분산객체그룹 프레임워크 기반 분산응용 프로그램 개발 도구☆

Developing Tool of Distributed Application Program Based on Distributed Object Group Framework

임 정 택* 신 창 선** 주 수 종***
Jeong-Taek Lim Chang-Sun Shin Su-Chong Joo

요 약

본 논문에서는 분산시스템 상의 클라이언트로부터 요청되는 객체 자원들에 대한 그룹관리 및 동적 바인딩을 지원하는 분산객체그룹 프레임워크 기반에서 편리하게 분산응용 프로그램을 작성할 수 있는 분산 프로그래밍 도구(Distributed Programming Developing Tool: DPDT)를 개발했다. 분산객체그룹 프레임워크는 서버객체에 대한 그룹등록/철회, 접근권한, 이름과 속성서비스 등의 그룹관리 서비스와 동적 바인딩, 중복객체 지원, 부하 균형화 및 분산응용 간의 연동 등의 분산응용 지원 서비스를 제공한다. 분산응용의 개발 시 본 툴을 이용하여, 서버 프로그램 개발자는 해당 서버시스템 상에서 객체들을 구현하고 객체그룹에 서비스 제공에 필요한 속성정보를 등록하며, 클라이언트 프로그램 개발자도 이들 객체 또는 객체그룹에 대한 접근권한을 받아 허가된 객체들의 속성정보를 사용하여 클라이언트 프로그램을 작성할 수 있다. 이를 위해 본 논문에서는 객체그룹에 대한 정의와 본 툴이 지원하는 분산객체그룹 프레임워크의 구조와 기능들을 살펴보고, 분산객체그룹 프레임워크와 분산응용 간의 편리한 인터페이스를 제공할 수 있도록 구현한 DPDT의 3개의 GUI 환경들에 대해 기술하였다. 마지막으로 DPDT를 사용하여, 서버 프로그램으로 구현된 객체들의 그룹등록/철회, 접근권한 부여 및 클라이언트 프로그램의 작성과정과 개발된 분산응용의 수행결과를 보였다.

Abstract

In this paper, we developed the Distributed Programming Developing Tool(DPDT) which can make distributed application program efficiently based on the distributed object group framework supporting group management and dynamic binding for object resources requested from clients on distributed systems. The distributed object group framework we constructed provides not only the group register/withdraw, the access right, and the name/property services for server objects from a point of view of group management services, but also dynamic binding, replicated object supporting, load balance, and federation among the object groups from a point of view of the supporting services of distributed application. When developing distributed application, by using our tool, server programming developer implements objects in each server system, next registers the properties to need for service provision to the object group. Client programming developer can also develop client program easily by obtaining the access right for the object or the object group and using the properties of objects with the access right permitted to the client. For providing above application developing environment, in this paper, we described the definition of object group, the architecture of the distributed object group framework which our tool supports, and its functionalities, then specified the 3 GUI environments of DPDT implemented for providing efficient interfaces between the distributed object group and distributed applications. Finally, by using the DPDT, we showed the group register/withdraw and the access right grant procedure of objects which are server programs, the developing process of client program, and the executing results of the distributed application developed.

☞ Keyword : Distributed Application, Object Group, Distributed Object Group Framework, Distributed Application Developing Tool

* 준 회 원 : 원광대학교 전기전자 및 정보공학부 석사과정
jtlim@wonkwang.ac.kr(제1저자)
** 정 회 원 : 순천대학교 정보통신공학부 교수
csshin@sunchon.ac.kr
*** 종신회원 : 원광대학교 전기전자 및 정보공학부 교수
scjoo@wonkwang.ac.kr
[2005/03/25 투고 - 2005/04/04 심사 - 2005/10/05 심사완료]

☆ 이 논문은 2005년도 한국학술진흥재단-지역대학우수과학자 연구(R-05-2004-000-12006-0)의 지원에 의하여 연구되었음.

1. 서 론

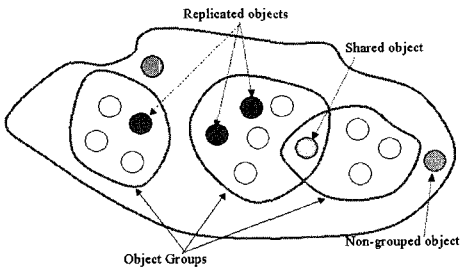
최근 분산 네트워크 컴퓨팅 환경이 보편화되고 멀티미디어 상용 서비스가 일상화되면서, 단일 시스템에서 개발되던 기존 응용들이 분산시스템 상에서 개발되고 있으며 응용간의 복잡한 상호동작에 의한 분산서비스가 이루어지고 있다. 이러한 환경에서 분산서비스는 단일시스템과 달리 네트워크 및 시스템 자원의 공유, 그리고 부하문제를 추가적으로 고려해야만 한다[1,2,3]. 즉, 분산응용이 수행 될 때, 서로 다른 시스템 내의 구성요소들 사이에 복잡한 상호동작을 최소화 시킬 필요가 있다. 이를 위해 분산환경에서 클라이언트의 원격서비스 요청 시, 분산 프레임워크 상에서 분산응용의 수행성을 향상시키기 위한 분산자원 관리 기술들이 필요하다[4,5]. 이러한 요구에 맞추어서 진행해온 대표적인 연구 그룹인 CORBA(Common Object Request Broker Architecture)[6,7]와 TINA(Telecommunication Information Networking Architecture)[8,9]에서는 분산객체 관리모델을 제안하고 있다. 그러나 위 그룹들의 연구들은 분산객체들의 객체그룹 관리에 대한 상세정의를 내리고는 있지만, 분산 응용 서비스의 관점에서 객체그룹을 관리할 때, 그룹 내에 존재하는 중복객체들(replicated objects)의 관리와 클라이언트 객체로부터 서비스를 제공하는 서버객체들에 대한 동적바인딩 서비스 스킴은 포함하지 않고 있다. 이러한 스킴들을 적용하기 위해, 우리는 객체그룹모델 개념과 그룹간 또는 객체간 동적바인딩 전략을 제공할 수 있는 분산객체그룹 프레임워크[10-13]를 구현하였으며, 본 연구에서는 분산객체그룹 프레임워크 기반에서 편리하게 분산응용 프로그램을 개발할 수 있는 도구(Distributed Programming Developing Tool: DPDT)를 제안한다. 본 툴은 분산 프로그램 개발의 편리성 및 개발시간 단축을 위해 크게 3 개의 GUI 환경을 포함한다. 각 GUI는 분산응용의 수행환경을 전체적으로 관리하는 객체그룹 운영자(object group administrator) GUI와, 서버 프로그램의 그룹 등록 및 접근권한의 설정이 가능한 서버 프로그램 개발자(server program developer) GUI, 객체그룹으로 제공되는 분산응용의 검색을 통해 서비스 요청 프로그램을 개발하는 클라이언트 프로그램 개발자(client program developer) GUI로 구성된다. 마지막으로 이들 GUI 환경을 이용하여 간단한 분산응용을 구현하고 실행시킨 결과로부터, 본 DPDT가 분산객체 자원들의 그룹관리 및 동적바인딩을 지원하는 분산객체그룹 프레임워크의 지원을 받아 편리하게 분산 프로그램을 작성할 수 있는 분산응용 프로그램 개발 도구 임을 보였다.

