CORBA-ORB, JAVA-RMI, 소켓을 이용한 그룹 통신의 구현 및 성능 분석

Implementation and Performance Analysis of the Group Communication Using CORBA-ORB, JAVA-RMI and Socket

한 윤 기* 구 용 완**
Yun-Ki Han Yong-Wan Koo

요 약

대다수의 인터넷 기반의 분산 어플리케이션이나 클라이언트/서버의 응용은 부하균등, 통신 지연, 네트워크 결함 등의 문제점을 처리하여 사용자에게 서비스해야 한다. 또한 화상 회의, VOD, 병행 소프트웨어 공학과 같은 정교한 응용프로그램들은 추상적인 그룹 통신을 필요로 한다. 이러한 패러다임들을 현재의 CORBA 버전들은 적절히 수용하지 못한다. CORBA는 주로 Point-to-Point 통신을 하기 때문에 분산 시스템에서 예측 행위를 하는 신뢰성있는 응용 기술에 대한 구현은 지원하지 않는다. 따라서, 본 논문에는 분산 컴퓨팅 환경 하에서 CORBA-ORB를 이용한 그룹 통신, JAVA-RMI를 이용한 그룹 통신, 소켓을 이용한 그룹 통신 등을 설계 및 구현을 하였으며, 이에 따른 성능 분석을 실시하였다. 성능 분석은 객체의 증가에 따른 지연시간으로 측정하였고, CORBA의 ORB를 이용한 그룹 통신의 경우 평균은 14.5172msec, JAVA의 RMI를 이용한 그룹 통신의 경우 평균은 21.4085msec, 소켓을 이용한 그룹 통신의 경우 평균은 18.0714msec가 나왔다. 멀티캐스트와 UDP를 이용한 그룹 통신은 각각 0.2735msec, 0.2157msec로 측정되었음을 알 수 있다. 논문의 결과로 객체의 증가에 따라 CORBA-ORB 그룹 통신의 성능향상을 보였다. 본 연구는 결함 허용 클라이언트/서버 시스템, 그룹웨어, 텍스트 검색엔진, 금융 정보 시스템 등에 적용 가능하다.

Abstract

Large-scale distributed applications based on Internet and client/server applications have to deal with series of problems. Load balancing, unpredictable communication delays, and networking failures can be the example of the series of problems. Therefore, sophisticated applications such as teleconferencing, video-on-demand, and concurrent software engineering require an abstracted group communication. CORBA does not address these paradigms adequately. It mainly deals with point-to-point communication and does not support the development of reliable applications that include predictable behavior in distributed systems. In this paper, we present our design, implementation and performance analysis of the group communication using the CORBA-ORB, JAVA-RMI, and Socket based on distributed computing. Performance analysis will be estimated latency-time according to object increment, in case of group communication using ORB of CORBA the average is 14.5172msec, in case of group communication using RMI of Java the average is 21.4085msec, in case of group communication using socket the average is becoming 18.0714msec. Each group communication using multicast and UDP can be estimated 0.2735msec and 0.2157msec. The performance of the CORBA-ORB group communication is increased because of the increased object by the result of this reseach. This study can be applied to the fault-tolerant client/server system, group-ware, text retrieval system, and financial information systems.

1. 서 론

최근 사용자들은 기존의 정적인 미디어에서 연속적인 스트림 처리 등의 서비스 이용 욕구가 증대하고 있다. 이러한 요구는 분산 처리 기술과 인터넷의 발달로 인해, 방대한 데이터 처리가 가능해진 것이 사실이다. 그러나, 이러한 최근의 환경에서는 객체처리의 복잡성과 관리의 어려움을 초래하고 있다[1,2]. 따라서 이 기종 간의 하드웨어 플랫폼 혹은 소프트웨어 플랫폼에 적합한 분산 컴

* 정 회 원 : 수원대학교 컴퓨터학과
hyksea@hanmail.net
** 종신회원 : 수원대학교 컴퓨터학과 교수
ywkoo@mail.suwon.ac.kr

퓨팅 환경으로 바뀌어가고 있다[3,12]. 이러한 분산 환경에서 응용들을 효율적으로 처리하기 위해서는 객체 그룹화가 필요하며[4,6,9,10], 방대한 연속적인 스트림 처리를 하기 위해서는 그룹 통신이 필요하다. 따라서, 본 논문에서는 그룹 통신 기법들을 각각 설계 구현하였으며, 이에 따른 성능 분석을 하였다. 또한 SPSS를 이용한 분석을 실시하였다.

