분산 트레이딩 서비스의 성능 향상에 관한 연구

○
송병권* 진명숙** 김건웅***

**서경대학교 정보통신학과 *경인여자대학 멀티미디어정보전산학부 ***목포해양대학교 해양전자통신공학부
e-mail:jinms@dove.kyungin-c.ac.kr

A Study on the Performance Improvement of Distributed Trading Service

Byung-Kwen Song* Myung-Sook Jin** Geonung Kim***

*Dept of Information and Communication Engineering, Seo-Kyeong University
**School of Multimedia Information Computing, Kyung-In Womens College
***Faculty of Electronic and Communication Engineering, Mok-Po National Maritime University

요 약

본 논문은 분산 환경에서 서버의 위치 투명성을 지원하는 트레이딩 서비스의 성능 향상에 대한 연구이다. 성능 향상을 위해 단일 트레이딩 영역에서는 트레이딩 정보의 집합인 컨텍스트의 구조를 활용하여 트레이딩 정보를 분산시키는 방안을 제시한다. 또한 글로벌 영역에서는 서로 연합(federation)된 각 트레이딩 영역간의 트레이딩 정보의 복제를 통해서 트레이딩 서비스의 성능 향상을 위한 방안을 제시한다.

1. 서론

컴퓨터와 유·무선 통신망의 보급 및 발달은 서로 상이한 분산 처리 환경(DPE: Distributed Processing Environment)을 구성하며 응용 서비스들 간의 성능, 확장성, 호환성, 통신방법의 이질성 등의 문제를 야기한다. 이러한 문제의 해결을 위해 각 표준화 기관에서는 개방 분산 처리 하부 구조(infrastructure)를 제시하고 있다. 또한 학계와 산업계에서는 이에 부합하는 플랫폼의 개발과 응용 서비스를 지원할 기반(하부) 서비스에 대한 연구가 함께 진행되고 있다. 이러한 기반 서비스 중의 한 가지로 주목을 받고 있는 것이 트레이딩(trading) 서비스이다.

트레이딩 서비스는 클라이언트가 원하는 서비스를 수행하는 서버의 주소, 전달 파라메타 등 서비스에 관한 사전 지식이 없더라도 해당 서비스를 제공하는 가장 적절한 서버에 대한 정보를 제공해 준다. 이 때 트레이딩 서비스를 제공하는 서버가 트레이더(trader)이다. 응용 서비스를 제공하는 서버는 자신의 정보를 트레이더에 등록(export)해야 하며, 클라이언트는 서비스를 사용하기 전에 서버에 대한 정보를 트레이더를 통해 알아낸다(import). 트레이더는 이러한 단순한 익스포트, 임포트 과정의 지원뿐만 아니라 타 트레이더들과 연합(federation)을 결성해 서비스에 대한 정보를 교환할 수도 있다.

분산 환경에서 차지하는 트레이딩 서비스의 비중으로 인해, 트레이더에 대한 연구는 비교적 활발히 추진되고 있다. OMG 등 표준화 활동에서 기본 트레이딩 서비스의 모델링과 시스템 구조의 제시하고 있으며 이를 바탕으로 연구기관들에서 CORBA, X.500 디렉토리 시스템을 활용한 구현 방법들을 제시하고 있다. 하지만 대부분의 트레이더 관련 연구가 기능의 제공에 중점을 두고 있으며 성능에 대한 고려가 이루어지지 못하고 있다. 현재까지 진행된 트레이더의 성능 향상에 관한 연구는 트레이딩 정보인 서비스 오퍼(offer)의 증가에 따라 단일 트레이딩 영역(trading domain)을 단순히 여러 개의 트레이딩 영역으로 분할하고 분할된 트레이더 사이의 연합(federation)에 따른 기본 기능 제공에 관한 것이 대부분이고[9], 연합을 통한 효율적인 트레이딩 영역의 활용 방안이나 성능향상에 대한 연구는 대단히 미흡한 실정이다.

본 논문에서는 트레이딩 정보의 증가에 대비하여 단일 트레이딩 영역에서는 트레이딩 정보를 해당 영역 내에서 분산시키는 분산 트레이더와 글로벌 트레이딩 영역에서는 영역간의 트레이딩 정보의 복제를 통해서 트레이더의 성능 향상을 위한 방안을 제시한다.