본 논문의 구성은 다음과 같다. 2장에서 우리는 DPDT의 개발을 위한 배경 및 기존연구로 객체그룹의 정의와 분산객체그룹 프레임워크의 구조 및 구성요소의 기능들을 소개한다. 3장에서 분산객체그룹 프레임워크가 제공하는 분산서비스와 그들의 기능들을 포함하는 DPDT를 구현한다. 4장에서는 DPDT를 이용한 분산응용 개발관점에서 분산응용의 수행성과 개발의 수월성을 향상시킬 수 있음을 보이고, 5장에서 본 논문의 요약과 향후 연구내용을 기술한다.

2. 연구배경 및 기존연구

2.1. 객체그룹

분산응용을 구성하는 각각의 객체들은 서버객체로서 단일 및 중복형태로 분산 서버시스템들 상에 존재한다. 이러한 분산객체들은 다수의 분산응용 구현을 위해 효율적인 서비스 그룹별 관리 및 공유를 통해서 분산응용에 적절한 서비스를 제공할 수 있어야 한다. 이를 위해, 분산객체그룹 환경에서는 분산응용을 구성하는 분산객체들을 하나의 논리적인 그룹으로 관리하며, 클라이언트는 객체그룹 내의 분산객체들에 대한 식별자나 중복 상황에 대한 인지 없이도 서비스를



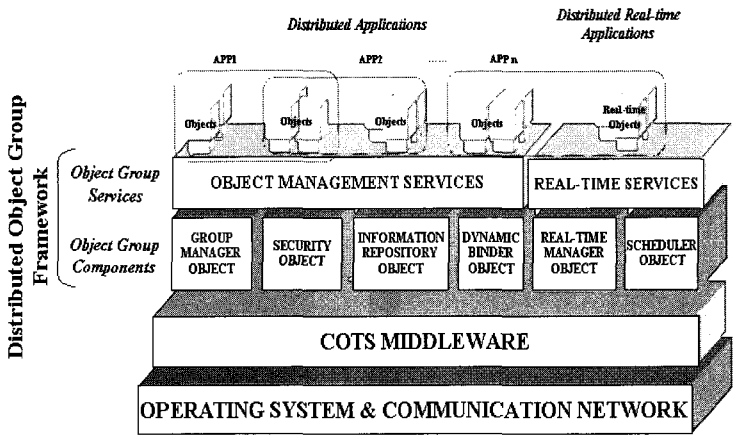
〈그림 1〉 분산응용에 대한 객체그룹들의 도메인

요청할 수 있어야 한다. 즉, 분산시스템 상에 수행되는 하나의 분산응용은 하나 또는 그 이상의 객체그룹들로 구성될 수 있으며, 이들은 하나의 분산응용 수행을 위해 논리적인 단일 뷰 시스템(single view system)으로 관리될 필요가 있다. 그림 1은 분산환경에 존재하는 중복/비중복 및 공유객체들을 그룹핑하여 다양한 분산응용으로 구성하는 객체그룹의 영역을 보인다.

2.2. 분산객체그룹 프레임워크

그 동안 우리는 분산응용 서비스를 수행하는 객체들의 그룹관리 및 단일시스템 환경의 논리적인 체계를 만들기 위해 물리적인 분산시스템 내 분산객체들 간의 복잡한 인터페이스에 대한 분산투명성을 제공할 수 있는 객체그룹 관리 기술들을 연구하여 분산객체그룹 프레임워크를 개발하였다[12-14]. 본 분산객체그룹 프레임워크는 크게 객체그룹 관리 지원 컴포넌트와 실시간 서비스 지원 컴포넌트들로 구성된다. 먼저, 객체그룹 관리 컴포넌트는 그룹관리자객체(Group Manager object: GM), 보안객체(Security object), 정보저장소객체(Information Repository object), 동적 바인더객체(Dynamic Binder object)로 구현되었다. 실시간 서비스 지원 컴포넌트는 실시간관리자객체(Real-Time Manager object: RTM)들과 스케줄러객체 (Scheduler object)들로 구현되었다. 그림 2는 위에서 설명한 분산객체그룹 프레임워크의 구조를 보인다.

분산객체그룹 프레임워크는 분산지원 COTS (Commercial-Off-The-Shelf) 미들웨어와 분산응용의 중간층에 위치한다. 본 프레임워크의 상단에 위치하는 분산응용은 응용서비스의 특성에 따라 각종센서, 장비 또는 정보시스템들로부터 취득한 데이터를 입력으로 비실시간 또는 실시간 응용을 지원할 수 있다. 위에서 열거한 프레임워크 구성요소의 기능과 동작과정을 살펴보면, 그룹관리자객체는 객체그룹 내의 분산객체들의 전반적인 관리를 책임지며, 응용서비스를 지원하는 분산객체들과 클라이언트 간의 바인딩을 지원하기위한 인터페이스 역할을 수행한다. 클라이