본 논문의 2장에서는 다수의 사용자 요구에 따른 서비스에 기본이 되는 그룹 통신에 대하여 살펴보고, 3장에서는 CORBA(Common Object Request Broker Architecture)의 ORB(Object Request Broker)를 이용한 그룹 통신, 기존의 RPC(Remote Procedure call)을 이용한 자바의 RMI(Remote Method Invocation) 그룹 통신, 고전적인 소켓(socket) 그룹 통신을 설계 구현한다. 또한, 그룹 통신을 할 때 데이터 전송시에 이용되는 멀티캐스트와 UDP를 이용한 그룹 통신을 설계 구현한다. 4장에서는 각 그룹 통신간의 성능 측정한 후, SPSS를 이용한 확률적 예측을 통하여 성능 분석을 기술하였다. 마지막으로 5장에서는 향후 연구 방향에 대해 언급하고, 본 논문에 대해 결론을 맺는다.

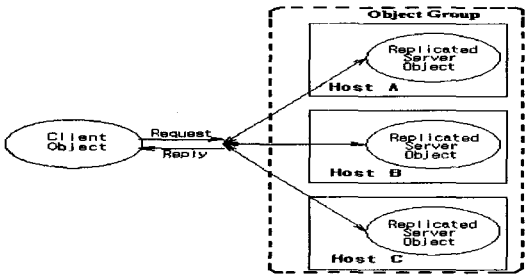
2. 기본 개념

본 장에서는 다수의 사용자들간의 상호 통신을 지원할 때 기본이 되는 그룹 통신에 대하여 살펴본다.

2.1 그룹 통신을 위한 요구 사항

그룹 통신이란 컴퓨터 네트워크와 분산처리 시스템 분야에서 메시지를 하나의 자원으로부터 목적지 그룹의 집합으로 전달하는 방법을 의미한다. 신뢰성 있는 그룹 통신을 지원하기 위해서 필요한 기능 항목을 기술하면 다음과 같다[4,5,7,8,11].

- 객체 그룹(object group)
- 결함 탐지(failure detection)
- 능동적 복제(active replication)
- 뷰(view)
- 요구 원자성(request atomicity)
- 상태 전송(state transfer)
- 객체 모니터링(object monitoring)
- 요청 순서화(request ordering)
- 가상 동기화(virtual synchrony)



(그림 1) 하나의 객체 그룹으로 구현된 서버

객체 그룹은 그룹 통신에 있어서 중요시된다. 요구의 원자성 문제나 객체 복제를 위한 기본 그룹으로 이용된다. 하나의 객체 그룹으로 구현된 서버의 구조는 그림 1과 같다.

객체들의 신뢰성은 각 객체들에게 추상화(abstraction)를 지원함으로써 높아지고 분산 객체 그룹을 관리한다. 추상화는 분산 시스템 내에 있는 다수의 복잡한 속성으로부터 프로그래머를 보호하며, 클라이언트는 신뢰성 있는 멀티캐스트로 객체 그룹과 통신을 한다. 객체 그룹 행위는 클라이언트에게 투명하다.

- 하나의 객체그룹과 바인드하기 위해 사용되는 어플리케이션 코드는 단일 객체와 바인드하는 것처럼 사용한다.
- 클라이언트는 마치 하나의 객체와 통신하는 것처럼 하나의 요구를 송신하고, 하나의 응답을 수신한다.
- 객체 그룹은 모든 신뢰성 있는 프로토콜을 자동적으로 처리하기 때문에 클라이언트에게는 투명하다.
- 실행시간에 결함이 발생하면 다중 서버중의 하

나만이 클라이언트에게 응답한다.

2.2 그룹 통신을 위한 객체 그룹 특성

분산 컴퓨팅 환경에서 서비스나 시스템은 다수의 노드에 분산되어 있는 객체들을 분산객체(Distributed Object)라 하며, 이러한 분산객체들로 구성된다. 그러나 분산된 객체들은 응용 프로그램 개발자들에게 복잡한 설계와 관리의 어려움을 주고 있다. 그래서 분산 객체들의 복잡한 설계와 관리의 어려움을 해결하기 위해 객체들을 하나의 집합 단위로 구성하는 객체 그룹(Object Group)에 대한 개념이 도입되어, 객체 그룹마다 관리를 할 수 있게 되었다. 객체 그룹은 분산된 객체들의 집합으로 이루어져 있으며, 그룹 내의 객체들을 관리하는 그룹 관리자(Group Manager)를 두어 그룹 단위로 관리 및 유지되고 있다. 객체 그룹은 외부의 다른 객체 집합과의 인터페이스를 위해 각각 관리적 측면에서 접근할 수 있는 srmfnq 관리자(Group Manager)를 두어 그룹 단위로 관리 및 유지되고 있다. 객체 그룹은 외부의 다른 객체 집합과의 인터페이스를 위해 각각 관리적 측면에서 접근할 수 있는 그룹 관리자(Group Manager)와 그룹을 생성하거나 삭제할 수 있는 객체 그룹(Object Group)과 객체 그룹 내의 메시지를 송수신할 수 있는 메시지 전송자(Message Sender)와 메시지 수신자(Message Receiver), 실제 처리를 담당하는 그룹 서비스(Group Service). 그리고 객체의 정보와 상태를 나타내기 위한 그룹 정보(Group Information)로 구성된다. 다음은 객체 그룹의 특성을 나타낸다.

