CORBA 환경에서 실시간 서비스 지원을 위한 분산 객체의 그룹화 및 관리

신 경 민† · 김 명 희† · 주 수 종††

요 약

객체지향 기반의 초고속 통신망을 통한 분산 컴퓨팅 환경에서 어플리케이션 서비스 수행에 따른 통신망의 복잡성과 객체 관리의 용이성을 위하여 객체들의 집합체인 객체그룹의 정의가 개방형 정보 통신망 구조인 TINA에서 제안되었다. TINA 사양에 따라 본 연구실에서 연구해 온 새로운 분산 객체그룹 모델[13]을 기초로 본 논문에서는 CORBA 기반에서 서비스 요청 객체들에게 실시간으로 서비스를 지원할 수 있도록 스케줄러 객체를 포함하는 객체그룹 모델과 객체들의 관리 방안을 제안하였다. 이를 위해, 객체그룹의 정의와 실시간 서비스를 지원할 수 있는 객체그룹 모델의 제시에 따른 요구사항들을 James Rumbaugh의 모델링 방법[12]으로 객체그룹 클래스 다이어그램으로 객체그룹의 구조 및 기능적인 구성요소들을 설계하고 제시하였다. 본 논문에서는 객체그룹의 구성 객체들 중 객체그룹 관리자와 스케줄러를 IDL로 설계하였으며, 다른 객체그룹들 내의 객체간의 서비스 및 관리 접속과정은 ETD를 통해 표현하였다.

Distributed Objects' Grouping and Management for Supporting Real-time Service in CORBA Environments

Gyung-Min Shin† · Myung-Hee Kim† · Su-Chong Joo††

ABSTRACT

It is proposed in TINA, the open information telecommunication network architecture, that the definition of object group which is collection of objects provides a decrease of complex networking and a facility of object managing by service executing of application on distributed computing environment. Based on a new distributed object group model[13] we have been researched according to TINA specification, this paper proposed the object group model with the scheduler object and objects management mechanisms that can support real-time services on CORBA. To do this, we described the definition of object grouping and the requirements to suggest the object group model supporting real-time service, designed the object group structure and functional components containing in an object group using James Rumbaugh's modelling[12], and showed a class diagram of components in an object group. This paper designed IDLs of an object group manager and scheduler among the components, and finally showed the procedures of management and service interconnections between objects in the different object groups via ETD.

※ 본 논문은 1998년도 한국 학술진흥재단의 공모과제 연구비에 의하여 연구되었음.
† 준 회 원 : 원광대학교 대학원 컴퓨터공학과
†† 정 회 원 : 원광대학교 컴퓨터공학과 교수
논문접수 : 1998년 8월 25일, 심사완료 : 1999년 3월 4일

1. 서 론

오늘날 분산 컴퓨팅 환경은 사용자들에게 물리적인 위치에 상관없이 신속한 서비스를 제공하는 위치 투명성이 부각되고 있으며, 이에 따라 통신망의 관리가 더욱 어렵고 복잡해지며, 분산 어플리케이션을 설계 시 어플리케이션 개발자는 네트워크 전반에 걸친 상호작용(데이터의 상호교환에 따른 프로토콜 정의)의 이해와 학습에 많은 시간을 소비한다. 거기에 덧붙여, 많은 분산 어플리케이션 개발자들이 각기 다른 프로토콜의 개념과 규약을 사용하므로, 다른 개발자들이 작성한 분산 어플리케이션간의 호환이 불가능하다.