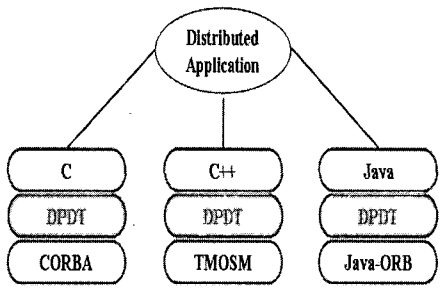
〈그림 2〉 분산객체그룹 프레임워크 구조

언트는 그룹관리자객체를 통하여 분산응용을 지원하는 객체들을 요청하게 된다. 이때, 보안객체는 클라이언트 객체가 요청한 서버객체에 대해 보안정책을 적용하여 접근권한을 검사한다. 접근권한 검사는 접근제어 리스트(access control list)를 참조하여 이루어지며, 접근제어 리스트는 클라이언트명과 서비스명을 속성으로 가진다. 그룹관리자객체는 클라이언트 객체가 서비스 요청시 제공한 클라이언트명과 서비스명을 보안객체에게 전달하고, 보안객체는 접근제어 리스트를 검색하여 클라이언트 객체의 서버객체 접근을 인증한다. 그룹관리자객체로부터 새로운 서버객체의 그룹 소속이나 탈퇴 요청을 받으면 접근제어 리스트를 갱신한다. 다음으로 그룹관리자객체는 정보저장소객체에게 서비스를 수행 할 서버객체의 레퍼런스를 요청한다. 정보저장소객체는 객체들의 속성정보를 저장한 객체리스트(object list)를 포함한다. 서버객체가 비 중복으로 등록되었을 경우 정보저장소객체는 그룹관리자객체에게 해당 서버객체의 레퍼런스를 반환한다. 동적바인더객체는 정보저장소객체에 존재하는 중복 객체들에 대한 각각의 바인딩 우선순위 리스트(binding priority list)를 유지한다. 정보저장소객체로부터 서비스 요청을 받은 중복 서버객체 중 적정 서버객체를 선정전략에 따라 결정한 후, 그룹관리자객체에게 결정된 객체의 레퍼런스를 반환한다. 중복 객체들로부터 적절한 하나의 객체를 선정하기 위해, 동적바인더객체에서는 분산응용의 서비스 특성을 고려하여 다양한 서버객체 바인딩 알고리즘들 중 하나를 적용시킬 수 있다.

실시간관리자객체는 클라이언트로부터 마감시간 정보를 전달 받아 시간제약조건을 적용하여 서비스 마감시간을 계산 후 스케줄러객체에 실시간 스케줄링을 요청한다. 스케줄러객체는 서버객체가 수행해야 할 요청 작업들에 대한 작업 우선순위 리스트(task priority list)를 가지며, 클라이언트 객체 정보와 마감시간 정보를 이용하여 요청작업들을 실시간 스케줄링 한다. 동적바인더객체와 같이 스케줄러객체에도 응용의 특성에 따라 다양한 실시간 알고리즘들을 적용시킬 수 있다. 위의 구성요소들로 이루어진 분산객체그룹 프레임워크는 분산응용을 구성하는 객체들의 등록 및 철회관리, 서버객체에 대한 클라이언트의 접근보안관리, 이름과 속성관리를 지원하며, 분산서비스 관점에서 네이밍 서비스, 동적바인딩 서비스, 다중복객체지원 서비스, 부하균형화 서비스, 응용그룹 간의 연동서비스를 제공한다. 우리는 [14]에서 분산응용의 한 예로 분산방어시스템을 구현하여 분산객체그룹 프레임워크의 수행성을 검증했다. 본 논문에서 제시한 분산프로그램 개발 도구(DPDT)는 위 프레임워크를 기반으로 구현된다.

3. DPDT의 기능구현

DPDT는 2장에서 설명한 분산객체그룹 프레임워크를 포함하고 있으며, 프레임워크로부터 객체그룹 관리 및 분산 동적서비스 지원 API(Application Program Interface)들의 호출을 통해 서버와 클라이언트 프로그램 개발자의 관점에서 편리하게 분산응용을 개발할 수 있도록 하였다. 이를 위해 본 DPDT는 분산응용을 구성하는 분산객체들의 관리 투명성과 편리한 분산응용 개발 GUI 환경을 제공한다. 그림 3에서는 본 툴이 이종의 응용 프로그래밍 언어와 분산 미들웨어에 독립적인 환경에서 분산응용 프로그램을



〈그림 3〉 DPDT와의 호환 언어 및 미들웨어

개발할 수 있음을 보이고 있다.

DPDT 구현 시, 분산응용을 수행하는 서버객체들의 그룹관리를 위한 분산객체그룹 프레임워크의 구성요소들은 객체지향 언어인 C++로 구현했으며, 이를 DLL(Dynamically Linked Library)로 패키징하여 분산응용의 개발자에게 제공한다.

3.1. DPDT 지원 인터페이스

분산객체그룹 프레임워크의 구성요소 중 그룹관리자객체는 분산응용 서비스를 지원하는 그룹내의 모든 객체들을 관리하며, 그룹에서 외부 클라이언트 객체와의 인터페이스를 가지는 유일한 객체이다. 즉, 그룹관리자객체를 통해서만 객체그룹에 대한 객체관리 서비스가 이루어진다. 그룹관리자객체에 구현된 인터페이스를 통해 응용을 구성하는 분산객체는 그룹등록, 보안검사, 동적바인딩 서비스를 받게 된다. 그림 4는 DPDT로부터 지원되는 그룹관리자객체의 기능 인터페이스들을 보인다.

그룹관리자객체의 구현기술과 동일하게 DPDT내에 분산객체그룹 프레임워크의 다른 구성요소들인 보안객체, 정보저장소객체, 동적바인더객체를 구현했다. 구성요소들의 세부적인 기능 및 구조는[10,11,14]를 참조한다.

3.2. DPDT의 GUIs

본 DPDT는 분산응용 개발자들에게 분산객체그룹 프레임워크의 기능들을 편리하게 제공할 수 있도록 3개의 GUI를 포함한다. 구현된 GUI들은 분산응용의 수행환경을 전체적으로 관리하는 객체그룹 운영자 GUI와, 서버객체의 그룹 등록 및 접근권한 설정이 가능한 서버 프로그램 개발자 GUI, 객체그룹으로 제공되는 분산응용의 검색을 통해 서비스 요청 프로그램을 개발하기 위한 클라이언트 프로그램 개발자 GUI로 구성된다.