본 논문은 2 절에서 관련 연구로 트레이더에 대한 연구를 살펴보고 3 절에서 분산 트레이딩 서비스의 성능향상 방안을 기술하며 4 절에서 결론을 맺는다.

2. 관련 연구

트레이더에 대한 표준화기관의 연구로는, ODP[9], OMG[12], TINA-C[13], ANSA[14] 트레이더들이 있다.

이들은 주로 표준 모델과 사양의 제시에 중점을 두고 있으며 분산 환경의 확장에 대한 대책으로 트레이딩 영역(domain)을 분할하는 방안을 제시하고 있다. 또한 트레이딩 정보의 증가에 대비하여, 트레이딩 정보를 그 특성에 따라 그룹으로 분류하고 계층화하여 관리하는 트리(tree) 형태의 컨텍스트(context) 구조를 제시하고 있다. 이들은 대부분 개방 환경을 위한 모델 및 사양의 제시가 궁극적인 목표이므로, 분산 환경의 확대와 트레이딩 정보의 증가에 대비한 컨텍스트나 영역분할 방안을 제시하고 있지만, 구체적인 성능 향상에 대한 고려가 이루어지지 못하고 있다.

트레이더 구현에 관한 연구는 학교와 연구소를 중심으로 수행되고 있다. TBRMS(Trader-Based Resource Management System)[15]와 오스트레일리아 Sydney 대학[3, 4]에서 연구 중인 시스템은 X.500 디렉토리를 활용하였고 DRYAD[11], RHODOS[2] 등은 전용 데이터베이스를 사용하여 트레이딩 서비스의 기반 기능을 구현하였다. 이들 역시 트레이더의 기반 기능의 제공을 목표로 수행된 것이 대부분으로 국제 표준안이 나오기 전의 드래프트에 기반을 두고, 기본 트레이더 기능의 제공과 분산 환경의 이질성 극복을 위한 방안의 제시를 주요 목표로 하고 있다[1,3,5,7,8].

관련 연구를 통해 살펴본 바와 같이 현재 트레이더 관련 기술은 표준화된 사양의 제시와 기반 기능의 구현 단계이다. 따라서 분산 서비스의 활성화에 대비한 트레이딩 서비스의 성능 면에서의 고려나 트레이딩 서비스의 질적인 향상에 대한 것은 반영하지 못하고 있는 실정이다.

3. 트레이딩 서비스의 성능 향상 방안

본 논문에서는 서비스 오퍼의 증가에 따라 하나의 트레이딩 영역을 단순히 여러 개의 트레이딩 영역으로 분할하는 방안 대신에 단일 트레이더로서의 인터페이스를 유지하면서 로컬 트레이딩 영역 내의 트레이딩 정보를 분산시키는 컨텍스트 모델을 제안한다. 또한 트레이딩 영역간의 트레이딩 정보의 복제를 통한 성능 분석 모델을 제시하고 제시된 모델의 시뮬레이션을 통하여 분산 트레이더의 성능 최적화 방안을 제시한다.

3.1 로컬 트레이딩 영역에서의 성능 향상 방안

단일 트레이딩 영역에서는 서비스 오퍼의 저장, 검색 및 갱신 방법에 따라 트레이더의 성능이 결정된다. 본 논문의 선행연구로, 기존의 X.500 디렉토리나 데이터베이스를 이용한 트레이더의 단점을 극복하고, 정보의 양이 증가에 대비해 객체 기법을 바탕으로 분산 트레이더의 저장구조를 제시한 바 있다[16].

단일 트레이딩 영역에서 하나의 트레이더가 처리해야할 정보의 양이 많아지게 되면 트레이더는 일정한 규칙으로 트레이딩 정보인 서비스 오퍼를 분류하여 저장함으로써 효율적으로 관리하며 신속하게 검색할 수 있어야 한다. 이러한 이유로 도입하는 것이 컨텍스트(context)이다. 따라서 논리적이고 합리적인 구조의 컨텍스트가 도입되지 못하면 서비스 오퍼들은 평면적인 구조를 가지며 오퍼들을 저장하는 저장 체계에 전적으로 의존하게된다.