- 객체 그룹은 캡슐화되어 구현된다.
- 객체 그룹은 분산객체의 집합으로 구성된다.
- 객체 그룹은 그룹내에 하나의 그룹 관리자를 포함한다.
- 객체 그룹의 객체들은 그룹 관리자를 통하여 그룹에 가입, 탈퇴한다.
- 객체 그룹 내의 각 객체들은 외부의 객체와 인터페이스를 위해 그룹 관리자를 호출한다.

2.3 객체 그룹 통신의 기본 구조

서버의 그룹 관리자는 클라이언트 객체로부터 접속이 시도되기 전에 대기하고 있어야 한다. 클라이언트 객체는 대기 중인 그룹 관리자에게 통신을 요청하게, 그룹 관리자는 자신의 서비스 객체인 객체 그룹을 호출하여 그룹을 생성한다. 객체 정보 저장소에 클라이언트의 정보를 저장한다. 객체 그룹 내의 메시지 전송자는 객체 정보 저장소를 참조하여 그룹 관리자를 통해 클라이언트 객체에게 그룹 정보를 전달한다. 그룹 정보가 모든 클라이언트 객체들에게 전송된 후, 클라이언트 객체들은 서버 측의 그룹 관리자를 통하여 다른 객체들과 통신 설정을 할 수 있게 된다.

3. 시스템 설계와 구현

3.1 연구 배경의 동기

대다수의 인터넷 기반의 분산 어플리케이션이나 클라이언트/서버의 응용은 부하균등, 통신 지연, 네트워크 결함 등의 문제점을 처리하여 사용자에게 서비스해야 한다. 이러한 페러다임들을 현재의 코바 버전들은 적절히 수용하지 못한다. 코바는 주로 Point-to-Point 통신을 하기 때문에 분산 시스템에서 예측 행위를 하는 신뢰성있는 응용 기술에 대한 구현은 지원하지 않는다. 따라서, 본 논문에는 분산 컴퓨팅 환경 하에서 코바 ORB를 이용한 그룹 통신, 자바 RMI를 이용한 그룹 통신, 소켓을 이용한 그룹 통신 등을 설계 및 구현을 하였으며, 이에 따른 성능 분석을 실시하였다. 본 장에서는 객체 그룹에 추가된 객체에 대해서 다른 객체의 상태 정보를 받아 새로운 객체 그룹에 전달함으로써 동일한 객체 그룹내의 모든 객체가 같은 상태를 유지하도록 원자성(atomicity)을 보장할 수 있는 그룹 통신을 코바의ORB, 자바의 RMI, 소

켓을 이용하여 각각 설계 및 구현하였다. 또한, 데이터 전송 시에 이용되는 멀티캐스트와 UDP를 이용한 그룹 통신을 설계 구현한다.

3.2 설계 및 구현에 이용된 요구순서화 (request ordering)

객체 그룹 외부로부터의 요구에 대해 전달 순서를 결정하여 유지함으로서 객체 그룹 내 객체간의 수신 요청의 순서를 보장한다. 그룹 통신 기법을 사용하는 대부분의 어플리케이션들은 요구 순서화를 보증해야 하며, 전체적 순서화(total ordering)와 인과적 순서화(causal ordering)가 있다. 전체적 순서화는 서로 다른 클라이언트가 그룹 내의 모든 멤버들에게 동일한 순서로 전달함으로서, 다양한 요구들이 병행적으로 객체 그룹에게 전송할 수 있게 한다. 실행시간이 메시지 순서를 결정하는 동안 모든 요구에게 지연을 요구한다. 본 논문에서는 인과적 순서화를 이용하여 코바의 ORB를 이용한 그룹 통신, 자바의 RMI를 이용한 그룹통신, 소켓을 이용한 그룹 통신을 각각 구현하였다. 인과적 순서화를 이용한 이유로는 요구의 우선순위에 잠정적으로 의존하게 되고 이전 요구가 객체에 먼저 전달되며 전체적 순서화 보다 적은 통신 오버헤드를 가진다. 인과적 순서화에 있어서 지연은 잠정적으로 원인적 관계가 있는 요구들만 지연시키고, 인과 관계가 없는 요구들은 즉시 전송한다는 장점이 있기 때문이다.

3.3 설계 및 구현에 따른 비교

3.3.1 코바의 ORB 기법

코바의 ORB는 기존에 존재하는 여러 가지 형태의 C/S 모델을 지원하는, 객체지향 기반의 코바 미들웨어의 소프트 버스이다. 즉, 코바 분산 환경의 인터페이스 통합에 대한 표준이다.

- 코바 표준 : 분산환경 트랜잭션 서비스와 같은 인프라 구조에 필수 요소들을 만족시키기 위한 다양한 서비스를 제공한다.