이러한 문제점으로 인해서, 분산 환경에서 객체(즉, 분산 어플리케이션)를 설계하고 구현하는데 따른 표준화 방법의 필요성이 대두되었고, 데이터 상호교환에 따른 어떠한 공통 구조에 대한 관심이 높아졌다. 이러한 요구로 1990년 OMG(Object Management Group)는 OMA(Object Management Architecture)를 도입하여 OMA의 추상객체모델 위에 CORBA라는 표준을 정의[6]하였고, 1992년 설립된 TINA-C(Telecommunications Information Networking Architecture-Consortium)는 네트워크 선역에 광범위의 서비스를 제공하기 위한 통신망에 기반을 둔 분산 객체 지향 구조의 설계와 기술에 대한 TINA를 제시[8]하였다. 이렇게 최근에 제시된 분산환경인 CORBA와 TINA는 특정한 인터페이스의 다중 구현을 지원하고, 클라이언트에게 서버의 위치 투명성을 제공하며, 이종의 광역 네트워크 기반으로 동일 도메인의 분산객체 환경을 정의한다는 유사성을 가지는 반면에, 객체의 개념과 분산 환경 그리고 서비스의 관점에서는 상이성을 나타낸다.

이러한 분산 환경을 기반으로 복잡한 분산 어플리케이션의 개발과 관리의 복잡성을 줄이기 위해 객체들의 집합체적인 개념의 객체그룹에 대한 정의가 필요했다. 또한 정의에 따른 객체그룹의 구성요소들에 대한 고려 시, 분산된 객체들을 효율적으로 관리하면서 서비스 수행을 요청한 객체들에게 실시간적으로 서비스를 제공하기 위한 객체들의 접속 기능에 따른 요구사항이 대두되었다. TINA에서 이미 객체그룹에 대한 정의가 제안[13]되었으나, 객체그룹에 대한 구조나 구성요소들의 상세한 기능정립이 아닌 개념적인 정의에 대한 기술이었다. 따라서, 본 논문은 TINA-C에서의 객체그룹 정의를 도입하고 현재 분산 시스템의 표준으로 사용하는 CORBA 기반에서 본 논문에서 제안하는 객체그룹의 요구사항과 객체그룹내의 구성요소들에 대한 기능을 정립하여 실시간 서비스를 지원하는 객체그룹의 모델 즉, 구조를 제시한다. 또한, 제시된 구조에서 실질적인 서비스를 수행하는 객체그룹에게 클라이언트들의 서비스 수행 요청 시 실시간 수행 지원을 위한 객체들의 관리방안과 서비스 수행 과정을 기술한다.

본 논문의 구성은 다음과 같다. 2장에서는 분산환경의 표준인 CORBA와 TINA에 대한 공통점과 차이점 그리고 현재 표준화를 위해 추진 중인 실시간 CORBA에 대해 기술한다. 3장에서는 TINA에서 제안된 객체그룹의 정의를 도입하여 CORBA 기반의 실시간 지원 객체그룹 모델링에 필요한 요구사항들을 분석하고, 객체그룹 모델의 구조를 제시한다. 4장은 실시간 지원 객체그룹 구성 요소들의 기능을 James Rumbaugh의 모델링 방법 중 객체모델로 정립하고, 제시된 구조에서 실시간 서비스 수행에 대한 객체들의 관리방안과 객체그룹내의 객체들의 관리접속 및 서비스 수행 과정을 기능 모델로 나타내며, 동일 과정을 동적 모델로 기술한다. 또한, 구성요소들 중 그룹관리자와 스케줄러 객체에 대한 IDL(Interface Definition Language) 설계를 보여준다. 마지막으로 결론과 향후 연구과제를 제시한다.

2. TINA와 CORBA

이 장에서는 본 논문의 기반 미들웨어인 CORBA와 객체그룹의 정의를 도입한 TINA에 대한 기본적인 원리와 개념상의 공통점과 차이점 그리고 최근에 대두되고 있으며 본 연구와 약간의 관련성을 갖는 실시간 CORBA에 대하여 살펴본다. OMG의 CORBA는 분산 객체지향 시스템의 설계와 개발에 대한 새로운 표준이고, TINA는 통신망에 기반을 둔 분산 객체지향 시스템에서의 상호작용(Interworking)에 대한 표준을 정의하고 있다. 또한 OMG에서는 표준 CORBA에 분산객체들의 실시간 개념의 상호작용을 지원하는 실시간 CORBA를 정의하고 있다[10].

