論文 98-23-9-13

개방형 분산 환경에서 객체그룹 모델의 설계

정회원 이 승 용*, 정 창 원*, 신 영 석**, 주 수 종***

A Design of Object Group Model in Open Distributed Processing Environments

Seung-Yong Lee*, Chang-Won Jeong*, Yong-Suk Shin**, Su-Jong Joo*** *Regular Members*

* 본 논문은 1997년도 한국전자통신연구원(ETRI)의 위탁과제 연구비에 의하여 연구되었음.

요 약

최근 컴퓨팅 환경은 초고속 통신망을 통해 다양한 개방형 멀티미디어 서비스를 제공하고 있으며, 기존의 통신망에 큰 변화 없이 통신망의 복잡성을 감소시키며, 새로운 서비스를 적용시킬 수 있는 객체지향 개념과 분산 시스템을 기반으로 한 개방형 정보통신망 구조로 발전되고 있다.

본 논문은 이러한 개방형 정보통신망에서 제안하고 있는 객체그룹 개념을 기반으로 객체그룹의 모델을 제안함으로 개별된 객체들의 모델링 한계와 객체 관리의 복잡성의 문제점을 해결하는데 목적을 둔다. 먼저 객체그룹 모델에 필요한 요구 사항들과 기능들을 분석하고, 분석된 내용과 요구 사항을 기반으로 객체그룹의 기능 구조를 정립한다. 또한, 제안한 객체그룹 모델을 토대로 관리 및 서비스 기능을 수행하기 위한 객체그룹의 구성 요소들간의 접속 절차와 그에 따른 ETD를 제시하고 TINA-ODL로 객체그룹의 모델을 설계한다.

ABSTRACT

Recently, the distributed processing environments provide various open multimedia services through telecommunication network and have been developing into information networking structure based on object-oriented concepts and distributed systems which can apply new services with a few changes the existing networks. This paper proposes the object group model which is the collection of objects and can functionally and efficiently manage the individual object. This paper presents the analysis of the requirement and the function specifications to propose the object group model, and depicts the functional structure in details using its analysis. The goal of this paper is to decrease the complexity of the object's management and to overcome the limitations of the individual object modeling by suggesting the object group model. Also, we suggest the interaction procedures among the components of object group for management and service functions based on our proposed the object group model and show interaction procedures to ETD(Event Tracing Diagram)s and finally we design the object group model by TINA-ODL.

* 원광대학교 대학원 컴퓨터공학과
** 호남대학교 정보통신공학과
*** 원광대학교 컴퓨터공학과 부교수
論文番號:97440-1203
接受日字:1997年 12月 3日

Ⅰ. 서 론

컴퓨터 기술 분야와 네트워크 기술의 급속한 발전으로 인해 컴퓨팅 환경은 지리적으로는 떨어져 있지만 통신망을 통해 서로 연결되어 하나의 시스템같이 사용되고 있다. 또한, 사용자들의 서비스 요구가 다양해지고 있다. 이로 인해, 분산처리 환경을 이용한 시스템과 다양한 종류의 멀티미디어 서비스를 수용하는 새로운 통신망의 필요성이 대두되고 있다. 따라서, 기존의 통신망에 큰 변화 없이 새로운서비스를 수용하는 통신 소프트웨어 구조의 정립과 분산 시스템의 개발과 관리의 용이성을 위한 유연한 망 구조 개념을 형상화한 개방형 정보통신망 구조[2,4,8,11]가 요구되고 있다. 이러한 요구에 따라,개방형 정보 통신망 구조인 TINA(Telecommunication Information Networking Architecture)의 연구가 컨소시엄을 통해서 활발히 진행되고 있다[3,9]. 개방형 정보통신망을 기반으로 한 서비스, 어플리케이션, 시스템은 전형적으로 분산된 다중 컴퓨팅 노드들에 분산된 객체들의 인스턴스들과 바인딩을 가지고 있는 객체 인스턴스들의 집합으로 구성되므로 시스템의 규모가 커짐에 따라 복잡하다. 따라서 개방형 통신망에서는 객체들의 집합을 관리하는 구조화 메커니즘으로써 객체그룹(Object Group)을 정의하고 있다[5].

본 논문은 이러한 개방형 통신망 구조에서 정의한 객체그룹 개념을 기반으로 요구 사항과 기능을분석하여, 소규모의 동일 영역 분산 처리 환경에 적용할 수 있는 객체그룹 모델링의 요구사항과 기능을 정립하고 객체그룹 모델을 TINA-ODL을 사용하여 설계하였다. 또한, 관리 접속을 위해 상호작용하는 객체그룹 내의 구성요소들간의 접속 절차도와 이를 기반으로 ETD(Event Trace Diagram)를 기술하였다. 또한, 객체그룹 내의 구성 요소인 서비스 객체는 개방형 통신망의 컨소시엄에서 정의된 연산객체(Computational Object) 모델링 규격[5]을 기반으로 몇 가지 기능을 추가하여 정립하였다.

본 논문의 구성은 다음과 같다. 제 2장에서 개방형 통신망 구조에서의 객체그룹에 대한 정의와 구조및 요구 사항과 관리 기능을 살펴본다. 제 3장은 본 논문에서 제안하는 객체그룹 모델링에 대한 구조와 요소들의 기능들을 정립한다. 제 4장은 객체그룹 내의 객체간에 이루어지는 여러 연산들 중에 관리접속인 생명주기 서비스에 따른 오퍼레이션 수행 시 객체그룹의 구성 요소들간의 상호작용에 따른 접속 절차도와 ETD(Event Trace Diagram)를 나타내고, 제 5장은 객체그룹 구성 요소들 중 객체그룹관리자 객체에 대한 TINA-ODL 설계를 하고, 끝으로 논문의 결론과 추후 연구 방향을 기술한다.

Ⅱ. 개방형 통신망에서의 객체그룹

이 장에서는 개방형 통신망 구조에서 제안하는 객체그룹의 정의[1]와 구조 그리고 객체그룹 모델링요구사항에 대해 기술한다[1,4,5].