3.3.2 자바의 RMI 기법

- RMI : 서로 다른 런타임 환경에서 클라이언트가 원격의 객체를 로컬 객체처럼 자신의 런타임 환경에서 원하는 메소드를 호출할수 있도록 하는 자바의 분산 객체 표준이다. 단점으로는 서버측에 존재하는 객체를 클라이언트로 복사해서 가져온다는 것이다. 자바의 RMI는 해당 객체에 대한 원격 레퍼런스를 제공하고 클라이언트는 원격 객체에 대한 레퍼런스를 통해 메소드를 호출한다. 자바의 RMI에서 사용하는 RMP(Remote Method Protocol)는 자바 RMI에 의존적인 프로토콜이다.
- 코바와 RMI 비교 : 표 1은 CORBA와 RMI의 기능상의 비교를 나열한 것이다.

(표 1) 코바와 RMI의 기능 비교

기 능	코 바	RMI
언어독립성	Yes	No
프로토콜	IIOP/GIOP	RMP
Parameter marshaling	Yes	Yes
Parameter 전달	in/out/inout	in
인터페이스	IDL	자바 인터페이스
동적 객체 발견	인터페이스 레퍼지토리(IR)	No
동적 호출	DII(Dynamic Interface Invocation)	No
콜백	Yes	Yes
보안	Security Service SSL(Object Transaction Service)	SSL(1.2)
트랜잭션	OTS(Object Transaction Service)	No
지원 서비스	Life Cycle Persistence Concurrency Control, Naming RelationShip Query, Licensing, Properties, Time, Trader, Collection, Externalization	JTS(CORBA OTS) Registry Interface Remote Object Activation

3.3.3 고전적인 소켓 기법

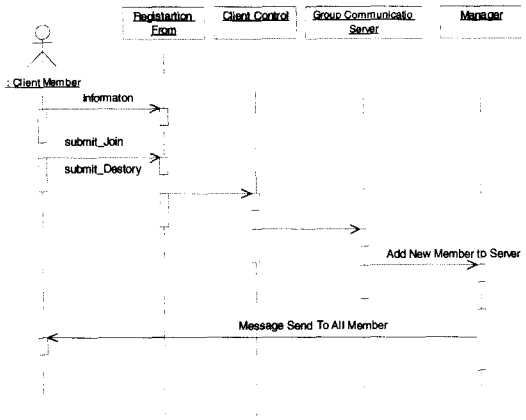
- 소켓 : 원래 BSD 소켓에서는 소켓을 개설할 때 socket() 함수의 인자로 TCP와 UDP 프로토콜을 구분한다. 또한 서버와 클라이언트용 소켓이 미리 구분되지 않고 소켓에 어떤 함수들을 호출하는가에 따라 listen()나 connect() 기능이 구분한다. 본 논문에서는 코바의 ORB, 자바의 RMI, 소켓의 그룹 통신을 자바 버전으로 구현하였다.

3.4 시스템 설계 및 구현

본 장에서는 각 그룹 통신에서의 설계 방법 및 구현을 언급한다.

3.4.1 코바의 ORB를 이용한 그룹 통신

이용된 시스템은 JDK와 Jbuilder를 사용하였다. 시스템 분석 도구로는 Rational Rose를 사용하였고, 다변량 확률 분석을 위하여 SPSS 통계 패키지를 이용하였다. 코바의 ORB를 이용한 그룹 통신은 네이밍 서비스(naming service)를 이용하여 객체를 참조한다. 그룹에 참여한 사용자, 즉 클라이언트 프로세스가 투명하게 서비스 받아야 하기 때문에 제공받을 수 있는 서비스로는 그룹 생성, 그룹 참여, 그룹 탈퇴, 다중 메시지 전송 등을 제공한다. 이때 시스템 고장에 대해서는 한 클라이언트가 고장난 그룹 멤버들과 클라이언트들로 부터 응답을 무한정 기다리는 것을 방지하며, 런타임시 활동중인 그룹 멤버들은 통신에 방해 없이 자동적으로 고장 탐지를 한다. 응답이 없는 그룹 멤버들은 타임 오버 결함 탐지기를 이용하여 응답 시간을 체크한 후 응답이 없는 멤버들을 고장이라고 선언한다. 고장을 탐지하고 고장통보를 전달함으로써 실행 시간에는 객체 그룹 멤버들과 투명하게 일관성을 유지하고, 멀티캐스트 요구 완료를 보장한다. 그림 2는 Sequence Diagram으로 프로세스 처리 과정을 표현하였다. 표 2는 코바의



(그림 2) 코바의 ORB 그룹 통신 Sequence Diagram

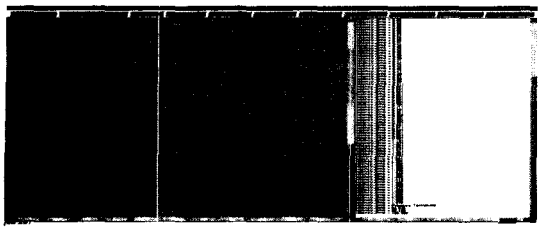
```
module GroupCommunication
  {
 interface groupClient {
   boolean sendMsg( in wstring msg );
   boolean setClientNum( in wstring num );
   };
 interface groupServer{
   boolean broadcast( in wstring msg );
   boolean register( in CClient clientObjRef ) ;
   boolean deregister( in CClient clientObjRef , in
   wstring cnum ) ;
   };      };
```

(표 2) 코바의 ORB 그룹 통신을 위한 IDL 명세

ORB 그룹 통신에 대한 IDL 명세이다.