2.1 공통점

TINA와 CORBA는 이기종(Heterogeneous) 통신망을 기반으로 한 통일된 분산 객체 지향 환경을 정의한다. 이는 분산 객체들에게 물리적인 통신망 구조를 숨

기는 블랙박스(Black box)개념의 도입으로 가능하다. 두 시스템 모두 특정 인터페이스의 다중 구현을 지원하고 클라이언트는 실제로 사용하는 구현에 대해서 알 필요가 없다. 또한 객체의 템플릿(Template)은 인터페이스의 단일 상속과 다중 상속을 모두 지원하는 객체 정의어(Object Definition Language)인 TINA ODL과 OMG IDL을 사용하여 기술한다[8].

2.2 차이점

앞 절에서 본 것처럼, TINA와 CORBA는 원리와 개념에서 매우 유사하지만, 객체와 분산 환경 그리고 서비스 측면에서 차이점을 가지고 있다.

객체에 대한 개념에서 보면, TINA는 서비스를 제공하는 객체와 객체그룹이라는 두 개의 기본적인 개체(entity)로 구성되므로, TINA ODL은 객체 그룹핑을 지원하는 반면, CORBA는 객체만을 기본 개체로 고려하므로, 객체그룹에 대한 지원이 없다. 인터페이스에서, TINA는 오퍼레이션 인터페이스와 스트림 인터페이스를 지원하지만, CORBA는 하나의 인터페이스를 지원하고 그것은 TINA의 오퍼레이션 인터페이스에 대응되며, TINA의 스트림 인터페이스와 유사한 기능을 수행하는 호출방식을 가지고 있다.

분산환경 측면에서는, CORBA는 대부분의 서비스를 ORB의 일부분이 아닌 공통 기능으로 정의하고 있고, ORB를 통한 객체간의 상호작용을 위한 요청과 응답에서의 마샬드 데이터(marshalled data)의 표준 형태를 기술하며, 보안에 관련된 내용은 공통 객체 서비스로 제공한다. 반면에, TINA에서는 TINA-DPE(Distributed Processing Environment)를 통해 사용자 어플리케이션은 서버 객체에 직접 바인드할 수 없고, 서비스를 사용하기 위해서는 보안 검사 수행 후에만 가능하며, 실행중인 객체를 관리하기 위해 부가적인 서비스들이 요구되고, 즉, 자유로운 서비스의 사용으로 PA(Provider Agent)와 제공자간에 불필요한 많은 상호작용 때문에 CORBA보다 느리며, 마샬드 데이터에 대한 표준형태에 대한 언급이 없다.

서비스 측면에서는, CORBA에서 ORB는 단지 클라이언트로부터 서버로의 요청 전송과 클라이언트로의 응답 전송 기능만을 제공하고, 객체 관리와 같은 기능들은 ORB의 공통 기능으로 지원된다. 반면에 TINA DPE는 CORBA의 공통 기능들 중에 일부 기능들만을 지원하고 있다[8].

위에서 살펴본 TINA와 CORBA의 공통점과 차이점에서 알 수 있듯이 CORBA는 분산환경에서 어플리케이션을 설계하고 구현하는 표준방법을 정의하고 있다. 일반적으로, TINA는 통신망에 대한 CORBA로 즉, 표준 CORBA의 확장 개념으로 고려되고 있다. 그러나, 아직까지 TINA는 어떠한 구현을 가지고 있지 않고 단지 기술서만을 도입하고 있으나, CORBA는 Orbix, VisiBroker 등의 다수의 구현(Implementation)들이 존재한다. 따라서, 구현이 존재하는 CORBA를 기반으로 TINA에서 기술된 객체그룹의 정의를 도입하여 분산환경에서 서비스 수행 객체그룹의 요청에 대한 클라이언트들의 실시간 지원을 가능케 하여 분산된 객체들을 관리하기 위한 객체들의 그룹화에 대한 객체그룹의 모델과 구조를 제시한다.