1. 객체그룹의 정의

개방형 통신망 구조에는 객체들의 집합적인 개념으로 몇 가지 정의[3,9]가 있다. 먼저 '93년도에Bellcore의 OSCA와 INA구조에 정의된 빌딩블럭 개념은 시스템 관리, 캡슐화, 보안의 단위로써 중앙집중적(분산된 소프트웨어 요소들의 관리의 지원이 안됨)하는 특성을 가지고 있다. '94년에는 빌딩블럭 개념이 객체 합성(composition)에 대한 모든 요구를 만족하고 정의될 수 있는 실제 그룹핑 개념에대한 일반화로써 잠재적으로 분산이 가능하고 내포될 수 있으나 추상화가 약한 특성을 갖은Group-94 개념이 정의되었다. Group94는 현재 개방형 통신망 구조에서 제안하는 객체그룹 개념과 유사하다. 또 다른 개념 정의로 '94년에 개방형 통신망 연결 관리 컴포넌트의 프로토타입의 일부분으로써, 한 서비스가 객체들의 그룹에 대한 관리를 위해 개발되어졌고, 그룹과 유사한 특성을 가지면서 하나의 어플리케이션으로 고려되는 객체들의 그룹으로 객체들의 분산이 가능하며 컴포넌트에 있는 객체들 중의 하나는 컴포넌트의 구성(configuration) 관리자로 설계되고 캡슐화, 내포성에 대한 개념이 없는 컴포넌트 개념이 있다. 앞서 말한 개념들의 특성들과 유사성을 가지는 개방형 통신망의 객체그룹특성은 다음과 같이

정의된다.

- 객체그룹은 캡슐화 단위이다.
- 객체그룹은 분산 객체로 구성되는 단위객체이다.
- 객체그룹은 적어도 객체그룹 내의 객체들의 관리를 책임지는 관리자 객체를 포함해야 한다.
- 객체그룹은 서브객체그룹으로 계층적으로 구성된다.
- 객체그룹은 객체들의 비결합(disjoint) 집합이다.

2. 객체그룹 구조

개방형 통신망 구조에서 정의된 객체그룹[9]은 객체그룹의 외부와 접속되는 객체그룹 인터페이스,객체그룹 내에서 연산객체 및 서브객체그룹들을 관리하는 객체그룹 관리자(Group Manager), 객체그룹 내의 구성요소인 연산객체 및 서브객체그룹들로 구성된다.

객체그룹은 캡슐화의 단위이므로 객체그룹의 외부와 접속되는 인터페이스를 필요로 한다. 이러한 외부 인터페이스를 콘트랙트(Contract)라 정의한다. 다른 그룹에 있는 연산객체는 콘트랙트를 통해서 또다른 그룹에 있는 연산객체와 상호작용하고 하나의 객체그룹은 한 개 이상의 콘트랙트를 가질 수 있다. 다음 그림 1은 개방형 통신망에서 제안한 객체그룹의 구조를 나타낸다.

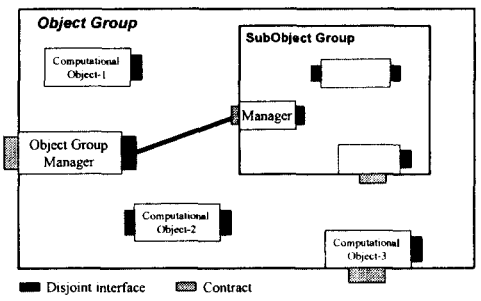


그림 1. 개방형 통신망의 객체그룹 구조
Fig. 1. The structure of Object Group in Open Telecommunication Network

3. 객체그룹의 요구 사항

본 절에서는 개방형 통신망 구조에서 정의한 객체그룹의 구조를 기반으로 구성 요소들 각각에 대한요구사항[10]에 대해 기술한다.

3.1 객체그룹 구성

객체그룹은 분산된 객체가 모인 집합체로 캡슐화된 단위이며, 그룹 단위로 관리 및 유지되고, 객체그룹간의 상호작용을 위한 접속은 객체그룹 밖에 보여지는 외부 인터페이스를 통해서 이루어진다. 객체그룹은 분산처리 환경에서 노드 위치에 관계없이 다른 객체나 객체그룹과 접속이 가능하며, 장애 복구, 가용성, 신뢰성 등을 지원해야 한다. 객체그룹은 동일 분산처리 환경에서 다른 노드 상의 캡슐 단위로 분산된 객체그룹이나 동일 분산처리 환경의 캡슐에 분산된 객체그룹을 서브객체그룹으로 포함할 수 있다. 객체그룹의 효율적인 관리 측면에서 각각 객체그룹 내에 하나의 객체를 객체그룹 관리자로 지정하여 자체적으로 객체그룹 내의 모든 객체들을 관리한다. 객체그룹은 그룹 내의 객체나 서브객체그룹들의 생성과 삭제에 의해 동적인 구성 형태를 가지며, 객체그룹은 서비스 타입의 인스턴스들에 대한 일반화 특성들을 기술한 정책 규약과 인터페이스에 대한 특성들을 기술한 객체그룹 규약을 통해 관리할 수 있어야 한다.

3.2 외부 인터페이스

객체그룹을 구성하는 연산객체들의 인터페이스 중에 외부와의 접속에 사용되는 인터페이스를 외부인터페이스 즉, 콘트랙트라 한다. 객체그룹은 보안의 단위라는 특성을 가지므로 콘트랙트를 이용하여보안 검사로서 엑세스 제어와 인증 검사를 수행한다. 그러나 객체그룹 내에 있는 연산객체들간의 상호작용은 콘트랙트를 통하지 않고 내부 인터페이스를 사용하므로 보안 검사를 거치지 않는다. 또한, 콘트랙트는 외부에서 접근할 수 있도록 레퍼런스를 트레이더나 네임서버에 익스포트(export)시킨다.

3.3 객체그룹의 구성 객체

객체그룹 관리자는 객체그룹 내에서 첫 번째로 인스턴스화 되는 객체로서, 객체그룹 내의 연산객체들과 동일한 캡슐 안에 위치하며, 객체그룹 내 연산객체들과 서브객체그룹을 관리한다. 연산객체와 서브객체그룹의 객체 정보를 객체그룹 내의 객체 인스턴스 레지스터리에 저장하며, 객체그룹 내 객체들간의 매핑하는 기능이 요구된다.

연산객체는 앞서 언급한 개방형 통신망의 연산객체 모델링 규격에 의거하나, 서브객체그룹 내의 연산객체가 다른 객체그룹의 연산객체간에 접속을 요구하는 경우에 객체그룹의 콘트랙트를 통하여 이루어진다.

서브객체그룹은 객체그룹의 요구 사항과 동일하다.

