬트레이더 A가 정보열람만을 원할 때는 관리자 객체는 도메인 레포지토리 객체에게 트레이더 위치에 관련된 정보에 대해 검색하여 트레이더의 레퍼런스를 얻는다. 관리자 객체는 검색을 통해 얻어진 트레이더의 레퍼런스를 가지고 해당 트레이더에게 트레이딩 서비스를 요청하기 위해 상호협력 객체에게 메시지를 전달하고, 상호협력 객체는 해당 트레이더에게 트레이딩 서비스를 요청한다. 트레이딩 서비스를 요청받은 트레이더는 요청에 대한 결과값을 반환하고, 호스트 트레이더는 로컬트레이더 B로부터 얻은 결과값을 요청한 로컬트레이더 A에게 전달한다.

(2) 서비스 타입이 다른 경우

하나의 호스트 트레이더와 여러 개의 로컬트레이더들로 구성된 각각의 트레이딩 도메인 A, B, C에서 상호간에 서비스 수행을 돕는 형태로 트레이딩 도메인 A에 있는 로컬트레이더 A가 같은 트레이딩 도메인 영역에 있는 호스트 트레이더에게 서비스 타입이 다른 서비스 요청을 하게 되는 경우 다른 트레이딩 도메인에 있는 호스트 트레이더들에게 Broadcasting하여 다른 트레이딩 도메인 영역에 있는 로컬트레이더의 정보를 이용할 수 있다.

트레이딩 도메인 A에 있는 로컬트레이더 A가 트레이딩 서비스 요청을 하면 호스트 트레이더의 관리자 객체는 서비스 타입에 대한 검색을 요청하여 서비스 타입이 다를 경우 관리자 객체는 다른 도메인 상의 호스트 트레이더와 상호협력하기 위해 도메인 레포지토리 객체에게 다른 호스트 트레이더에 대한 정보를 검색하고 결과를 전달받는다. 관리자 객체는 도메인 레포지토리 객체로부

터 얻은 호스트 트레이더에 대한 위치정보를 상호협력 객체에게 전달하고, 상호협력 객체는 다른 도메인 상에 존재하는 각각의 호스트 트레이더에

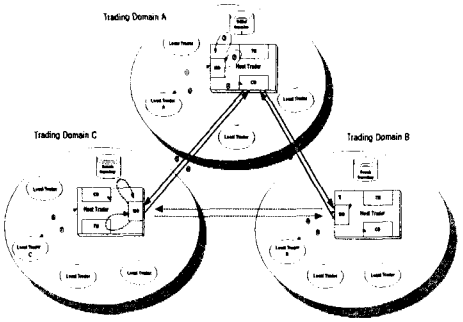

그림 8. 연합 서비스 수행

게 Broadcast방법으로 트레이딩 서비스를 위한 오퍼레이션을 전달하여 다른 도메인 상에 존재하는 로컬 트레이더와도 호스트 트레이더를 통하여 서비스를 수행한다.

III. 결 론

분산 컴퓨팅 환경의 분산 시스템을 구성하는 시스템들과 네트워크 환경에서 기존의 서비스 수행은 트레이더가 처리해야 할 정보가 지속적으로 증가하게 되어 정보의 양에 대한 문제와 그것을 최선의 버전으로 유지하는 문제로 인해 효과적인 서비스 제공이 불가능했다.

따라서 본 논문에서는 객체들간의 상호작용에 따른 문제점을 해결하기 위하여 서로 다른 영역에 위치한 독립적인 트레이더들을 하나의 호스트 트레이더와 여러 개의 지역 트레이더로 묶어 하나의 트레이딩 도메인을 형성하였다. 트레이딩 도메인내의 임의의 로컬트레이더가 호스트 트레이더에게 서비스를 요구하여 그 결과를 얻을 수 있고 호스트 트레이더는 도메인 레포지토리에 등록된 로컬트레이더의 정보를 이용하여 로컬트레이더의 서비스 요구를 대부분 자신의 트레이딩 도메인내에서 해결할 수 있도록 하였다. 이 때 로컬트레이더의 요구에 대한 결과를 줄 수 없다면 자신의 도메인 레포지토리 객체에 등록된 다른 도메인 상의 호스트 트레이더들에 대한 정보를 검색하여 다른 도메인 상의 호스트 트레이더와 상호협력을 할 수 있다. 또한, 서로 다른 영역에 위치한 독립적인 트레이더들의 서비스 타입, 서비스들의 수와 서비스 요청 빈도수에 따라 협력모델을 단순협력, 단순교섭, 그리고 연합으로 분류하여 호스트 트레이더간에 협력이 이루어 지도록 하므로서 분산 시스템을 이루는 요소 객체들의 접속을 효과적으로 할 수 있으며, 효율적으로 다량의 트레이딩 정보를 처리하여 서비스 응답시간이 단축되고 트레이더가 공포하는 서비스 오퍼는 중복없이 저장될 수 있고 클라이언트에게 보다 나은 서비스 속도와 서비스 선택의 폭을 넓힐 수 있게 하였다.

앞으로의 연구는 트레이딩 서비스에서 비용과 보완에 관한 문제를 고려하여 지속적으로 연구되어져야 한다.

참 고 문 헌

[1] OMG,"CORBA 2.0 Specification"

[2] DEC, HP, et al., "The Common Object Request Broker : Architecture and Specification", Revision 1.1, 91.12.1

[3] C. Burger, "Cooperation policies for trader", 1994

[4] Y. Ni and A. Goscinski, "Trader cooperation to enable object sharing among users of homoqenous", 1994

[5] M. Bearman and K. Raymond. Federating Trader : An ODP Adventure. International IFIP Workshop on Open Distributed Proceccing, 1991

[6] OMG, "OMG RFP5 Submission : Trading Object Service(Version 1.0)", May, 1996

[7] 이재승, 주용희, 홍원기, "CORBA를 이용한 ODP트레이딩 서비스", 1996

[8] 윤인숙, "이질적인 분산시스템에서 객체공유를 위한 협력모델", 석사학위논문, 1999