기존의 분산응용들은 대부분 단일 개발자에 의해 서버와 클라이언트 프로그램이 함께 개발됐었다. 그러나 DPDT상에서는 서버와 클라이언트 프로그램 개발자들은 서로 독립적으로 분산응용을 개발할 수 있다. 서버 개발자는 각 서버 시스템에 맞는 환경으로 서버 프로그램을 개발하고, 본 툴의 기능을 이용하여 분산객체그룹 프레임워크에 등록한다. 클라이언트 개발자는 툴로부터 제공된 클라이언트 GUI를 이용해 객체그

```
#include "DOGST_DLL.h"
class GroupManagerObject {
//객체그룹에 서버객체 등록.
  char* enter_objectgroup(char *group_name, char *service_name, char *object_name,
char *location_address);
//객체그룹에서 서버객체 탈퇴.
  char* withdraw_objectgroup(char *group_name, char *service_name, char *object_name);
//객체그룹에 등록된 서버객체 정보 수정.
  char* modify_objectgroup(char *group_name, char *service_name, char *object_name,
char *location_address);
//클라이언트 객체의 서버객체 접근권한 추가.
  char* insert_access_right(char *client_name, char *group_name, char *service_name);
//클라이언트 객체의 서버객체 접근권한 삭제.
  char* delete_access_right(char *client_name, char *group_name, char *service_name);
//서비스를 수행할 서버객체 레퍼런스 요청.
  char* request_object_infoToIRO(char *client_name, char *group_name, char *service_name);
};
```

〈그림 4〉 그룹관리자객체의 기능 인터페이스

룹에서 제공되는 서비스를 검색한 후, 서버로부터 접근권한이 허용된 객체들을 참조하여 클라이언트 프로그램을 개발할 수 있다. 다시 말해서, DPDT는 서버와 클라이언트 개발자로 하여금 각각의 통신 서비스와 서비스객체에 대한 요청 부분만을 프로그램하고, 객체그룹에 등록된 분산서비스 제공 서버객체들을 서로간에 공유 및 재사용함으로써 응용의 개발시간을 단축시킬 수 있을 뿐 아니라 효율적으로 분산자원을 활용하여 분산응용을 편리하게 개발할 수 있도록 지원한다.

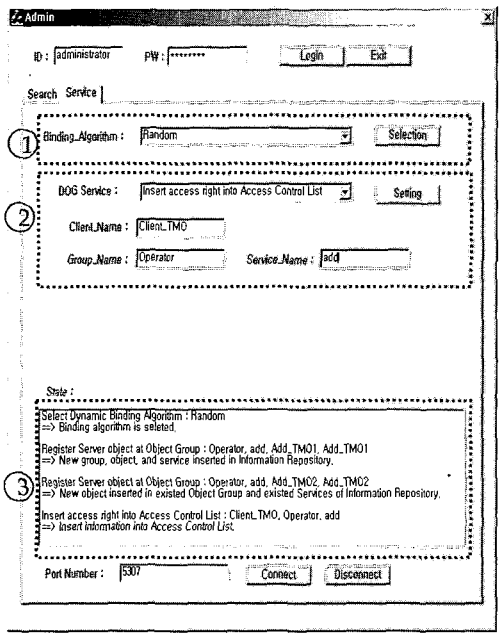
3.2.1. 객체그룹 운영자GUI

객체그룹 운영자의 GUI에서는 분산응용의 수행환경 설정 및 서버객체의 관리를 책임진다. 그림 5는 객체그룹 운영자 GUI로서, 그림 내 ①에서는 객체그룹 내의 중복객체들 중 하나의 객체를 선정하기 위한 기법으로 바인딩 알고리즘을 제공한다. 여기서 제공된 알고리즘들은 무작위선정, 라운드로빈선정, 부하균형화지원 알고리즘들이다. ②에서는 분산응용을 구성하는 서버용 객체들을 서비스명에 따라 객체그룹(*Operator*)에 등록하고, 해당 서비스명(*add()*)에 대한 클라이언트(Client_TMO)의 접근권한을 설정한다. 여기에서 객체그룹 운영자는 서버 프로그램 개발자들이 구현해놓은 모든 객체들에 대해 그룹설정 및 권한설정을 할 수 있다. ③에서는 객체그룹에 대한 설정정보와 이에 대한 접근권한 상태정보가 표시된다. 그림 5의 GUI 환경에서 그룹탈퇴와 클라이언트에 대한 접근권한 제거도 위와 동일한 과정으로 수행된다.

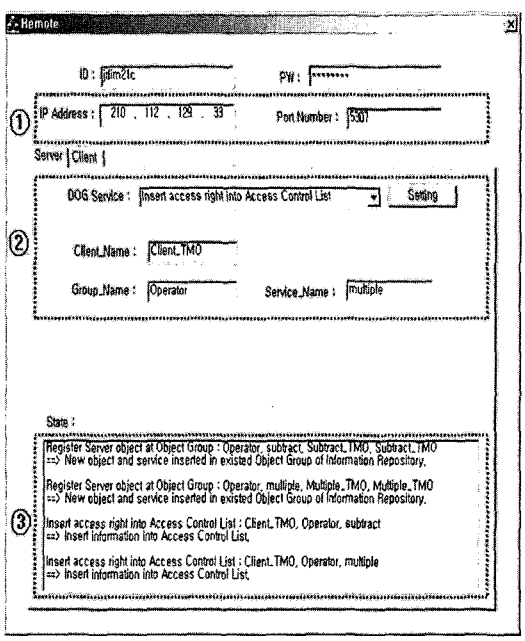
본 GUI를 통하여 객체그룹 운영자는 분산객체그룹 프레임워크에 객체그룹으로 등록된 분산응용들의 수행상태와 환경을 관리하고, 응용의 특성에 따라 프레임워크에서 제공하는 분산서비스를 설정할 수 있다.