4. 객체그룹의 관리

객체그룹은 객체그룹의 형상(Configuration) 관리에 필요한 기능(오퍼레이션)들을 다음과 같이 크게4가지로 분류한다. 첫째, 객체그룹의 생성과 삭제로써 객체그룹의 생성은 그룹 템플릿(template)을 인스턴스화 시킴으로써 이루어지며, 이때 객체그룹 관리자가 생성된다. 객체그룹 관리자의 생성과 마찬가지로 그룹의 삭제는 객체그룹 관리자를 삭제함으로써 이루어진다. 둘째, 연산객체와 서브객체그룹의생성과 삭제로써 그룹 내의 구성 요소들에 대한 생성과 삭제는 객체그룹 관리자에 의해 행해진다. 이때 객체그룹의 구성 요소가 되는 연산객체는 객체그룹 관리자와 같은 캡슐에 있어야 한다. 셋째, 콘트랙트(외부 인터페이스)의 생성과 삭제로써 객체그룹 관리자는 객체그룹의 콘트랙트 인스턴스를 생성하고 삭제하는 오퍼레이션을 제공한다. 넷째, 객체그룹의 활성화와 비활성화로써 객체그룹 관리자가콘트랙트를 활성화/비활성화 시키는 오퍼레이션을 제공한다.

Ⅲ. 객체그룹 모델링

본 논문에서는 앞서 살펴본 개방형 통신망 구조에서 제안한 객체그룹의 정의를 토대로 그림 2와 같은 구조의 객체그룹을 제안한다. 객체그룹을 구성하는 요소들로, 외부와의 접속을 위해 필요한 콘트랙트라 정의된 외부 인터페이스를 인터페이스 역할에 따라 각각 관리 인터페이스 객체(M-I/F Object)와서비스 인터페이스 객체(S-I/F Object)로 나누어 설계하고, 객체그룹의 구성 요소를 관리하기 위한 객체그룹 관리자 객체(GM Object), 객체들의 정보와 보안을 위해 각각 객체 인스턴스 레포지토리(Object Instance Repository)객체와 보안 레포지토리(Security Repository)객체, 객체그룹 내에 생성가능한 객체들에 대한 정보를 담고 있는 구현맵(Implementation Map)객체가 있으며, 실질적인 서비스를 제공하는 객체(연산객체)들이 존재한다. 또한, 객체그룹 내에 서브객체그룹을 내포할 수 있으며,그 기능과 구성 요소들은 상위 객체그룹과 동일하다. 단, 우리가 제안한 객체그룹의 기본구조에서 상위 객체그룹이 서브객체그룹들을 포함할 수 있으나 그 서브객체그룹은 또 다른 하위의 서브객체그룹(Sub Object Group)을 포함할 수 없다는 전제조건을 둔다.

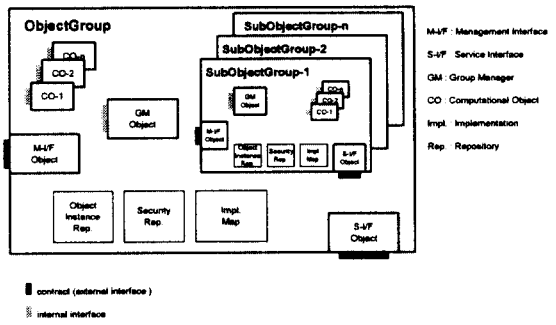


그림 2. 객체그룹의 기본 구조
Fig. 2. The basic structure of Object Group

1. 객체그룹 요소들의 기능 정립

앞에서 살펴본 바와 같이, 하나의 객체그룹을 구성하기 위해서는 여러 구성 요소가 필요하다. 이 절에서는 객체그룹을 구성하는 요소들이 객체그룹 내에서 어떠한 역할을 하며, 구성 요소 각각의 내부구조와 기능에 대해 기술한다.

1.1 관리 인터페이스(M-I/F Object)

객체그룹 외부에서 관리 서비스를 수행하기 위해 객체그룹 내의 연산객체나 서브객체그룹에 접근할 때,

먼저 관리 인터페이스 객체를 통하여 그룹관리자 객체에게 서비스를 요청한다. 이때, 관리 인터페이스 객체는 외부에서 들어온 관리 서비스 요청을 그룹관리자 객체에게 보내기 전에, 서비스를 요청한 개체(entity)가 서비스에 대한 이용 가능여부를 보안 레포지토리 객체를 통하여 확인하고, 그 결과에 따라 서비스 요청을 그룹관리자 객체에게 전달한다. 보안 검사뿐만 아니라 관리 인터페이스 객체가제공하는 기능을 정리하면 다음과 같다. 기본적으로 외부에서 들어온 객체의 생성, 삭제, 활성화, 비활성화와 서브객체그룹의 생성, 삭제, 활성화, 비활성화 요청 메시지를 그룹관리자 객체에게 전달한다.또한 객체에 대한 정보 변경(읽기 가능, 읽기 불가능)과 지정한 객체에 대한 보안 조건의 변경(생성,삭제, 수정 등)을 그룹관리자 객체에게 요구한다. 객체 인스턴스 레포지토리와 보안 레포지토리를 이용하여 각각 객체에 대한 정보 탐색과 요청 객체에 대한 보안 검색을 수행한다. 그리고 그룹관리자를 생성하고 삭제하는 기능을 수행한다. 이는 서브객체그룹의 삭제 시에 서브객체그룹의 구성 요소들의삭제 절차를 간편히 수행하기 위해 관리 인터페이스 객체의 기능에 포함한다.

1.2 그룹관리자(Group Manager)

그룹관리자 객체는 관리 인터페이스 객체로부터 전달된 요구에 따라 요청된 기능을 수행한다. 요청된 기능들은 객체를 생성, 삭제, 활성화, 비활성화와 서브객체그룹의 생성을 위한 서브객체그룹의 관리인터페이스 객체 생성과 삭제, 구성 요소 객체들(객체 인스턴스 레포지토리 객체, 보안 레포지토리 객체, 구현 맵 객체, 서비스 인터페이스 객체)의 생성과 삭제, 서브그룹의 활성화와 비활성화 등이 있다.또한 객체에 대한 정보 변경과 지정한 객체에 대한 보안 조건의 변경을 객체 인스턴스 레포지토리와보안레포지토리를 이용하여 수행한다. 기능 수행 시에 객체 인스턴스 레포지토리 객체, 보안 레포지토리 객체뿐만 아니라 구현 맵 객체와 각각 상호작용을 필요로 한다.

1.3 서비스 인터페이스(S-I/F Object)

객체그룹 외부에서 객체그룹 내부의 연산객체에게 서비스 요청 시에, 해당 서비스 객체와 접속할 수있는 통로 역할을 하는 객체이다. 서비스 인터페이스 객체는 관리가 아닌 실질적인 연산객체의 서비스접속을 위해서 존재한다.