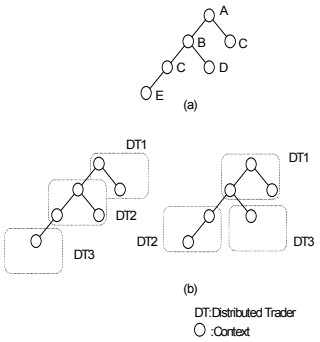


그림 1 컨텍스트 구조와 트레이더와의 대응 관계

본 논문에서는 서비스 오퍼 공간을 관리하기 위해 서비스 오퍼를 그룹지어 그림 1과 같은 트레이딩 정보 그래프 형식(정보관점:information viewpoint)으로 나타낸다. 트레이딩 정보 그래프의 각 노드는 같은 타입을 가지는 서비스 오퍼의 그룹으로 하나의 컨텍스트로 본다. 따라서 컨텍스트는 서비스 오퍼의 집합과 컨텍스트 고유의 속성들로 구성된다. 하나 이상의 컨텍스트는 계층 구조를 갖는 트리 구조로 이루어진다. 기존 연구[9]에서는 하나의 트레이더를 하나의 논리적 트레이더 객체(계산관점:computational view point)로 보지만 본 논문에서는 그림 1의 b와 같이 하나의 컨텍스트가 하나의 분산 트레이더에 대응할 수도 있으며 여러 개의 컨텍스트가 하나의 분산 트레이더와 대응될 수 있도록 하였다. 따라서 컨텍스트를 분산 및 관리의 단위로 하여 각 분산 트레이더에 임의로 할당될 수 있도록 하였다.

트레이딩 영역 내의 분산 트레이더는 하나의(통합된) 서비스 사용을 위한 인터페이스를 가진다. 본 논문에서는 각 컨텍스트가 일반적인 하나의 계산 객체 형태로 나타나지 않을 수 있으므로 컨텍스트의 구성 관리를 위한 별도의 인터페이스가 필요하다.

따라서 기본 트레이딩 서비스 인터페이스에 추가하여 그림 2와 같은 컨텍스트 자체 관리를 위한 인터페이스(ctxMgmtIf)와 컨텍스트 내의 서비스 오퍼 관리를 위한 인터페이스(ctxUserIf), 타입 관리를 위한 인터페이스(typeManagerIf)를 제안한다. 그림 2는 이들을 OMG IDL(Interface Definition Language)을 사용하여 간략히 표기

한 것이다.

```
interface ctxMgmtIf{ //context Mgmt. Interface
void ctxCreate(    //create context
    in IdType clientId,
    in CtxNameType ctxName,
    in PropertyListType ctxPropValues
)raises(OP_FAILURE)
void ctxDelete(...)  //delete context
void ctxRemove(...)    //create context
void ctxRename(...)       //rename context
void ctxDescribe(...)    //describe context
void ctxList(...)  //list context
void ctxModifyProp(...)  //modify context property
void ctxDescribeProp(...)  //describe context property
void ctxListProp(...)  //list context property
void ctxDeleteProp(...)  //delete context property
void ctxListSvcOffers(...) //list service offer
}  //ctxMgmtIf
```

(a) 컨텍스트 자체 관리 인터페이스

```
interface ctxUserIf{ //context user interface
void addSvcOffer(    //add service offer
    in IdType clientId,
    in CtxNameType ctxName,
    in ServiceOfferType svcOffer
    out IdType svcOfferId
)raises(OP_FAILURE)
void deleteSvcOffer(...)  //delete service offer
void nodifySvcOffer(...)  //modify service offer
void serarchSvcOffer(...)  //search service offer
}  //ctxUserIf
```

(b) 서비스 오퍼 관리 인터페이스

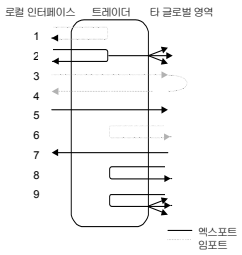
```
interface typeMgmtIf{ //type Management Interface
void typeRegister(    //register type
    in IdType clientId,
    in ServiceDescType svcDesc,
    out IdType svcDescId
)raises(OP_FAILURE)
void typeDelete(...)  //delete type
void typeList(...)  //list type
void typeCheck(...)  //check type
}  //typeManagerIf
```