구현에 있어서 서버 측에서는 그룹에 포함된 멤버간의 메시지 전송을 모니터링한다. 이때 원자성 즉, 한 멤버가 메시지를 수신하면 다른 멤버도 동등한 메시지를 수신해야 올바른 그룹 통신 전송을 할 수 있다. 또한, 한 멤버가 메시지를 받지 못하면 마찬가지로, 다른 멤버도 메시지를 받아서는 안 된다. 본 논문에서는 인과적 순서화를 이용하였다.

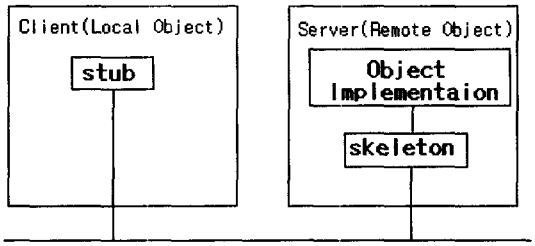
그림 3은 코바의 ORB 그룹 통신 구현을 실행한 화면이다. 이때 그룹에 접속하여, 모든 멤버에게 메시지를 전송을 한다. 성능분석을 하기 위하여, 예측에 사용하는 변수, 설명변수를 추출하기 위한 프로그램의 구현에 포함하여 나타냈다.



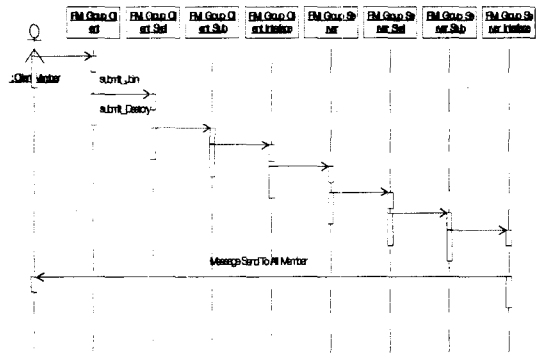
(그림 3) 코바의 ORB 그룹 통신 구현

3.4.2 자바의 RMI를 이용한 그룹 통신

RMI의 분산 객체의 정의는 인터페이스 정의와 구현으로 명시되며, 그림 4에서 처럼 분산 객체 인터페이스와 구현 객체를 클라이언트 프로세스가 이용할 수 있어야 하기 때문에 스터브(Stub)와 스켈레톤(Skeleton)을 만든다.

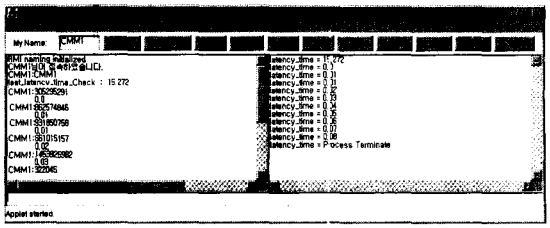


(그림 4) RMI 구조



(그림 5) 자바의 RMI GroupCommunication Sequence Diagram

스터브와 스켈레톤은 서버와 클라이언트 프로세스 사이의 통신을 담당한다. 그림 5는 자바의 RMI를 이용한 그룹 통신 Sequence Diagram이다.



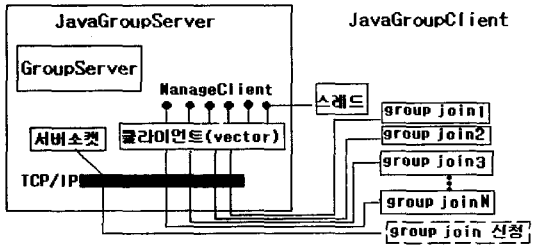
(그림 6) 자바의 RMI 그룹 통신 구현

자바의 RMI 그룹 통신에서는 클라이언트 프로세스가 분산 객체를 이용하기 위해 서버는 자신이 만든 서비스를 네이밍 레지스트리에 등록해야 한다. 이때 네이밍 서비스가 제공하는 bind(), rebind()를 이용한다. 그림 6은 클라이언트 프로세스의 실행화면이다.