3.2.2. 서버 프로그램 개발자 GUI

서버 프로그램 개발자 GUI에서는 서버 프로그램 개발자 자신들이 구현한 서버객체들의 그



〈그림 5〉 객체그룹 운영자 GUI



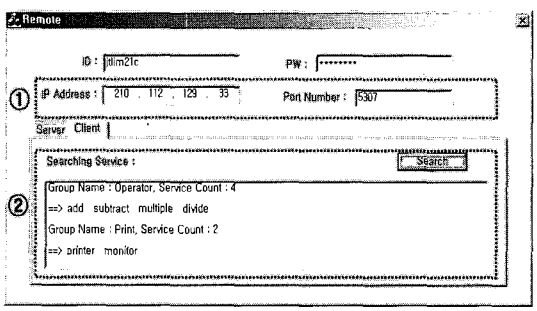
〈그림 6〉 서버 프로그램 개발자 GUI

룹 등록 및 접근권한을 설정하여 클라이언트로 하여금 이들 객체들을 사용할 수 있도록 지원한다. 그림 6의 ①에서는 객체그룹 등록을 위한 프레임워크와 통신 인터페이스를 설정하며, ②에서는 분산응용을 수행하는 서버용 구현 객체들을 그룹에 등록하고, 해당 서비스에 대한 클라이언트들의 접근권한을 부여한다. 여기에서는 객체그룹 운영자 GUI에서와 달리 각 서버 프로그램 개발자 자신이 구현해 놓은 객체들에 대해서만 한정하여 그룹설정 및 권한설정이 가능하다. ③에서는 서버 프로그램 개발자가 직접 구현한 객체들에 대한 그룹설정 및 권한설정 상태정보가 표시된다.

서버 프로그램 개발자는 클라이언트 프로그램과 독립적으로 서비스 프로그램을 개발할 수 있다. 이때, 등록된 그룹 내 클라이언트의 접근권한을 설정함으로써, 클라이언트는 서버 프로그램 개발자가 구현한 객체들의 물리적 위치와 관계없이 접근이 허가된 객체들로부터 투명하게 분산서비스를 제공받을 수 있다.

3.2.3. 클라이언트 프로그램 개발자 GUI

본 GUI는 클라이언트 프로그램 개발자에게 분산응용을 개발할 때, 사용할 수 있는 그룹과 객체정보들을 검색할 수 있는 기능을 제공한다. 그림 7의 ①에서는 사용할 객체그룹이나 객체들의 접근을 위해 분산객체그룹 프레임워크와 통신환경을 설정하며, ②에서 객체그룹에 등록된 서버



〈그림 7〉 클라이언트 프로그램 개발자 GUI

프로그램의 속성정보를 검색한다. 검색결과로 해당 클라이언트가 사용할 수 있는 객체그룹 이름과 서비스명이 출력된다.

클라이언트 프로그램 개발자는 객체그룹에 등록된 그룹정보와 서비스 정보를 검색하여 클라이언트가 접근할 수 있는 서비스를 선택 후, 검색된 그룹 및 허가된 객체들을 참조하여 클라이언트 프로그램을 개발한다.

4. DPDT를 이용한 분산 프로그램 개발 과정

본 장에서는 우리가 개발한 DPDT를 이용한 분산 프로그램 개발의 편리성과 분산서비스 수행성을 검증한다. 분산응용을 구성하는 객체들을 구현하고 분산시스템 간의 통신지원을 위해 UC at Irvine의 DREAM Lab.에서 개발한 실시간 객체 모델인 TMO(Time-triggered and Message-triggered Object) 스킴과 미들웨어인 TMOSM (TMO Support Middleware)을 이용하였다. TMO 스킴과 TMOSM에 대한 세부적인 특성 및 구조는 [15-17]에서 자세하게 기술하고 있다. 본 논문에서의 TMO스킴과 TMOSM은 자체적으로 분산 실시간 서비스 구현 및 미들웨어 지원이 가능하기 때문에 분산객체그룹 프레임워크를 개발할 때, 내부 구성요소들 중 실시간관리자객체와 스케줄러객체의 서비스 지원 API들은 구현하지 않았다. TMOSM에서 제공되지 않는 객체그룹 지원 및 동적 객체바인딩 지원 API들만을 분산객체그룹 프레임워크 내에 구현하여 DPDT의 기능을 강화하였다.

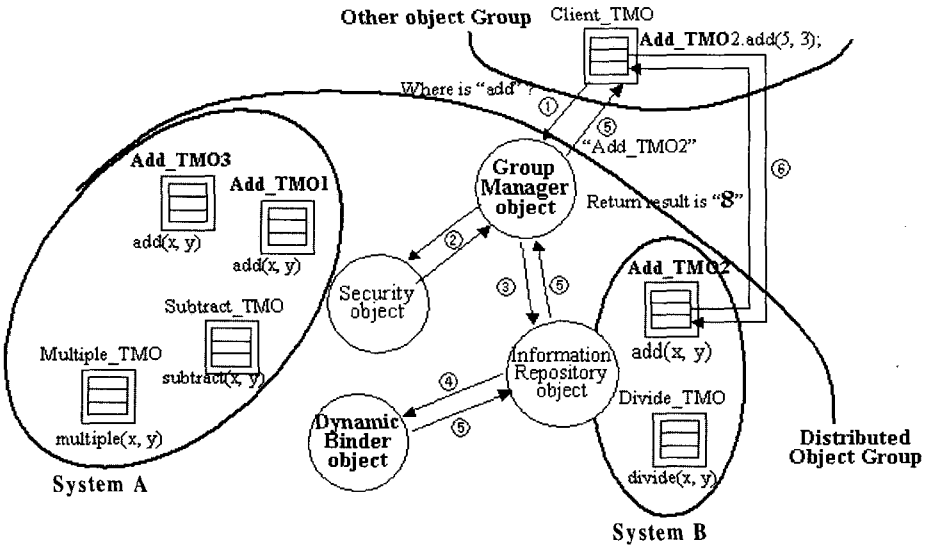
4.1. DPDT를 이용한 분산응용의 서버 프로그램 개발 및 수행실험

본 절에서는 TMO스킴 기반으로 구현된 분산응용의 예를 보이고, DPDT상에서 수행과정을 고찰한다. DPDT를 이용하여 4개의 오퍼레이션들(*add(),*

subtract(), multiple(), divide())을 갖는 분산응용을 구현한다. 이들 오퍼레이션들은 분산된 두 시스템 상에서 서버 프로그램 개발자에 의해 TMO들로 구현하였다. 즉, Add_TMO1, Add_TMO3, Subtract_TMO, 그리고 Multiple_TMO들은 시스템 A에 위치하며, Add_TMO2와 Divide_TMO들은 시스템B에 위치한다. Add_TMO1, Add_TMO2 및Add_TMO3들은 동일한 서비스(*add()*) 특성을 갖는 중복객체들(replicated objects)이다. 클라이언트 요청에 따라 중복객체 중 한 객체와 최적 바인딩이 제공되어야 한다. 이를 위해 분산응용에 적합한 바인딩 알고리즘을 선택적으로 제공할 수 있도록 하였다.