(c) 타입 관리 인터페이스

그림 2 컨텍스트 관리 인터페이스

3.2 트레이딩 정보 복제를 통한 연합 트레이더 간의 성능 향상 방안

분산 트레이더로 구성된 분산 환경에서 트레이더 영역간의 트레이더의 성능 향상 방안으로, 자연 발생적인 트레이딩 영역의 분할 외에 컨텍스트 구조를 반영한 서비스 정보의 유형과 수용 정책 및 서비스 요구량, 서비스 처리율, 엑스포트와 임포트 비율 등 서비스 요구 특성을 고려한 트레이딩 영역 분할 정책으로 분할된 트레이딩 영역간에 트레이딩 정보인 서비스 오퍼(service offer)의 복제를 통해 트레이딩 서비스의 처리 속도를 개선할 수 있는 방안을 제시한다.



구분	요구 (source)	Import / Export	설명	비고
1	Local	I	로컬 Imp.요구를 처리 후 응답	
2		E	로컬 Exp.요구를 처리 후 응답	
3		I	원격 Imp. 요구를 처리 후 원격지로 요청	
4	Global	I (Res.)	원격 Imp. 결과를 받고 응답	3의응답
5	Local	E	원격 Exp. 요구를 처리 후 원격지로 요청	
6	Global	I	원격 Imp. 요청을 처리 후 원격지로 응답	
7		E (Res.)	원격 Exp. 결과를 받고 응답	5의응답
8		E	원격 Exp.요구(최종)를 처리 후 원격지로 응답	복제를 지원하는 경우에만 적용
9		E	원격 Exp. 요구(중간)를 처리 후 다시 원격지로 요청 전달	

그림 3 트레이딩 복제에 따른 서비스 처리 시나리오

그림 3은 트레이딩 정보인 서비스 오퍼의 복제를 허용하는 트레이더와 그렇지 못한 트레이더에서의 임포트와 엑스포트 서비스의 처리 과정이다.

다음은 앞에서 정의한 트레이더의 처리 과정과 유도된 각종 시스템의 효율 계수를 사용하여 복제 메카니즘이 지원되지 않는 트레이더로만 구성된 시스템과 복제 및 비복제가 혼합된 시스템의 환경 하에서 서비스율, 서비스 요구확률, 복제를 지원하는 트레이더의 비율, 예상 지연시간 등 효율 계수의 값의 변화에 따른 전체 시스템의

성능분석을 통하여 트레이딩 영역분할을 위한 최적의 방안을 제시한다.

3.2.1 서비스율 변화

표 1 서비스율 변화에 따른 성능 분석

μ	대기 시간	k=0 → 29 Rt = 0	비고	k=0 → 29 Rt=2.0	비고
5	WTN WTC	0.6393 0.6393→0.9580	불변 증가	1.5893 1.5893→1.1163	불변 k>2 이후 감소 k>3 이후 WTN>WTC
6	WTN WTC	0.4815 0.4915→0.5205	불변 증가	1.4315 1.4315→0.6789	불변 k>2 이후 감소 k>3 이후 WTN>WTC
7	WTN WTC	0.3861 0.3861→0.3581	불변 k>6 이후 감소 k>11 이후 WTN>WTC	1.3361 1.3361→0.5164	불변 k>1 이후 감소 k>2 이후 WTN>WTC
8	WTN WTC	0.3223 0.3223→0.2731	불변 k>4 이후 감소 k>6 이후 WTN>WTC	1.2723 1.2723→0.4314	불변 k>1 이후 감소 k>2 이후 WTN>WTC
9	WTN WTC	0.2766 0.2766→0.2208	불변 k>3 이후 감소, k>5 이후 WTN>WTC	1.2266 1.2266→0.3791	불변 k>1 이후 감소 k>2 이후 WTN>WTC
10	WTN WTC	0.2422 0.2422→0.1854	불변 k>3 이후 감소, k>4 이후 WTN>WTC	1.1922 1.1922→0.3437	불변 k>1 이후 감소 k>2 이후 WTN>WTC
20	WTN WTC	0.1080 0.1080→0.0712	불변 k>2 이후 감소, k>3 이후 WTN>WTC	1.0580 1.0580→02296	불변 k>1 이후 감소 k>2 이후 WTN>WTC
30	WTN WTC	0.0695 0.0695→0.0441	불변 k>2 이후 감소, k>3 이후 WTN>WTC	1.0195 1.0195→0.2024	불변 k>1 이후 감소 k>2 이후 WTN>WTC
50	WTN WTC	0.0406 0.0406→0.0250	불변 감소	0.9906 0.9906→0.1834	불변 k>1 이후 감소 k>2 이후 WTN>WTC