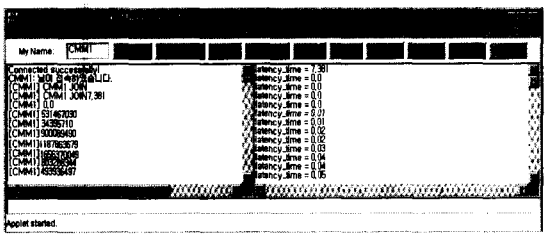
3.4.3 소켓, 멀티캐스트, UDP를 이용한 그룹 통신

소켓을 이용한 그룹 통신은 구현은 그룹 생성, 삭제, 탈퇴, 조인이 가능하다. 또한 그룹 서버는 클라이언트를 스레드 기법을 이용하여 관리한다. 서버와 클라이언트 프로세스 관계는 그림 7과 같다.

그림 8은 소켓을 이용한 그룹 통신 구현 화면이다.



(그림 7) 소켓 그룹 통신의 클라이언트와의 관계

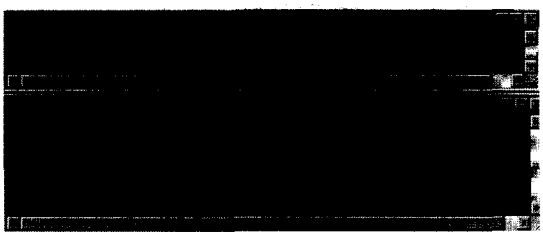


(그림 8) 소켓 그룹 통신 구현

(그림 9) 멀티캐스트 그룹 통신

그림 9는 멀티캐스트로 구현한 그룹 통신에 관한 화면이다. 멀티캐스트 패킷을 수신하려면 멀티캐스트소켓을 생성하고 멀티캐스트 소켓이 제공하는 joinGroup() 프로세스를 이용해 멀티캐스트 그룹에 가입해야 한다.

그림 10은 UDP를 이용한 그룹 통신 실행화면이다.



(그림 10) UDP 그룹 통신

3.5 적용 범위

그룹 통신을 이용한 객체 그룹 패턴은 결함 허용, 효율적인 데이터 보급(dissemination), 부하 균형이나 조합 등에 매우 효과적이다.

- 결함허용 클라이언트/서버 시스템 : 객체 그룹 패턴은 객체의 이용성이 높다. 능동적 복제, 수동적 복제, 객체 그룹을 이용하여 서비스를 제공받을 수 있다.
- 그룹웨어 : 객체 그룹은 화상회의, vod, 분산된 화이트 보드 그리고 다른 종류의 그룹웨어와 같은 어플리케이션에 대한 구현을 쉽게 해준다. 개별적인 그룹간의 정보는 효과적인 방법으로 정형화되고 구현될 수 있다.

4. 성능 측정

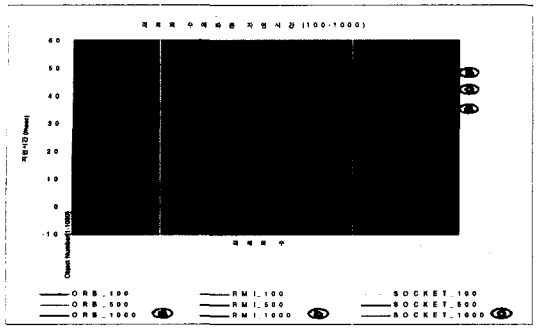
성능측정은 객체의 수를 증가시키면서 지연시간(latency time)을 밀리초로 측정하였다. 지연시간은 클라이언트 프로세스가 서버 그룹에 조인한 후, 서버에게 메시지를 보내고, 서버는 그 메시지를 그룹에 전송하는데 걸리는 시간으로 측정한다. 구현된 성능측정은 NT OS환경 시스템에 P-III 800Mz에서 측정하였다. 각각 구현된 ORB를 이용한 그룹 통신, RMI를 이용한 그룹 통신, 소켓을 이용한 그룹 통신간을 측정하였고, UDP와 멀티캐스트는 메시지 통신간만을 측정했다.

이때, 객체의 수를 증가시키는 방법은 랜덤함수를 이용하여 임의의 객체를 발생시키는 방법으로, 랜덤 참조열 생성(1~N)사이의 수 N개를 발생시킨다.

표 3을 보면 객체의 수가 20개일 때 까지는 소켓을 이용한 그룹 통신이 가장 빠르다. 다음으로 ORB를 이용한 그룹 통신, RMI를 이용한 그룹 통신 순으로 처리 속도를 보인다. 하지만 객체의