2.3 실시간 CORBA

CORBA는 시간 제약을 가지는 실시간 응용프로그램을 지원하기에는 제한 사항들이 많기 때문에 OMG의 RT SIG(Real-Time Special Interest Group)에서는 표준 CORBA에 실시간 지원을 위한 기능을 추가하거나 확장하는 작업을 진행[1]하고 있다. 여기에서는 실시간 CORBA를 위해서 CORBA 시스템에서 종단간 실행에서 실시간 규약의 표현과 시행을 하고 있으며, 결과적으로 1996년 표준 CORBA를 실시간 CORBA로 확장할 경우에 필요한 기술과 개념을 정의하기 위하여 실시간 CORBA White Paper를 발표[1]하였는데, 기본적인 실시간 개념인 시간 제약, 스케줄링, QoS, 성능, 동기/비동기적 분산 시스템 모델, 결함허용 등을 정의하고 있다. 또한, 실시간 CORBA에 관련된 여러 RFP(Request For Proposal)가 발표되었는데, 실시간 ORB를 위한 기술로 고정 우선순위 스케줄링과 종단간 예측성을 위한 ORB 자원관리, 융통성을 가진 통신 등을 권고[10,11]하고 있다.

세계적으로 실시간 CORBA를 연구하여 개발중인 제품으로는 워싱턴 대학에서 개발한 고성능의 표준 CORBA와 호환되는 실시간 ORB인 TAO[2]와 Rhode Island 대학과 미 해군 NRaD(US Navy Research and Development Laboratories), 그리고 MITRE사가 개발한 NRaD[16]는 다중 스레드를 지원하며, CORBA 시스템 내의 동적 종단간 시간제약을 표현하며 클라이언트나 서버가 추가 또는 삭제되고, 시간 제약이 변경될 수 있는 환경을 제공하며, 이외에 Chorus 사의 CHO-

RUS/COOL 등 여러 제품들이 있다.

위의 실시간 CORBA에 대한 연구들은 ORB를 수정하거나 확장하여 실시간을 지원[16]하고 있다. 본 연구는 ORB를 확장하지 않고, CORBA를 기반으로 하여 또 다른 응용 플랫폼으로써 객체그룹을 위치시킴으로써 클라이언트들의 서비스 수행 요청에 대한 실시간 지원이 가능토록 한다.

3. 실시간 지원 객체그룹의 정의 및 구조 제안

분산환경에서 분산 객체들의 관리의 복잡성을 해소하기 위해 객체들의 집합체인 객체그룹이라는 용어의 정의가 TINA에서 제안되었으며, 본 연구실에서는 TINA 사양에 따라 분산 객체그룹 모델[13]을 제시하였다. 이를 토대로 본 장에서는 객체들의 그룹화와 관련 개념들을 간단히 기술하고, 본 논문에서 정의하는 실시간 서비스 수행을 지원하는 객체그룹이 지원되는 분산환경과 객체그룹 모델링에 따른 객체그룹 모델의 구성요소들에 대하여 기술한다.

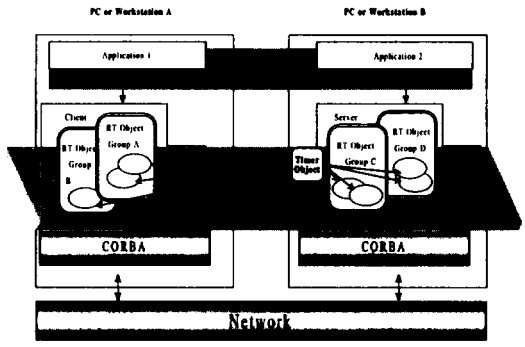
3.1 객체그룹의 정의

1993년 Bellcore에서 OSCA와 INA 구조에서 객체그룹 개념과 유사한 빌딩 블럭(building block), 1994년에는 그룹-94(group-94), 그리고 TINA 연결관리 프로토타입의 일부분으로써 객체들의 그룹관리를 위해 개발된 컴포넌트라는 개념 등이 정의되었다. TINA에서는 객체들의 집합체적이고, 관리와 서비스의 효율성을 위한 객체그룹의 정의가 제안[13]되어 있으나, 객체그룹의 완전한 모델이 제시되어 있지 않다는 것은 앞에서 언급했다. 따라서, 본 논문은 TINA에서 제안한 객체그룹의 정의를 분산객체시스템 CORBA 환경에 적용하여 CORBA 기반의 객체그룹 모델을 정의한다. 이 절 이후에 언급되는 객체그룹은 CORBA 기반에서 이루어진다.