Rt:네트워크 지연
WTN:비복제 시스템, WTC:복제 허용 시스템 대기 시간

효율 계수를 이용한 수학적인 성능분석을 통해, 임포트가 전체의 90퍼센트를 차지하고, 요청 정보가 글로벌과 로컬에 있을 확률이 각각 50퍼센트이며, 네트워크 지연을 고려하지 않을 경우 도착율에 비해 서비스 처리율(μ: 처리율/도착율)이 7배 미만이면 복제 메커니즘이 지원되는 트레이더의 비율(k)이 증가할수록 복제에 따른 오버헤드로 오히려 성능(대기시간:WT)이 저하되었다. 하지만 도착율에 비해 서비스 처리율이 7배 이상이면 복제 메커니즘이 지원되는 트레이더의 비율에 따라 시스템의 성능이 향상됨을 알 수 있다(표 1).

3.2.2 서비스 유형(요구전달, 임포트, 엑스포트 비율)별 서비스 율의 변화

임포트가 전체의 90퍼센트를 차지하고, 요청 정보가 글로벌과 로컬에 있을 확률이 각각 50퍼센트이며, 네트워크 지연을 고려하지 않을 경우 서비스 유형(요구의 전달, 임포트, 엑스포트 비율)별 서비스 율을 달리하여도 복제 메카니즘이 지원되는 트레이더가 복제에 따른 오버헤드로 인하여 오히려 성능이 저하되었다. 그러나 네트워크 지연(Rt)을 고려할 경우, 복제가 지원되는 트레이더의 비율을 7 퍼센트 이상 유지하면 복제 메커니즘이 지원되는 시스템의 성능이 향상됨을 알 수 있다(표 2).

표 2 서비스 유형별 서비스율 차등 부여에 따른 성능 분석

비율	대기 시간	k=0 → 29 Rt = 0	비고	k=0 → 29 Rt =2.0	비고
1:1:1	WTN WTC	0.2422 0.2422→0.1854	불변 k>3 이후 감소, k>4 이후WTN>WTC	1.1922 1.1922→0.3437	불변 k>1 이후 감소, k>2 이후WTN>WTC
3:2:1	WTN WTC	0.0970 0.0970→0.1297	불변 감소	1.0470 1.0470→0.2880	불변 k>1 이후 감소, k>2 이후WTN>WTC
4:3:1	WTN WTC	0.0659 0.0659→0.1009	불변 감소	1.0159 1.0159→0.2592	불변 k>1 이후 감소, k>2 이후WTN>WTC
6:4:1	WTN WTC	0.0468 0.0468→0.0822	불변 감소	0.9968 0.9968→0.2405	불변 k>1 이후 감소, k>2 이후WTN>WTC
7:5:1	WTN WTC	0.0381 0.0381→0.0696	불변 감소	0.9881 0.9881→0.2280	불변 k>1 이후 감소, k>2 이후WTN>WTC

(비율:서비스율)μp(서비스 전달):μr(임포트):μw(엑스포트)

3.2.3 임포트/엑스포트 비율의 변화

요청 정보가 글로벌과 로컬에 있을 확률이 각각 50퍼센트이며, 네트워크 지연을 고려하지 않을 경우, 임포트의 요구가 70퍼센트 이하이면, 복제 메커니즘이 지원되는 트레이더가 복제에 따른 오버헤드로 인하여 오히려 성능이 저하되었다. 하지만 임포트의 요구가 80 퍼센트 이상 유지되면 복제 메커니즘이 지원되는 시스템의 성능이 향상된다. 표 3은 임포트 요구 변화율(Plr)을 감소시키고, 엑스포트 요구율(Plw)을 증가시키면서 대기시간을 계산한 것이다.