DPDT를 이용한 분산응용의 세부적인 개발과정은 다음과 같다. 먼저, 각 서버 프로그램 개발자는 응용에 해당하는 서버객체를 구현한 후, 그림 6에서 보인 서버 프로그램 개발자 GUI 환경에서 이들에 대한 그룹(*Operator*)을 설정하고 정해진 클라이언트에 대한 접근권한을 부여한다. 클라이언트 프로그램 개발자는 그림 7에서와 같이 GUI 환경을 통해 그룹으로 제공되는 서비스들을 검색하고, 허가된 권한을 가진 객체그룹 및 객체정보들을 사용하여 클라이언트 프로그램을 개발한다. 서버와 클라이언트 프로그램 개발자들은 각자 독립적으로 프로그램을 개발할 수 있으며, 그룹 내에서 연산서비스는 정보저장소객체에 의해 물리적인 위치와 무관하게 분산 위치투명성을 가진다. 위와 같은 수행환경이 갖추어질 때, 분산응용의 수행절차는 아래 그림 8과 같이 동작하게 된다.

먼저, ①에서 클라이언트는 분산객체그룹 프레임워크의 그룹관리자객체에게 *add(5,3)* 서비스를 수행하는 객체(Add_TMO?)의 레퍼런스를 요청한다. 아직 중복 객체들(Add_TMO1, Add_TMO2 및Add_TMO3) 중 어느 서버객체가 해당 서비스를 수행할 것인지 결정되지 않았다. ②에서 그룹관리자객체는 클라이언트의 요청에 대해 해당 서비스를 제공하는 서버객체에 대한 접근권한을 검사한다. ③ 접근이 허가된 객체인 경우, 해당 객체의 레퍼런스를 정보저장소객체에게 요청한다. 정보저장소객체는 *add()* 서비스를 수행하는 객체가 중복된 객체인지를 확인하여, 비중복일 경우 요청한 객체의 레퍼런스를 즉시 반환하고, ④ 중복객체(Add_TMO1, Add_TMO2, Add_TMO3)

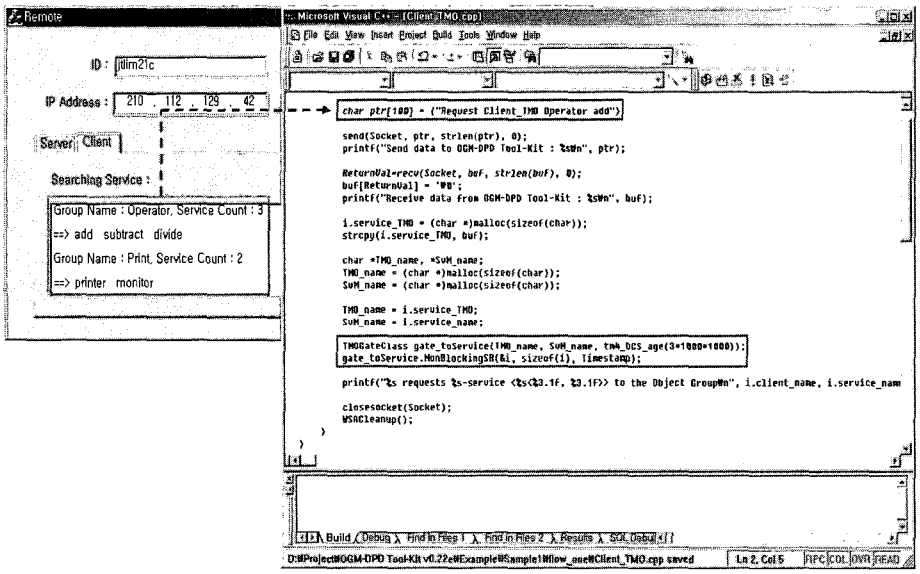


〈그림 8〉 객체그룹 환경에서 TMO 기반 분산응용의 동작절차

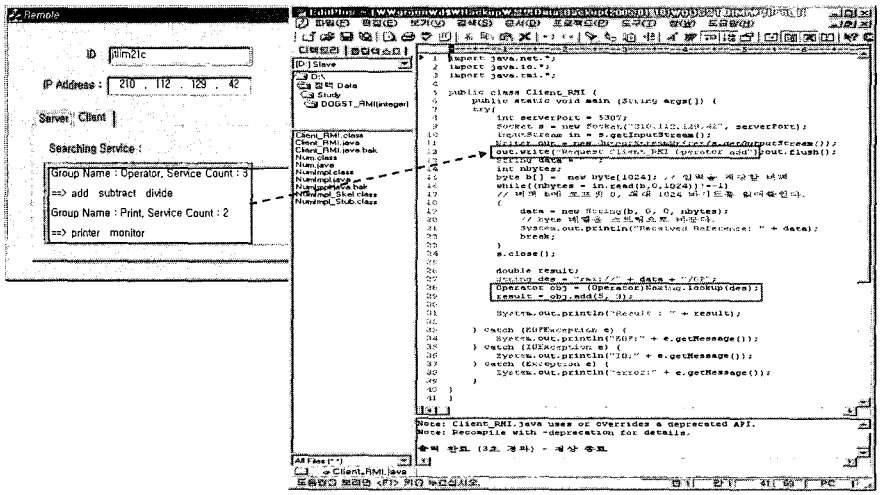
일 경우 동적바인더객체에게 바인딩을 위한 적절한 객체를 선정(Add_TMO2)토록 하여 그의 레퍼런스를 반환 받는다. 그 이후, ⑤에서 선정된 서비스 지원 객체(Add_TMO2)의 레퍼런스를 클라이언트 객체에게 반환하고, ⑥ 클라이언트는 반환 받은 객체(Add_TMO2)의 레퍼런스를 참조하여 바인딩한 후, 자신이 요청한 서버객체로부터 수행결과인 "8"을 받게 된다.