또한, 본 논문의 실시간 서비스 수행 지원 객체그룹은 단순한 객체들의 모임이 아닌 특성을 가진 즉, 특정 서비스를 수행하기 위해 필요한 객체들의 집합으로 정의한다. 다시 말해서, 하나의 서비스를 수행하는 일련의 객체들을 그룹화한 것이며, 이에 따라 이 객체그룹은 평균적인 수행 시간을 예측 가능하게 한다.

3.2 객체그룹 지원 환경

객체그룹은 CORBA 기반 위에 또 다른 플랫폼과 같은 역할을 한다. 따라서, 객체그룹은 물리적인 이더넷이나 ATM망으로 CORBA 환경을 구성하며, 실시간 어플리케이션 개발자는 CORBA위에 올려진 객체그룹 모듈을 이용하여 원하는 분산 어플리케이션을 구현할 수 있다. 개발자가 구현한 어플리케이션의 플랫폼인 한 객체그룹에서 다른 객체그룹 플랫폼의 호출은 CORBA의 ORB를 통하여 이루어진다. 또한, 분산 객체그룹간의 서비스 수행 시 객체그룹간의 동기화는 객체그룹 외부에 타이머 객체를 두어 제어한다. 이때 사용하는 동기화 방법은 클라이언트 객체가 서버 객체에게 메시지 전송 전에 타이머 객체로부터 전역 시간을 요청하여 얻어오는 Cristian의 알고리즘이다. 다음 (그림 1)은 객체그룹을 지원하는 환경을 나타낸다.



(그림 1) 분산환경에서 실시간 서비스 수행 지원 객체그룹 플랫폼

3.3 객체그룹의 요구사항

실시간 서비스 수행 지원 객체그룹을 모델링하기 위해서는 객체그룹도 하나의 객체이고 분산 환경하에 존재하므로, 서로 다른 시스템들간 객체그룹의 분산성 그리고 객체간 접속에 따른 제한과 보안성 등을 고려하여야 한다. 다음은 실시간 서비스 수행 지원을 위한 객체그룹의 요구사항이다.

- 객체그룹은 그룹관리자(Group Manager), 객체팩토리(ObjectFactory), 스케줄러 객체(Scheduler Object), 보안 객체(Security Object), 객체정보 레포지토리(Object Information Repository), 객체(Object) 등의 구성요소가 필요하다.
- 객체그룹내의 객체나 서브객체그룹들은 동일 도메인 내의 다른 시스템으로 분산 가능하다.
- 객체그룹은 그룹관리자를 두어 객체그룹내의 객체와

서브객체그룹들을 관리하는 기능을 가진다.
- 클라이언트는 객체그룹내의 보안 관리를 위해서 그룹관리자를 통하여 객체그룹에 접속하여야 한다.
- 실시간 지원에 따른 서비스 수행 시간의 예견성(Predictability)을 위하여 그룹관리자가 객체그룹의 생성 시에 객체그룹 내 서비스 객체들의 수행으로 서비스 실행시간을 얻는 과정이 필요하다.
- 서비스 수행을 요구하는 클라이언트 객체들에 대한 우선순위 등의 실시간 서비스 지원에 따른 관리는 스케줄러 객체를 두어 해결한다.
- 객체그룹과 객체들의 정보(실시간 지원 정보도 포함)를 저장하기 위해 객체정보 레포지토리가 필요하다.
- 객체팩토리는 그룹관리자의 객체생성 요구를 수행하기 위해 객체정보 레포지토리와 상호 작용하여 객체를 생성하여야 한다.
- 보안 객체를 두어 분산 객체 시스템이 필요로 하는 보안 기능 중 인증(Authentication)과 인가(Authorization) 기능을 수행하도록 한다.
- 객체그룹내부에 서브객체그룹이 존재하며, 객체그룹의 구조와 기능이 동일하다.
- 객체그룹은 1개 이상의 서브객체그룹을 가질 수 있으며, 깊이(Depth)가 1인 서브객체그룹을 가질 수 있다. 즉, 서브객체그룹은 하위 레벨이나 서브객체그룹을 내포할 수 없다.