표 3 임포트/엑스포트 요구 확률 변화에 따른 성능 분석

P1r	P1w	대기 시간	k=0 → 29 Rt=0	비고	k=0 → 29 Rt =2.0	비고
0.45	0.05	WTN WTC	0.1080 0.1080→0.712	불변 k>2 이후 감소, k>3 이후 WTN>WTC	1.0580 1.0580→0.2296	불변 k>1 이후 감소, k>2 이후WTN>WTC
0.40	0.10	WTN WTC	0.1050 0.1050→0.0923	불변 k>6 이후 감소, k>11 이후 WTN>WTC	1.0050 1.0050→0.3423	불변 k>2 이후 감소, k>4 이후WTN>WTC
0.35	0.15	WTN WTC	0.1019 0.1019→0.1245	불변 감소	0.9519 0.9519→0.4662	불변 k>4 이후 감소, k>7 이후WTN>WTC
0.30	0.20	WTN WTC	0.0989 0.0989→0.1800	불변 감소	0.8989 0.8989→0.6133	불변 k>6 이후 감소, k>11 이후WTN>WTC
0.25	0.25	WTN WTC	0.0959 0.0959→0.1800	불변 감소	0.8459 0.8459→0.8231	불변 k>10 이후 감소, k>20 이후WTN>WTC
0.20	0.30	WTN WTC	0.0929 0.0929→0.7201	불변 감소	0.7929 0.7929→1.3367	불변 감소
0.15	0.35	WTN WTC	0.0899 0.0899→1.0821	불변 감소	0.7399 0.7399→1.8113	불변 감소
0.10	0.40	WTN WTC	0.0870 0.0870→1.1492	불변 감소	0.6870 0.6870→1.9396	불변 감소
0.05	0.45	WTN WTC	0.0840 0.0840→1.6551	불변 감소	0.6340 0.6340→2.4719	불변 감소

3.2.4 서비스 오퍼의 로컬, 글로벌 트레이더 존재 비율

임포트가 전체의 90퍼센트를 차지하고, 요청 정보가 글로벌과 로컬에 있을 확률(Plr:Prr)이 변화할 경우, 네트워크 지연을 고려하지 않으면 글로벌 트레이더에서 처리될 확률이 78퍼센트 이하일 때 복제 트레이더의 비율에 따라 성능이 향상됨을 알 수 있다(표 4). 그러나 글로벌 트레이더에서 처리되는 확률이 이보다 높아지면 성능이 저하된다.

표 4 요구된 정보가 있을 확률 변화에 따른 성능 분석

P1r	Prr	대기 시간	k=0→29 Rt = 0	비고	k=0 → 29	비고
0.1	0.8	WTN WTC	0.1553 0.1553→0.07744	불변 감소, k>1 이후 WTN>WTC	1.8442 1.8442→0.2807	불변 k>1 이후 감소, WTN>WTC
0.2	0.7	WTN WTC	0.1414 0.1414→0.0734	불변 k>1 이후 감소, WTN>WTC	1.6192 1.6192→0.2660	불변 k>1 이후 감소, WTN>WTC
0.3	0.6	WTN WTC	0.1278 0.1278→0.0726	불변 k>1 이후 감소, k>2 이후 WTN>WTC	1.3945 1.3945→0.2514	불변 k>1 이후 감소, WTN>WTC

지금까지 트레이딩 서비스의 각 효율 계수의 변화에 따른 성능을 분석하였으며 이를 통해 최적의 트레이딩 서비스 환경을 구축할 수 있을 것이다.