1.4 객체 인스턴스 레포지토리(Object Instance Object)

객체그룹 내에 있는 객체들에 대한 정보들을 관리한다. 레포지토리의 관리는 객체 인스턴스 레포지토리 객체가 담당하고 정보는 객체 정보(Object Information) 객체를 두어 관리하며, 관리 인터페이스객체나 그룹관리자 객체와 상호작용을 한다. 객체 인스턴스 레포지토리는 그룹관리자 객체를 통해 등록된 객체 정보 객체를 저장한다. 객체 정보 객체는 정보 객체 이름과 연산객체의 레퍼런스를 결합(bind)할 정보와 객체의 상태를 관리한다. 객체 인스턴스 레포지토리의 기능은 객체그룹 내 객체들의정보를 생성, 삭제, 추가, 검색 등이다.

1.5 보안 레포지토리(security Repository Object)

객체그룹 내 각 객체들에 대한 보안 규칙을 관리한다. 보안 레포지토리 객체를 두어 레포지토리를관리하며, 보안 규칙을 관리하기 위해서 각 객체마다 하나의 ACL(Access Control List)객체를 두고,하나의 ACL객체마다 여러 개의 Access Rule객체를 두어 관리한다. ACL 객체는 보안 레포지토리 객체에 의해서 관리되며, Access Rule 객체는 ACL객체가 관리한다. 보안 레포지토리 객체는 ACL객체를생성, 삭제, 갱신, 속성 획득 등의 기능을 가지고 있으며, ACL객체는 Access Rule객체를 생성, 삭제,속성 획득, 속성 변경하는 기능이 있다.

1.6 구현 맵(Implementation Map Object)

객체그룹 내에 객체를 생성하기 위해 필요한 객체들과 CORBA 정의 객체간의 레퍼런스를 관리한다. 구현 맵 객체는 객체와 CORBA정의 객체간의 매핑정보를 가지고 있는 Impl 정보 객체들을 두어관리한다. 구현 맵 객체는 지정된 Impl 정보 객체의 생성, 삭제, 갱신, 속성 획득의 기능을 갖는다.

1.7 연산 객체(Computational Object)

연산객체는 실질적인 서비스를 제공하는 객체이다. 이 연산객체는 서비스 요청을 받으면, 서비스를제공하기 전에 보안 레포지토리와 객체 인스턴스 레포지토리를 이용하여 요청 객체의 접근 가능 여부와 자신의 상태(활성화인지 비활성화인지)를 검색한 후, 서비스를 수행한다.

2. 객체그룹 요소들간의 관계

1절에서 살펴본 객체그룹의 구성 요소들이 가지는 각각의 기능들을 실질적으로 수행하기 위해서는구성 요소들간의 상호작용을 필요로 한다. 다음 그림 3은 구성 요소들간의 상호작용을 나타낸다.

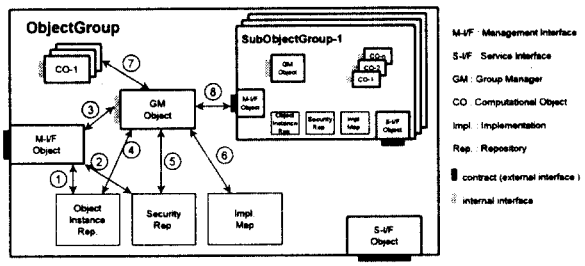


그림 3. 객체그룹 요소들간의 관계
Fig. 3. The components of object group

① 관리 인터페이스 객체는 객체 인스턴스 레포지토리 객체에게 그룹내 객체들에 대한 정보 검색 요청을 하고, 객체 인스턴스 레포지토리 객체는 이에 대한 수행 후 결과를 반환한다.

② 관리 인터페이스 객체는 외부에서 서비스를 요청한 객체에 대한 접근 가능 여부를 보안 레포지토리 객체에게 요청하고 보안 레포지토리는 접근 가능 여부를 반환한다.

③ 관리 인터페이스 객체는 그룹관리자 객체에게 외부에서 요청한 기능 중에 보안 접근 여부와 정보 검색을 제외한, 객체의 생성, 삭제, 활성화, 비활성화와 서브객체그룹의 생성, 삭제, 활성화, 비활성화, 객체 인스턴스 레포지토리에 저장된 객체 정보에 대한 생성, 수정, 보안 레포지토리에 저장된 보안 조건의 수정, 검색 등의 요청을 하고, 그룹관리자 객체는 이에 대한 수행 후 결과를 반환한다.

④ 그룹관리자 객체는 객체 인스턴스 레포지토리 객체에게 객체에 대한 정보를 읽기 가능, 불가능으로 바꾸도록 요구하거나, 객체에 대한 정보를 찾아 줄 것을 요청한다. 이에 대해 객체 인스턴스 레포지토리는 결과를 반환한다.

⑤ 그룹관리자 객체는 보안 레포지토리 객체에게 객체에 대한 보안 조건을 생성, 삭제, 수정을 요청하 고, 외부의 개체가 사용하기를 요구한 객체에 대한 보안 조건을 찾아보도록 요청한다. 또한 ACL 객체를 찾아보도록 요청한다. 즉, 임의의 객체에 대해 외부에서 보안 조건을 검색할 수 있도록 하는 것이다. 이에 대해 보안 레포지토리는 해당 정보나 결과를 반환한다.

⑥ 그룹관리자 객체가 그룹내에 객체나 서브객체 그룹을 생성할 때 구현 맵 객체에게 객체의 매핑정보를 요구하게 되고, 구현 맵 객체는 이에 대한 정보를 반환한다.

⑦ 그룹관리자 객체가 객체를 생성, 삭제, 활성화, 비활성화 할 수 있음을 나타내고 있으며, 이때 그룹관리자 객체는 객체 인스턴스 레포지토리 객체에게 객체 정보를 생성, 삭제, 읽기 가능, 읽기 불가능하게 해주도록 요청한다. 객체 인스턴스 레포지토리 객체는 처리된 결과를 반환한다.

⑧ 상위객체그룹의 그룹관리자 객체가 하위객체 그룹의 관리 인터페이스 객체에게 요구를 전달하고, 하위객체그룹 내부에서 처리된 결과를 하위객체그룹의 관리 인터페이스 객체가 상위객체그룹의 그룹관리자 객체에게 반환한다.

그림 4는 객체그룹 요소들 각각에 대한 기능들과 그들간의 상호작용을 고려하여 객체그룹 요소의기능들을 OMT 기법[5]으로 나타낸다.