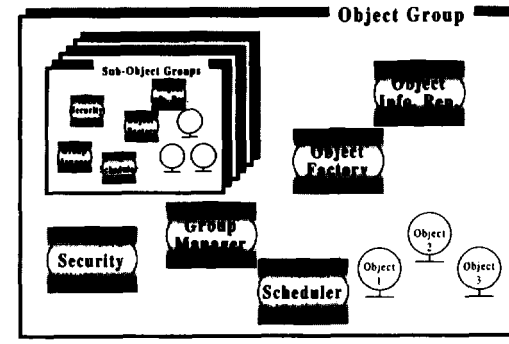
위의 요구사항 외에 실시간 지원을 위한 실시간 CORBA의 요구사항[1,3,15]은 다음과 같다.

- 절대적 시간과 상대적 시간의 표준 타입 명시가 필요하다.
- 전역 우선순위의 지원을 하여야 한다.
- 모든 CORBA 개체 서비스에서의 우선순위 큐잉 지원이 있어야 한다.
- 고정 우선순위로 스케줄 가능한 개체가 정의되어야 한다.
- 클라이언트가 요청에 관련된 시간 제한 정보에 대한 통보할 수 있는 기능과 인터페이스가 명시되어야 한다.

3.4 제시된 객체그룹의 구조

본 절에서는 앞 절의 요구사항을 바탕으로 실시간 서비스를 지원하는 객체그룹의 구조를 제시한다.

(그림 2)는 본 논문에서 제시한 객체그룹의 구조를 나타낸다. 본 객체그룹은 그룹관리자, 객체팩토리, 객체정보 레포지토리, 보안 객체, 스케줄러, 그리고 서비스를 수행하는 객체들로 구성된다.



(그림 2) 객체그룹의 구조

4. 객체그룹의 구성요소 설계

본 장에서는 앞에서 살펴본 객체그룹의 구조에 따라 James Rumbaugh의 모델링 방법 중 객체 모델로써 각 구성요소들의 기능을 분석하여 클래스 도표를 이용하여 나타내고, 객체그룹 구성요소들간의 관리방안과 상호관계를 설명하기 위해, 클라이언트로부터 객체그룹의 서비스 수행이 요청될 때(즉, 객체그룹내의 소속 객체들의 생성 후 서비스 수행)의 기본 동작을 동적 모델과 기능 모델로 나타낸다.

4.1 객체 모델

① 그룹관리자(Group Manager)

외부에서 요청한 객체그룹의 서비스 수행은 그룹관리자의 제어 하에 처리된다. 객체그룹의 서비스 수행 요청은 외부에서 해당 객체그룹의 그룹관리자를 호출함으로써 수행되는데, 이는 객체그룹의 보안과 객체들의 관리를 위해 모든 외부의 요청을 그룹관리자를 통하게 만드는 것이다. 그룹관리자는 또한 객체그룹과 연관된 객체그룹 내외의 모든 요구사항을 관리한다. 그룹관리자는 객체정보 레포지토리, 보안 객체, 스케줄러, 서비스 수행 객체들과 상호작용하여, 객체그룹내의 모든 구성요소 객체들의 관리기능을 수행한다. 즉, 객체의 생성, 삭제, 활성화, 비활성화와 서브객체그룹의 그룹관리자와 구성요소들(객체정보 레포지토리, 스케줄러 등)의 생성과 삭제, 서브객체그룹의 활성화와 비활성화

등을 수행한다. 또한 서비스 수행 객체의 생성을 위해서 객체팩토리와 상호 작용하며, 객체그룹 외부로부터의 모든 요청을 받아들이고 그에 따른 처리를 수행하며, 서비스의 실시간 처리를 위한 스케줄러의 호출과 클라이언트 객체에 대한 보안 검사 수행에 따른 보안 객체의 호출 등을 수행한다. 그룹관리자는 객체그룹이 생성되는 초기에 객체그룹내의 객체들을 실행시켜 서비스 수행시간을 스케줄러에게 넘겨줌으로써, 스케줄러를 이용한 실시간 서비스 수행 지원이 가능토록 한다.