5. 결론

본 논문은 트레이딩 서비스에서 트레이딩 정보의 증가에 대비하여 단일 트레이딩 영역에서는 트레이딩 정보를 해당 영역 내에서 분산시키는 분산 트레이더와 글로벌 트레이딩 영역에서는 영역간의 트레이딩 정보의 복제를 통해서 트레이더의 성능 향상을 위한 방안을 제시한다.

본 논문에서는 제안된 트레이딩 영역분할 모델의 성능분석을 위해 수학적 성능분석 모델을 설정하여 다양한 환경 하에서 시스템 효율계수의 변화에 따른 성능분석을 실시하였다. 그 결과 복제 메커니즘이 지원되지 않는 트레이더로만 구성된 시스템과 복제 및 비복제 시스템이 혼합된 환경 하에서의 서비스율, 복제를 지원하는 트레이더의 비율, 예상 지연시간 등 효율 계수의 값의 변화에 따른 전체 시스템의 성능분석을 통하여 트레이딩 영역분할을 위한 최적의 방안을 제시하였으며 성능분석 결과 분산 환경에서 복제 메커니즘이 지원되는 트레이더를 주어진 상황에 따라 적절한 비율로 혼합 구성하였을 경우, 최적의 성능을 가져옴을 알 수 있었다. 본 연구의 결과로 도출된 시스템의 유형에 따르는 평균 대기 시간을 활용할 경우, 분산 환경에서 트레이더간 연합 시 최적의 트레이딩 영역분할을 통한 전체 시스템의 성능 향상을 기대할 수 있다.

참고문헌

[1] Andew Waugh and Mirion Bearman, "Designing an ODP Trader Implementation using X.500", Proceedings of the International Conference on Open Distributed Processing, Brisbane, Australia, February 1995

[2] Andrzej Goscinski and Ying Ni, "Object Trading in Open Systems", Proceedings of the IFIP TC6/WG6.1 International Conference on Open Distributed Processing, Berlin, Germany, pp. 145~156, September 1993

[3] Ariella Richman and Doan Hoang, "Accomplishing Distributed Traders Utilizing the X.500 Directory", due to appear in MICC'95 2nd IEEE Malaysia International Conference on Communications, Malaysia, Nov 1995

[4] Ariella Richman and Doan Hoang, "Trader Interoperability: Why Two Models Are Better Than One", due to appear in ISCIS-X The Tenth International Symposium on Computer and Information Sciences, Kusadasi, Turkey, Nov 1995

[5] Claudia Popien and Bernd Meyer, "Federating ODP Traders: An X.500 Approach", Proceedings of the International Conference on Communications 1993 (ICC'93), pp. 313 - 317, Geneva, Switzerland,

[6] Cora Burger, "Cooperation Policies for Traders", Proceedings of the International Conference on Open Distributed Processing, Brisbane, Australia, February 1995,

[7] Dudet, "X.500 Directory for those familiar with Trading Concepts", SEPT, Report Number 358.90 SEPT/SCE/SPM/MD - V2, 14th December 1990.

[8] Gunnar Almgren and Magnus Anderson, "Open System Trader using X.500", Telia Research Report, Report Number Televerket/RC 02.01, 1 September 1991.

[9] ISO/IEC JTC1/SC21,"Draft Rec. X.950-1|ISO/IEC 13235-1-ODP Trading Function : Specification, 2nd DIS", 23 January 1997

[11] Lea Kutvonen, "Federation Transparency in ODP Trading Function", Techical Report, Report Number C-1994-32, Department of Computer Science, University of Helsinki, May 1994.

[12] OMG, "Trading Object Service Version 1.0.0", OMG Doc. 95-05-06, May 1996

[13] Hwang et al, "TINA DPE Service Specifications", TINA-C Doc. TR_HW.001_1.0_94, Dec 1994

[14] ANSA, " ANSAware 4.1 Application Programming in ANSAware", Doc. RM.102.02, Feb 1993

[15] J. W. Hong, M. A. Bauer, and J. M. Bennett. Integration of the Directory Service in the Network Management Framework. Proc. of the Third International Symposium on Integrated Network Management, San Francisco CA, April 1993.

[16] 송병권, 진명숙, "LOM:분산 트레이더를 지원하는 경량(lightweight) 객체 모델," 정보처리학회추계학술, 99.10