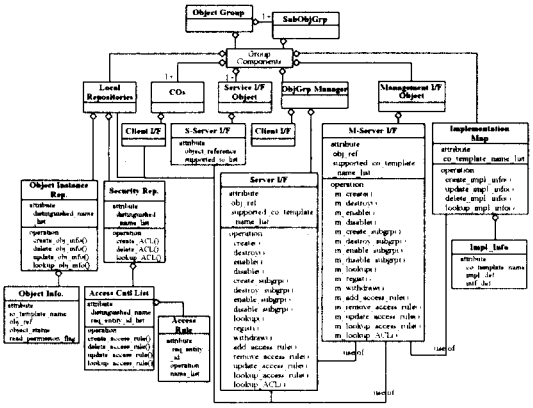


그림 4. 객체그룹 모델의 요소별 기능들
Fig. 4. The functions of components in Object Group model

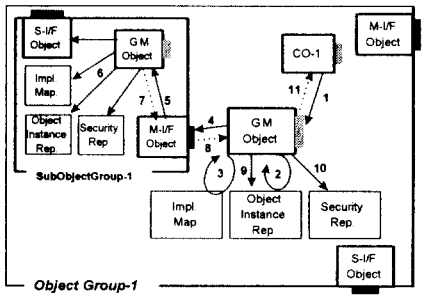


그림 5. 서브객체그룹 생성 절차
Fig. 5. A procedure for creation of SubObject Group

Ⅳ. 객체간의 접속 절차

본 장에서는 객체그룹 내에서 관리 접속을 위해 상호작용하는 객체그룹 내의 구성 요소들간의 접속절차에 대해 살펴본다. 여러 가지 관리 접속을 위한 서비스들 즉, 생성, 삭제, 활성화, 비활성화 중, 객체와 객체그룹의 생성과 삭제에 관련된 절차를 살펴보도록 한다. 먼저 구성 요소들의 상호작용에 대한접속 절차를 살펴본 후, 그에 따른 ETD(Event Trace Diagram)를 이용하여 설명한다.

1. 서브객체그룹 생성

객체그룹 내에 서브객체그룹을 생성하기 위해 그룹 관리자 객체는 먼저 서브객체그룹의 관리 인터페이스 객체를 생성하고 생성된 관리 인터페이스 객체가 서브객체그룹의 내부 구성요소 객체들을 생성한다. 그에 따른 접속 절차도와 ETD가 각각 그림 5와 그림 6이다.

그림 5의 접속 절차를 보면, 연산객체-1이 객체그룹 관리자에게 서브객체그룹-1의 생성을 요청한다.객체그룹-1의 객체그룹 관리자는 먼저 객체 인스턴스 레포지토리에 요청된 서브객체그룹의 존재 여부를 확인한다. 객체그룹 관리자는 구현 맵으로부터 해당되는 서브객체그룹에 대한 템플릿(서브객체그룹의 관리 인터

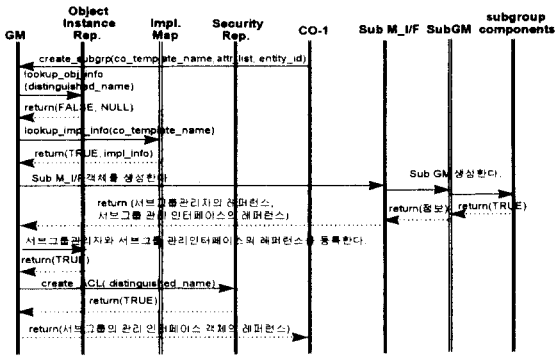


그림 6. 서브객체그룹 생성에 대한 ETD
Fig. 6. ETD for creation of SubObject Group

페이스 객체의 템플릿)을 참조한다. 객체그룹 관리자는 서브객체그룹-1(즉, 서브객체그룹의 관리 인터페이스 객체)을 생성한다. 서브객체그룹의 관리 인터페이스 객체는 서브객체그룹의 객체그룹 관리자 객체를 생성한다. 서브객체그룹-1안의 서브객체그룹 관리자는 서브객체그룹의 요소인 서비스 인터페이스 객체, 보안 레포지토리, 객체 인스턴스 레포지토리, 구현 맵 객체 등을 생성한다. 서브객체그룹-1의 객체그룹 관리자는 관리 인터페이스 객체에게 생성된 서브객체그룹에 대한 정보를 반환한다. 서브객체그룹-1의 관리 인터페이스 객체는 상위 객체그룹 관리자에게 자신의 레퍼런스등의정보를 반환한다. 반환된 정보를 객체그룹-1의 객체그룹 관리자는 객체 인스턴스 레포지토리에 등록한다. 객체그룹-1의 객체그룹 관리자는 서브객

체그룹-1에 대한 접근 권한 정보를 보안 레포지토리에 등록한다. 객체그룹-1의 객체그룹 관리자(서브객체그룹 생성 요청을 받은 그룹관리자)는 생성된 서브객체그룹-1에 대한 레퍼런스를 연산객체-1에 반환한다.

2. 서브객체그룹내에 연산객체 삭제

서브객체그룹 내의 연산객체 삭제는 서브객체그룹의 관리 인터페이스 객체를 통해서 서브객체그룹의 그룹관리자 객체에게 삭제 메세지가 전달되는 차이점을 제외한 나머지 과정은 객체그룹의 연산객체 삭제 과정과 동일하다. 그에 따른 접속 절차도와 ETD가 각각 그림 7과 그림 8이다.

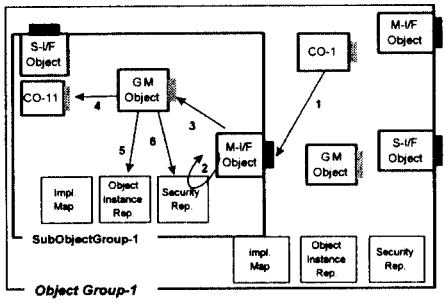


그림 7. 서브객체그룹내에 연산객체 삭제 절차
Fig. 7. A procedure for deletion of object in SubObject Group

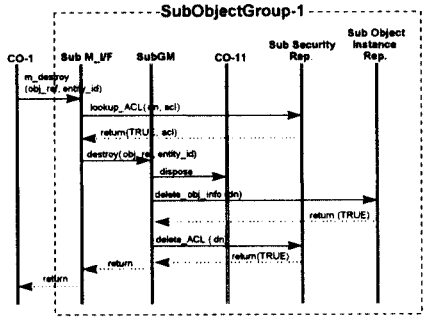


그림 8. 서브객체그룹내에 연산객체 삭제에 대한 ETD
Fig. 8. ETD for deletion of object in SubObject Group

그림 7의 접속 절차를 보면, 객체그룹-1의 연산객체-1이 서브객체그룹-1에 있는 연산객체-11의 삭제 요청을 한다. 서브객체그룹-1의 관리 인터페이스 객체는 서브객체그룹-1의 보안레포지토리를 통하여 연산객체-1의 접근 권한 여부를 확인한다. 접근 권한이 확인되면, 서브객체그룹-1의 관리 인터페이스 객체는 서브객체그룹-1의 객체그룹 관리자에게 연산객체-11 삭제 요청을 보낸다. 서브객체그룹-1의 객체그룹 관리자는 연산객체-11을 삭제한다. 서브객체그룹-1의 객체그룹 관리자는 서브객체그룹-1의 객체 인스턴스 레포지토리에 등록된 연산객체-11에 대한 정보를 삭제한다. 서브객체그룹-1의 객체그룹 관리자는 보안 레포지토리에 등록되어 있는 연산객체-11에 대한 접근 권한 정보를 삭제한다.