4.2. DPDT를 이용한 분산응용의 클라이언트 프로그램 개발

그림 7에서 본 바와 같이 클라이언트 프로그램 개발자는 DPDT를 이용하여 그룹 내 권한이 허가된 객체를 참조하여 서버환경과 무관하게 독립적으로 클라이언트 프로그램을 개발한다. 사전에 서버 프로그램 개발자는 클라이언트의 고려

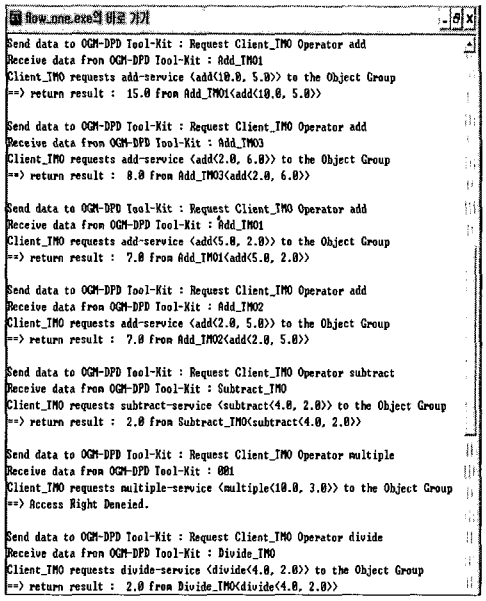


〈그림 9〉 C++로 개발된 클라이언트 프로그램



〈그림 10〉 Java로 개발된 클라이언트 프로그램

없이 서버 프로그램을 개발하여 분산객체그룹 프레임워크에 분산응용에 대한 서버객체들의 속성정보를 등록했기 때문에, 클라이언트 프로그램 개발자는 단지 이들의 속성정보를 이용하기만 하면 된다. 예로, *Operator* 그룹(*add(), subtract(), multiple(), divide()*)과 *Print* 그룹(*printer(), monitor()*)이 분산객체그룹 프레임워크 상에 등록되었을 때, 클라이언트 프로그램에서는 그림 9와 그림 10에서 보이는 바와 같이, 클라이언트 프로그램 개발자 GUI 환경에서 검색된 좌측부분 사각형내의 서비스들을 일반 프로그램 언어에서의 타입 선언처럼 클라이언트 프로그램 상단에 선언한다. 클라이언트 프로그램은 객체선언 부분, 클라이언트와 서버시스템 간에 통신 서비스와 서비스객체에 대한 요청 부분, 그리고 응용처리 부분으로 프로그래밍 된다.



〈그림 11〉 클라이언트 프로그램의 수행결과

4.3. 클라이언트 프로그램의 수행결과

그림 11은 DPDT를 이용하여 구현된 그림 9 클라이언트 프로그램의 수행결과를 보인다. 클라이언트 프로그램에서는 *Operator* 그룹의 *add(), subtract()* 또는 *divide()* 서비스를 반복 요청하여 동적바인딩 서비스 후 각 서비스에 대한 수행결과를 보이고 있다. *add()* 서비스의 결과를 얻기 위해 중복 객체(Add_TMO1, Add_TMO2, Add_TMO3) 중에 한 객체를 요청하여 수행하는 과정을 그림 11로부터 설명한다. 본 그림에서 보인 수행결과를 얻기 위해, 우리는 분산객체그룹 프레임워크 상의 동적바인더객체에 바인딩 알고리즘으로 무작위선정 알고리즘을 사용하였다. 앞서 언급한 것과 같이 동적바인더객체에는 시스템 부하 등을 고려한 다른 알고리즘의 적응이 가능하다. 처음 4개는 클라이언트의 *add()* 서비스 요청들로 Add_TMO1, Add_TMO3, Add_TMO1, Add_TMO2 순으로 중복객체들이 선정되어 호출 후, 바인딩된 *add()* 서비스를 수행한 결과를 보인다. 5번째와 7번째 클라이언트의 *subtract()*와 *divide()* 서비스 요청은 Subtract _TMO와 Divide_TMO를 호출하여 수행결과들을 각각 얻었다. 6번째 *multiple()* 요청에서는 해당 서비스에 대한 접근권한을 얻지 못하여 서버객체(Multiple_TMO)의 호출이 거절됨을 보인다.

5. 결론 및 향후연구

본 논문에서 우리는 분산응용의 효율적인 그룹관리 및 분산서비스를 제공하는 분산객체그룹 프레임워크 지원 분산 프로그램 개발 도구(Distributed Programming Developing Tool: DPDT)을 개발했고, 이를 이용하여 분산응용을 편리하게 구현하는 과정과, 구현된 분산응용의 수행결과를 보였다. 분산객체그룹 프레임워크 기반의 DPDT는 분산응용을 수행하는 객체들을 그룹단위로 관리하며, 서비스 수행 시, 네이밍 서비스, 동적바인딩 서비스, 다중복객체 지원 서비스, 부하균형화 서비스, 응용그룹 간의 연동서비스를 제공한다. 또한 본 툴은 분산 프로그램 개발의 편리성 및 개발시간 단축을 위해 크게 3개의 GUI들

을 포함한다. 각각의 GUI를 살펴보면, 분산응용을 수행하는 전체 서버객체들의 그룹관리 및 접근권한의 설정을 책임지는 객체그룹 운영자 GUI와, 서버 프로그램 개발자 자신들이 구현한 서버객체들의 그룹 등록 및 접근권한을 설정할 수 있는 서버 프로그램 개발자 GUI, 마지막으로 객체그룹 운영자 또는 서버 프로그램 개발자로부터 허가된 객체를 이용한 클라이언트 프로그램 개발을 위한 클라이언트 프로그램 개발자 GUI로 구성된다. 이들 GUI 환경을 이용하여 간단한 분산응용을 구현하고 실행시킨 결과로부터, 우리가 개발한 DPDT를 이용하여 분산 프로그램 개발자들은 분산객체 자원들의 그룹관리 및 동적 바인딩을 지원하는 분산객체그룹 프레임워크의 지원을 받아 편리하게 분산 프로그램을 작성할 수 있음을 보였다. 현재, 본 논문에서 개발한 DPDT는 국내의 공인기관으로부터 프로그램 등록을 받은 상태이다[18].

앞으로 우리는 본 연구결과를 활용하여, 본 툴을 지원하는 분산객체그룹 프레임워크 상에서 헬스케어 홈 서비스 구성요소들과 그들의 다양한 인터페이스들을 확장 및 개발하고, 유비쿼터스 컴퓨팅 환경하에서 헬스케어 기기 및 바이오센서 그리고 의료정보시스템들을 총괄적으로 통합 및 관리서비스를 지원하는 편리한 헬스케어 홈 서비스 프레임워크를 개발하고자 한다.