(표 3) 그룹간 객체의 수(Object Number) 증가에 따른 지연시간 측정

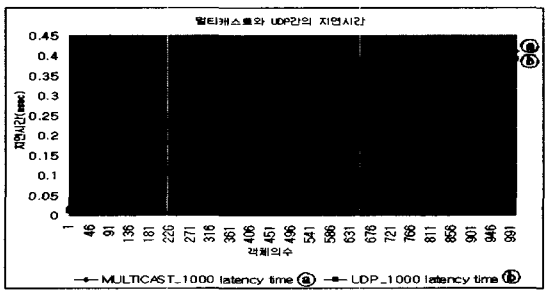
Group / 객체의 수	ORB-G	RMI-G	Sock-G	Multicast -G	UDP-G
10	0.05	0.171	0.04	0.02	0.02
20	0.14	0.571	0.11	0.04	0.02
30	0.24	0.741	0.241	0.04	0.03
40	0.35	0.992	0.401	0.04	0.03
50	0.48	1.272	0.531	0.04	0.04
60	0.621	1.583	0.671	0.05	0.04
70	0.761	1.883	0.852	0.05	0.04
80	1.262	2.204	1.342	0.020	0.05
90	1.412	2.305	1.480	0.021	0.05
100	1.552	2.483	1.592	0.21	0.05
300	6.599	9.954	7.931	0.24	0.141
500	13.169	19.428	16.513	0.27	0.211
700	21.47	30.604	26.518	0.31	0.291
1000	35.431	48.559	41.429	0.41	0.391



(그림 11) 그룹간 객체의 수의 증가에 따른 지연시간 측정

수가 증가하면 그림 11의 측정 결과와 같이 ORB를 이용한 그룹 통신, 소켓을 이용한 그룹 통신, RMI를 이용한 그룹 통신순으로 지연시간이 측정되었다.

그룹 간 객체의 수가 100개일 경우, 500개일 경우, 1000개일 경우의 지연시간을 측정하였다. ⓐ의 경우의 지연시간이 가장 적게 측정되었다. 그림 12는 그룹 통신에서 멀티캐스트와 UDP간의 객체의 수의 증가에 따른 지연시간을 측정하였다.



(그림 12) 멀티캐스트와 UDP간 객체 수의 증가에 따른 지연시간 측정

현 시스템에서는 객체의 수의 증가하여 성능을 측정하는데 한계가 있었기 때문에 SPSS 통계 패키지의 단회귀분석(simple regression analysis)을 이용하여 확률적 예측을 통한 성능 분석을 하였고, 측정된 자료의 단위는 msec이다. 이때, 예측하고 싶은 변수를 목적 변수라 하고, 예측에 사용하는 변수를 설명변수라 하였다. 예측하고 싶은 변수인 목적변수로는 코바를 ORB를 이용한 그룹 통신, 자바를 이용한 RMI 그룹 통신, 소켓을 이용한 그룹 통신으로 하였고, 설명변수는 객체의 수로 하였다. 이때 목적 변수의 데이터는 그림 11에서 측정된 데이터를 사용하였다. 가정은 다음과 같다.

- 종속변수(D)로서 목적변수(예측하고 싶은 변수 : 지연시간)가 될 "y".
- 독립변수(I)로서 설명변수(예측에 사용되는 변수 : 객체의 수)가 될 "x".
- 추정 값(E) : 회귀계수의 추정치를 표시.
- 신뢰구간(N) : 각각의 비 표준화 회귀계수에 대한 95% 신뢰구간을 표시.
- 예측치 보다도 클 때에는 e는 양(+)이 되고, 작을 때는 e는 음(-)이 된다. 이때 e를 잔차(residual)라고 한다.

여기에서 목적변수와 설명변수는 n의 객체의 증가로서 표 3를 이용하였다. 구현한 각각의 그룹 통신에서의 지연시간 x를 이용하여 표 4의 결과를 얻었다.

x는 ORB를 이용한 그룹 통신, RMI를 이용한 그룹통신, 소켓을 이용한 그룹 통신, 멀티캐스트를 이용한 그룹통신, UDP를 이용한 그룹 통신이다. 객체 증가치 n의 데이터 x_i, y_i(i=1,2, ... n)에 $y_i=b_0+b_1x_i$를 적용한다. 구하고 싶은 것은 b0와b1이다. 이 식을 적용 x로부터 y를 예측했다. i번째 y의 예측치 Y_i는 $Y_i=b_0+b_1x_i$가 된다. 따라서 y의 예측치 Y의 차 e는 $e_i=yi-(b_0+b_1x_i)$로 나타낼 수 있다. 잔차 통계량에 따른 예측 값은 표 4와 같다.

표 4의 결과를 보면 코바의 ORB를 이용한 그룹 통신의 경우 평균은 14.5172, 자바의 RMI를 이용한 그룹 통신의 경우 평균은 21.4085, 소켓을 이용한 그룹 통신의 경우 평균은 18.0714가 나왔다. 다음으로 멀티캐스트와 UDP를 이용한 그룹 통신은 데이터 전송간의 시간만을 측정하였으며, 평균은 각각0.2735, 0.2157로 측정되었음을 알 수