3. 연산객체나 서브객체그룹의 활성화/비활성화

객체그룹 내의 연산객체나 서브객체그룹의 상태를 활성화에서 비활성화로 또는 비활성화에서 활성화 상태로 변경시키기 위한 절차이다. 연산객체에 대한 상태는 객체 인스턴스 레포지토리 객체가 저장하고 있으므로, 그룹관리자 객체가 객체 인스턴스 레포지토리 객체와의 상호작용으로 오퍼레이션을 수행한다. 그에 따른 접속 절차도와 ETD는 각각 그림 9와 10이다.

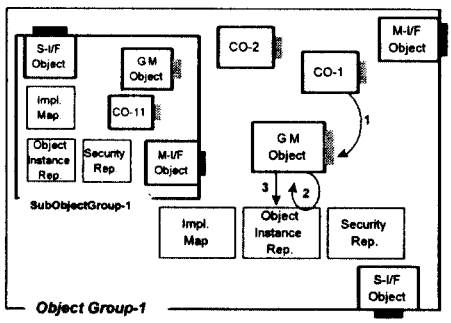


그림 9. 연산객체/서브객체그룹의 활성화/비활성화 절차
Fig. 9. A procedure for activation/deactivation of object/SubObject Group

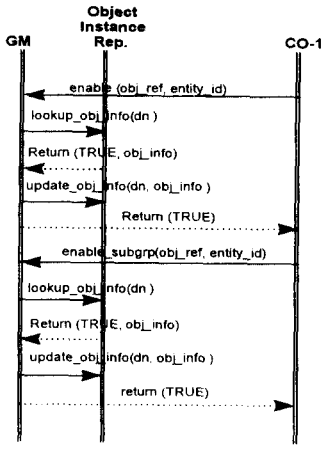


그림 10. 연산객체/서브객체그룹의 활성화/비활성화에 대한 ETD

Fig. 10. ETD for activation/deactivation of object/SubObject Group

그림 9의 접속 절차를 보면, 객체그룹-1의 연산객체-1이 연산객체-2/서브객체그룹-1의 활성화/비활성화 요청을 객체그룹-1의 객체그룹 관리자에게 요청한다. 객체그룹-1의 객체그룹 관리자는 객체그룹-1의 객체 인스턴스 레포지토리를 통하여 연산객체-2/서브객체그룹-1의 존재 여부를 확인한다. 객체그룹-1의 객체그룹 관리자는 객체 인스턴스 레포지토리에 있는 연산객체-2/서브객체그룹-1의 상태 값을 활성화/비활성화로 바꾼다.

V. 객체그룹 모델 설계

앞서 기술한 객체그룹 모델의 기능 정립과 구조를 기본으로 객체그룹 모델을 설계한다. 여기에서는구성요소 객체들인 그룹관리자 객체, 관리 인터페이스 객체, 객체 인스턴스 레포지토리 객체, 보안 레포지토리 객체, 구현 맵 객체에 대한 프로토타입 설계만을 제시하고, 설계 표기법은 TINA-ODL[6]을사용한다.

1. 그룹관리자 객체

그룹 관리자 객체의 인스턴스(instance)는 객체그룹 내의 객체들 및 서브객체그룹을 관리하기 위해서 Obj_Repos_intf 인터페이스의 create_obj_info, delete_obj_info, update_obj_info, lookup_obj_info등의함수들을 호출하고, 객체그룹에 대한 보안을 지원하기 위해서 Security_Repos_intf 인터페이스의create_ACL, delete_ACL, lookup_ACL 함수들을 호출하며, 객체 및 서브객체 그룹의 생성을 위해 구현 맵 객체 인터페이스의 create_impl_info, update_impl_info, delete_impl_info, lookup_impl_info함수를 호출한다.

그룹 관리자 객체의 인터페이스는 객체그룹 내의 객체 인스턴스 레포지토리와 보안 레포지토리 및구현 맵을 관리하는 서비스를 제공하며, 객체와 서브객체 그룹에 대한 생명주기 서비스를 제공한다.

```
object Group_Manager { •
  requires
  M-Server_intf,   Obj_Repos_intf,   Security_Repos_intf,
  Impl_Map_intf;
  supports Grp_Mng_intf;
  initial    Grp_Mng_intf;
   };
interface Grp_Mng_intf {
  struct  Impl_Info {
  Implementation_Definition   impl_def ;
  Interface_Definition          intf_def ;
            };
  typedef     sequence<Impl_Info>    Impl_Info_List ;
    /* 연산객체의 속성 값과 속성 이름 */
  struct  Attr { string                  attrName ;
                 any                     attrValue ;
            };
  typedef     sequence<Attr>          Attr_List ;
   /* 객체 인스턴스 레포지토리를 관리하기 위한 구조 */
  struct  Object_Info {
       Template_Name          io_template_name ;
       Object_Reference       obj_ref ;
       char                   object_status ;
       char                   read_permission_flag ;
            };
 typedef    sequence<Obj_Info>        Obj_Info_List ;
  struct  Security_Tag  {
```

```
        Req_Entity_Id  entity_id ; /* 요청한 개체의 ID */
        Operation_Name_List  allowed_operation_list ;
  /*허용되는 오퍼레이션의 종류 */
            };
  typedef  sequence<Security_Tag>  Security_Tag_List ;
     attribute  Object_Reference        obj_ref ;
/* 연산객체 생명주기 서비스 */
  Object_Reference create(in  Template_Nameco_
    template_name, in Req_Entity_Id  entity_id) ;
     void  destroy(in Object_Reference  obj_ref,
                    in Req_Entity_Id  entity_id) ;
     void  enable(in Object_Reference  obj_ref) ;
     void  disable (in Object_Reference  obj_ref) ;
       /* 서브객체그룹 생명주기 서비스 */
  Object_Reference create_subgrp(in Template_Name
     mo_template_name, in Req_Entity_ID entity_id ) ;
  void  destroy_subgrp(in Object_Reference
     obj_ref, in Req_Entity_Id  entity_id) ;
  void  enable_subgrp(in Object_Reference obj_ref) ;
  void  disable_subgrp(in Object_Reference obj_ref) ;
     /* 객체인스턴스 레포지토리 관리 서비스 */
  void  regist(in DN  distinguished_name) ;
  void  withdraw (in DN  distinguished_name) ;
     /* 보안 서비스 */
     void add_access_rule(in DN distinguished_name) ;
  Security_Tag lookup_access_rule(in DN distinguished_
                                  name ) ;
  void  remove_access_rule(in DN  distinguished_name) ;
  void  update_access_rule(in DN  distinguished_name ) ;
  Security_Tag_List m_lookup_ACL(in DN distinguished_
                                  name ) ;
};
```