참 고 문 헌

[1] Andreas Rasche and Andreas Ploze, "Configuration and Dynamic Reconfiguration of Component-based Applications with Microsoft .NET.", In Proceedings of the 6th IEEE International Symposium on Object-Oriented Real-Time Distributed Computing(ISORC), pp.164-171, 2003.

[2] Gorender, S., Macedo, R., and Raynal, M., "A Hybrid and Adaptive Model for Fault-Tolerant Distributed Computing", In Proceedings of Dependable Systems and Networks(DSN'05), pp.412-421, 2005.

[3] Cetintemel, U., Keleher, P.J., Bhattacharjee, B., and Franklin, M.J., "Deno: a decentralized, peer-to-peer object-replication system for weakly connected environments", IEEE Transactions on Computers, Volume 52 Issue 7, pp.943-959, 2003.

[4] V. Kalogeraki, P.M. Melliar-Smith, and L.E. Moser, "Dynamic Scheduling for Soft Real-Time Distributed Object Systems", In Proceedings of the IEEE 3rd International Symposium on Object-Oriented Real-Time Distributed Computing, pp.114-121, 2000.

[5] Yanan Zhang and Yingxu Wang, "An Internet-based distributed system by using real-time CORBA", In Proceedings of the IEEE Electrical and Computer Engineering, Vol.2, pp.1263-1266, 2003.

[6] Object Management Group, "Common Object Request Broker Architecture (CORBA/IIOP) 3.2.", http://www.omg.org/technology/documents/corba_spec_catalog.htm, 2003.

[7] OMG Real-time Platform SIG, "Real-time CORBA A White Paper-Issue 3.0", http://www.omg.org/realtime/real-time_white-papers.html, 2000.

[8] L. Kristiansen, P.Farley, R.Minetti, M. Mampaey, P.F. Hansen, and C.A. Licciardi, "TINA Service Architecture and Specifications", http://www.tinac.com/specifications, 2000

[9] Axel Kupper, "Locating TINA User Agents: Strategies of a Broker Federation and their Comparison", In Proceedings of

the 6th International Conference on Intelligence in Services and Networks(IS&N), 1999.

[10] C.S. Shin, M.H. Kim, Y.S. Jeong, S.K. Han, and S.C. Joo, "Construction of CORBA Based Object Group Platform for Distributed Real-Time Services", In Proceedings of the 7th IEEE International Workshop on Object-oriented Real-time Dependable Systems(WORDS'02), pp.229-302, 2002.

[11] S.C. Joo, C.S. Shin, C.W. Jeong, and S.K. Oh, "CORBA Based Real-Time Object-Group Platform in Distributed Computing Environments", Lecture Notes in Computer Science, Vol.2659, pp.401-411, 2003.

[12] C.S. Shin, M.H. Kim, and S.C Joo, "A Construction of TMO Object Group Model for Distributed Real-Time Services", The Transaction of Korea Information Science Society, Vol.30, No.5-6, pp.307-318, 2003.

[13] C.S. Shin, M.S. Kang, Y.S. Jeong, S.K. Han, and S.C. Joo, "TMO-Based Object Group Model for Distributed Real-Time Services", Proceedings of IASTED International Conference-Networks, Parallel and Distributed Processing, and Applications (NPDPA 2002), pp.178-183, 2002.

[14] Chang-Sun Shin, Chang-Won Jeong, and Su-Chong Joo, "Construction of Distributed Object Group Framework and Its Execution Analysis Using Distributed Application Simulation", Lecture Notes in Computer Science, Vol.3207, pp.724-733, 2004.

[15] K.H. Kim, "Object-Oriented Real-Time Distributed Programming and Support Middleware", In Proceedings of the 7th International Conference on Parallel & Distributed System, pp.10-20, 2000.

[16] K.H. Kim, Seok-Joong Kang, and Yuqing Li, "GUI Approach to Generation of Code-Framworks of TMO", In Proceedings of the 7th IEEE International Workshop on Object-oriented Real-time Dependable Systems(WORDS), pp.17-25, 2002.

[17] Kim, K.H., Ishida, M., and Liu, J., "An Efficient Middleware Architecture Supporting Time-triggered Message-triggered Objects and an NT-based Implementation", In Proceedings of the IEEE CS 2nd International Symposium on Object-oriented Real-time distributed Computing(ISORC'99), pp.54-63, 1999.

[18] S.C. Joo, C.S. Shin, and J.T. Lim, Distributed Programming Developing Tool-Kit Based on Object Group, Program License by Korea Computer Program Deliberation & Mediation Committee, Grant-No.2005-01-172-000228, 2005.

◐ 저 자 소 개 ◑

임 정 택 (Jeong-Taek Lim)
2004년 원광대학교 전기 · 전자 및 정보공학부 졸업(학사).
2004년~현재 원광대학교 컴퓨터공학과 석사과정.
관심분야 : 분산컴퓨팅, 객체지향프로그램
E-mail : jtlim@wonkwang.ac.kr

신 창 선 (Chang-Sun Shin)
1996년 우석대학교 전산학과 졸업(학사).
1999년 한양대학교 컴퓨터교육과 졸업(석사).
2004년 원광대학교 컴퓨터공학과 졸업(공학박사).
2004년~2005 : 원광대학교 헬스케어기술개발센터 Post-Doc.
2005년~현재 : 순천대학교 정보통신공학부 교수
관심분야 : 분산컴퓨팅, 분산알고리즘, 실시간 객체모델
E-mail : csshin@sunchon.ac.kr

주 수 종 (Su-Chong Joo)
1986년 원광대학교 전자계산공학과 졸업(학사).
1988년 중앙대학교 컴퓨터공학과 졸업(공학석사).
1992년 중앙대학교 컴퓨터공학과 졸업(공학박사).
1993년 미국 University of Massachusetts at Amherst, Post-Doc.
2003년 미국 University of California at Irvine, Visiting Professor.
1990년~현재 원광대학교 전기 · 전자 및 정보공학부 교수.
관심분야 : 분산 실시간 컴퓨팅, 분산객체모델, 시스템 최적화, 멀티미디어 데이터베이스
E-mail : scjoo@wonkwang.ac.kr