(표 4) ORB 잔차 통계

예측값	ORB_G	RMI_G	SOCKET_G	Multi_G	UDP_G
최소값	-3.5488	-3.9539	-3.5903	0.1357	2.700E-02
최대값	32.5832	43.7710	39.7331	0.4113	0.4045
평균	14.5172	21.4085	18.0714	0.2735	0.2157
표준편차	10.4461	14.6650	12.5252	7.969E-02	0.1091
예측값의 표준오차	ORB_G	RMI_G	SOCKET_G	Multi_G	UDP_G
최소값	4.733E-02	5.107E-02	3.338E-02	1.096E-03	2.111E-04
최대값	9.459E-02	.1021	6.671E-02	2.90E-03	4.219E-04
평균	6.533E-02	7.049E-02	4.607E-02	1.513E-03	2.914E-04
표준편차	1.461E-02	1.576E-02	1.030E-02	3.382E-04	6.514E-04
N	1000	1000	1000	1000	1000

있다. 결과로써 첫째로는 코바의 ORB를 이용한 그룹 통신, 두 번째는 소켓을 이용한 그룹 통신, 그 다음은 자바의 RMI를 이용한 그룹 통신순으로 성능을 보였다.

5. 결론 및 향후 연구 방향

본 논문에서는 코바의 ORB를 이용한 그룹 통신과 자바의 RMI를 이용한 그룹 통신, 소켓을 이용한 그룹 통신, UDP와 멀티캐스트에 대한 설계 및 구현을 하였고, 이에 따르는 성능 평가를 실시하였다. 본 실험에서 알 수 있듯이, 자바의 RMI는 TCP 타입의 소켓을 이용하므로 객체의 수가 많은 응용에는 적합하지 않다. 또한 소켓은 전송 효율성이 중요시되는 단순한 데이터의 송수신이 필요한 경우에 적합하고, 분산 객체 클라이언트/서버 시스템은 코바의 ORB를 이용한 기법이 효율적임을 알 수 있다. 본 그룹 통신의 응용 범위는 결함 허용 클라이언트/서버 시스템, 그룹웨어, 텍스트 검색 엔진에 이용될 수 있다.

향후 연구 과제는 신뢰성 있는 그룹 통신을 이용하여 실시간 CORBA 시스템과 결함 허용에 대한 연구가 필요하다.

참 고 문 헌

[1] The Common Object Request Broker: Architecture and Specification, Revision 2.3, OMG, 2001.

[2] CORBAservices:Common Object Services Specification. OMG, 2001.

[3] IONA, "Orbix Programmer's Guide," IONA Technologies PLC October, 2000.

[4] S. Maffeis and D. C. Schmidt, "Constructing Reliable Distributed Communication Systems with CORBA," IEEE Communication Magazine, 14(2), Feb 1997.

[5] Silvano Maffeis, & ASean Landis, "Building Reliable Distributed Systems with CORBA," Theory and Practice of Object Systems, John Wiley, New York, April 1997.

[6] V. Wolfe, L. DiPippo, R. Ginis, M. Squadrito, S.Wohlever, I. Zykh, R. Johnston, "Real-Time CORBA," In Proceedings of the third IEEE Real-time Technology and Applications Symposium, pp. 148-157, June 1997.

[7] Silvano Maffeis, "Olsen & Associates, Hunter of Crashed CORBA Objects," January 1996.

[8] Silvano Maffeis, Olsen & Associates, Zurich, - "The Object Group Design Pattern," Proceeding of the USENIX conference on Object Oriented Technologies, Toronto, January 1996.

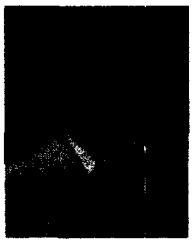
[9] Robert Orfali, Dan Harkey and Jeri Edwards, "The Essential Distributed Objects Survival Guide," John Wiley & Sons, Inc, 1996.

[10] Robbert van Renesse, Ken Birman, and Silvano Maffeis, "Horus: A Flexible Group Communication System," Communications of the ACM 39(4), April 1996.

[11] Z. Yang and K. Duddy, "CORBA:A Platform for Distributed Object Computing," ACM Operating System Review, 30(2):4-31, April 1996.

[12] Richard M. Adler, "Distributed Coordination Models for Client/Server Computing," IEEE Computer, pp. 14-22, April 1995.

◐ 저 자 소 개 ◑



한 윤 기
1997년 청운대학교 전자계산학과 졸업(학사)
1999년 수원대학교 대학원 컴퓨터학과 졸업(석사)
2002년 수원대학교 대학원 컴퓨터학과 졸업(박사수료)
관심분야 : 분산 시스템, 실시간 및 결함허용 처리, 미들웨어
E-mail : hyksea@hanmail.net

구 용 완
1976년 중앙대학교 전자계산학과 졸업(학사)
1980년 중앙대학교 대학원 전자계산학과 졸업(석사)
1988년 중앙대학교 대학원 전자계산학과 졸업(박사)
1983년~현재 : 수원대학교 정보공학대학 컴퓨터학과 정교수
관심분야 : 분산 및 운영체제, 실시간 시스템, 시스템 네트워크 관리, 멀티 미디어, 인터넷 등
E-mail : ywkoo@mail.suwon.ac.kr