2. 관리인터페이스 객체

관리 인터페이스 객체의 인스턴스는 지정된 Obj_Repos_intf 인터페이스의 lookup_obj_info 함수를호출하고, Security_Repos_intf 인터페이스의 Lookup_ACL 함수를 호출하며, Grp_Mng_intf 인터페이스의 객체 및 서브그룹객체 생명주기에 관련된 함수들을 호출한다.

관리 인터페이스 객체의 인터페이스는 객체그룹 내의 객체 인스턴스들에 대한 검색 서비스와 보안서비스를 제공하며, 객체 및 서브객체그룹의 생명주기 서비스를 그룹관리자 객체에게 전달한다. 또한 객체그룹의 외부 인터페이스로 사용된다.

```
object Management_Interface {
 requires Grp_Mng_intf,     Obj_Repos_intf,
          Security_Repos_intf;
 supports M-Server_intf;
initial      M-Server_intf;
};
interface M-Server_intf {
 struct  Attr {  ·  ·  ·  };
  typedef     sequence<Attr>          Attr_List ;
 struct  Obj_Info {  ·  ·  ·  };
  typedef     sequence<Object_Info>  Object_Info_List ;
 struct  Security_Tag {  ·  ·  ·  };
 typedef     sequence<Security_Tag> Security_Tag_List ;
    attribute     Object_Reference     obj_ref ;
    attribute     Template_Name_List
                  supported_co_template_name_list ;
 Object_Reference m_create(inTemplate_Name
   co_template_name, in Req_Entity_Id  entity_id) ;
   void m_destroy(in Object_Reference  obj_ref,
    in Req_Entity_Id  entity_id) ;
   void m_enable(in Object_Reference  obj_ref) ;
   void  m_disable(in Object_Reference  obj_ref) ;
 Object_Reference  m_create_subgrp(in Template_Name
    mo_template_name, in Req_Entity_ID  entity_id) ;
   void  m_destroy_subgrp(in Object_Reference obj_ ref,
    in Req_Entity_Id  entity_id) ;
   void m_enable_subgrp(in Object_Reference  obj_ref,
    in Req_Entity_Id  entity_id) ;
   void m_disable_subgrp(in Object_Reference  obj_ref,
    in Req_Entity_Id  entity_id) ;
     /* 객체그룹 내 연산객체 검색 서비스 */
 Obj_Info_List m_lookup(in Template_Name  io_template
   _name, in char  search_policy, in Req_Entity_Id
   entity_id) ;
   void m_regist(in DN  distinguished_name,
                 in Req_Entity_Id  entity_id) ;
```

```
  void m_withdraw(in DN  distinguished_name,
  in Req_Entity_Id  entity_id) ;
  void m_add_access_rule(in DN  distinguished_name,
  in Security_Tag  sec_tag, in Req_Entity_Id  entity_id);
Security_Tag m_lookup_access_rule(in DN distinguished_
  name, in Req_Entity_Id entity_id);
  void m_remove_access_rule(in DN  distinguished_
  name, in Req_Entity_Id entity_id) ;
  void m_update_access_rule(in DN  distinguished_
  name, in Security_Tag  sec_tag, in Req_Entity_Id
  entity_id) ;
Security_Tag_List m_lookup_ACL(in DN  distinguished_
  name, in Req_Entity_Id  entity_id);
};
```

3. 객체인스턴스 레포지토리 객체

객체 인스턴스 레포지토리 객체의 인터페이스는 객체 정보를 생성, 삭제, 갱신, 탐색하는 객체 인스턴스 레포지토리를 관리하는 서비스를 제공한다.

```
object Object_Instance_Repository {
  supports Obj_Repos_intf ;
  initial    Obj_Repos_intf ;
};
interface Obj_Repos_intf {
 attribute    DN_List            distinguished_name_list ;
   /* 연산객체들의 이름 목록 */
boolean create_obj_info(in DN  distinguished_name,
                        in Obj_info  obj_information) ;
 boolean delete_obj_info(in DN  distinguished_name) ;
 boolean update_obj_info(in DN  distinguished_name,
                         in Obj_info  obj_information ) ;
 boolean lookup_obj_info(in DN  distinguished_name,
                         out Obj_info obj_information ) ;
};
```

4. 보안 레포지토리 객체

보안 레포지토리 객체의 인스턴스는 Access Rule을 관리하기 위한 서비스를 제공하는 ACL 인터페이스의 create_access_rule, delete_access_rule, update_access_rule, lookup_access_rule 함수를 호출한다.

보안 레포지토리 객체의 인터페이스는 ACL객체를 생성, 삭제, 검색할 수 있는, ACL객체를 관리하는 서비스를 제공한다.

```
object Security_Repository {
   requires
            ACL_intf ;
   supports
            Security_Repos_intf ;
   initial
            Security_Repos_intf ;
};

interface Security_Repos_intf {

attribute     DN_List          distinguished_name_list ;

boolean  create_ACL (in DN  distinguished_name) ;
boolean  delete_ACL (in DN  distinguished_name) ;
boolean  lookup_ACL (in DN  distinguished_name,
                     out ACL  acl) ;
};
```

5. 구현 맵 객체

구현 맵 객체의 인터페이스는 구현 정보 객체의 생성, 삭제, 갱신, 검색할 수 있는, 구현 정보 객체를 관리하는 서비스를 제공한다.

```
object Implementation_Map{
  supports      Impl_Map_intf ;
  initial       Impl_Map_intf ;
};
interface Impl_Map_intf {
          Struct  Impl_Info { · · · };
attribute  Template_Name_List  co_template_name_list ;
 /* Impl_Information 객체를 식별하기 위한 연산객체 템
    플릿의 목록 */
boolean create_impl_info(in Template_Name  co_template
  _name, in Impl_Info  impl_info);
boolean delete_impl_info(in Template_Name  co-template
  _name ) ;
```

```
    boolean update_impl_info(in Template_Name  co_template
      _name, in Impl_Info  impl_info);
    boolean lookup_impl_info(in Template_Name co_template
      _name, out Impl_Info  impl_info);
};
```

Ⅵ. 결 론

컴퓨터와 네트워크 기술의 발전으로 인해, 통신망을 기반으로 구성되는 분산처리 환경에서 다양한서비스를 수용할 수 있는 환경의 필요성이 대두되고 있다. 따라서 기존의 통신망에 큰 변화 없이 요구되는 서비스를 수용할 수 있는 객체 지향 소프트웨어 기반인 개방형 정보통신망 구조의 연구가 활발히 이루어지고 있다. 개방형 통신망 기반의 분산처리 환경에서 분산 시스템은 분산 객체들로 이루어져다양한 서비스들을 지원한다. 이러한 분산 시스템의 규모가 커짐에 따라 객체들의 빈번한 상호작용과관리의 복잡성이 증가되므로, 개방형 통신망에서 객체들의 집합체인 객체그룹을 정의하고 있다. 이러한 이유로 본 논문에서는 객체지향 기법을 이용한 객체그룹의 모델링에 대한 연구와 객체그룹간의접속 절차와 객체 관리 방법을 제시한다. 이를 위해, 먼저 개방형 통신망에서 제안한 객체그룹 요구사항을 추출하고, 분산처리 환경에서의 객체그룹 관리와 객체그룹 내에 서브객체그룹을 갖는 객체그룹모델의 요구 사항과 기능을 정립하여 객체그룹의 모델을 제시했다. 제시된 객체그룹 모델을 이용하여객체그룹의 관리 서비스를 수행하기 위한 객체그룹 내의 각 구성요소들간의 상호작용에 따른 접속 절차를 나타냈고, TINA-ODL을 이용하여 객체그룹의 구성요소 객체들을 세부 설계하였다. 본 연구에서의 객체그룹 모델 제시로 인해 복잡한 분산 시스템의 개발 및 관리의 복잡성을 줄일 뿐만 아니라, 분산처리 환경에서 객체 또는 객체그룹들간의 멀티미디어 서비스 응용에 적용시킬 수 있는 기반을 구축하였다. 앞으로의 연구 방향으로는 대규모의 분산처리 환경 시스템에 적용하는 객체그룹 모델의 객체 지향설계와 서브객체그룹들의 계층화에 따른 각 구성요소들의 기능 확장과 그에 대한 기능 검증과 성능분석과 구현에 따른 많은 연구가 요구된다.

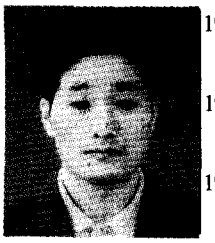
참 고 문 헌

1. Silvano Maffeis, "The Object Group Design Pattern", Dept. of Computer Science, Cornell Univ., 1996. 2
2. ISO/IEC JTCI/SC21/WG7, "Basic Reference Model of Open Distributed Processing", Parts 1-4, 1994. 7
3. Bersia, Bosco, Monione, Moiso, and Spinolo, "A CASE environment for TINA-oriented application", CSELT, 1994.
4. Pier Giorgio Bosco and Corrado Moiso, "A distributed processing model for telecounications service management", The proceedings of DSOM' 95, 1995.
5. James Runbaugh et al. "Object Oriented Modeling & Design", 1991.
6. "TINA ODL-Manual Ver. 2.3", TINA-C, 1996. 7
7. OMG, "CORBA Services : Common Object Services Specification", 1997.
8. William J. Barr, Trevor Boyod and Yuji Inoue, "The TINA Initiate", IEEE Communications Magazine, March, 1993.
9. 신영석, 오현주, "이기종 분산처리환경상에서 연결관리 객체의 정보공유", 한국정보통신학회논문지, Vol. 22 No.4, 1997.4
10. 고창록, 신영석, 김명희, 주수종 "개방형 분산 시스템에서 객체그룹 모델링에 관한 연구", 한국정보과학회 추계학술발표, 1997. 10
11. 신영석, 오현주, 고병도, 김재근, "차세대 개방타입 정보통신망 구조인 TINA 연구(1), (2)", 한국전자통신연구소, 주간기술 동향 744/5호, 1996.5.
12. 주수종, "분산처리환경에서 객체그룹 모델링 및 성능분석에 관한 연구의 최종보고서", ETRI,1997. 11

이 승 용 (Seung-Yong Lee) 정회원

1986년 : 원광대학교 전자공학과 공학사.
1990년 : 조선대학교 대학원 전산기공학과 공학석사.
1997년 : 원광대학교 대학원 컴퓨터공학과(박사수료)
1991년 ~ 현재 : 원광보건전문대학 전자계산과 부교수.
※ 관심분야는 멀티미디어 데이터베이스, 데이터 통신, 분산 시스템, 분산객체모델.

정 창 원(Chang-Won Jeong) 정회원



1993년 : 원광대학교 컴퓨터 공학사
1998년 : 원광대학교 교육대학원 컴퓨터공학과 석사과정
1998년 원광대학교 대학원 컴퓨터공학 박사과정
※ 관심분야는 분산 객체 컴퓨팅, 분산 멀티미디어 데이터베이스

신 영 석(Young-Seok Shin) 정회원

1982년 : 전북대학교 전자공학과 공학사
1984년 : 전북대학교 대학원전자공학과 공학석사
1992년 : 전북대학교 대학원 전자공학과 공학박사
1993년 ~ 1994년 : 일본 NTT 통신망연구소 객원 연구원
1984 ~ 1997년 : 한국전자통신연구원 선임연구원
1998년 ~ 현재 : 호남대학교 정보통신공학과 교수
※ 관심분야는 ATM 트래픽 제어, 객체지향 설계 및 모델링, 분산 시스템(CORBA), 개방형정보통신망구조(TINA)연구

주 수 종(Su-Chong Joo) 정회원

1986년 : 원광대학교 전자계산공학과 공학사.
1988년 : 중앙대학교 컴퓨터공학과 공학석사.
1992년 : 중앙대학교 컴퓨터공학과 공학박사.
1993년 : 미국 Univ of Massachusetts at Amherst 전기 및 컴퓨터공학과 Post-Doc.
1990년 ~ 현재 : 원광대학교 컴퓨터 공학과 교수.
※ 관심분야는 멀티미디어 데이터베이스, 분산 실시간 컴퓨팅, 시스템 최적화, 분산객체 모델.