② 객체정보 레포지토리(Object Information Repository)

객체정보 레포지토리는 객체그룹 내의 객체들과 서브객체그룹들에 대한 정보(객체그룹내의 객체 생성을 위한 템플릿과 객체의 상태 등)를 관리한다. 그룹관리자와 상호 작용하여 그룹관리자가 생성한 정보 즉, 객체그룹내의 객체들과 서브객체그룹의 그룹관리자에 대한 정보를 저장하며, 객체그룹 내 객체들의 정보를 레포지토리에서 삭제, 추가, 검색하는 기능도 가진다. 객체정보 레포지토리는 객체의 이름과 레퍼런스를 바인드시킬 수 있는 정보와 객체의 상태를 관리한다. 개별 객체들에 대한 정보(객체의 위치, 객체의 소유자, 사용허가 등)는 이미 CORBA의 객체 구현 레포지토리(Object Implementation Repository)에 저장되어 있으므로, 객체정보 레포지토리에는 객체그룹 내에 속하는 객체들에 대한 세부적인 정보만 저장된다.

③ 객체팩토리(ObjectFactory)

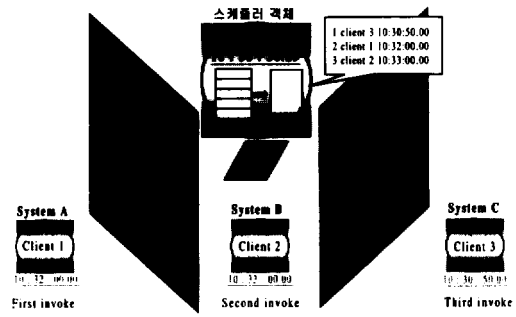
객체팩토리는 그룹관리자의 객체 생성 요구에 따라 객체정보 레포지토리에 있는 소속객체들의 정보를 이용하여 객체그룹의 모든 객체(객체 및 서브그룹관리자)를 생성한다. 이때, 객체팩토리는 BOA의 create 오퍼레이션을 사용하여 객체의 레퍼런스를 작성하고 초기화하여 처리한다.

④ 보안 객체(Security Object)

보안 객체는 객체그룹 외부에서 객체그룹 내의 객체나 서브객체그룹의 객체에 대한 서비스 요청 시 보안 검사를 하고 그룹 내에 접근할 수 있도록, 이와 관련된 작업을 수행하는 객체이다. 보안 객체는 클라이언트 객체에 대한 인증(Authentication)과 서비스 수행을 요구받은 객체에 대한 엑세스 권한이 있는지를 검사하는 인가(Authorization)에 필요한 정보들을 참조한다. 특히, 인가에 대한 수행은 접근 조정 목록(ACL, Access Control List)과 접근 규칙(Access Rule)을 참조하여 이루어진다. 보안 객체는 그룹관리자와 상호 작용하면서 이러한 기능들을 수행한다.

⑤ 스케줄러(Scheduler)

실시간 서비스 수행을 지원하기 위해, 스케줄러는 서비스 수행이 요청된 객체그룹에 대한 실행시간 등의 정보를 이용하여 서비스 수행을 스케줄링 한다. 스케줄링 시 사용되는 시간(Time)은 서비스를 요청한 객체가 위치한 시스템마다 일치하지 않을 수 있으므로 스케줄링의 공정성을 위하여 전역시간 서비스(Global Time Service)[3]을 이용한다. 또한 스케줄러가 사용하는 스케줄링 알고리즘은 짧은 마감시간을 먼저 실행하는 EDF(Earlist Deadline First)[9] 기법이다. 스케줄러는 마감시간이 짧은 클라이언트를 높은 우선순위로 설정하며 그룹관리자로부터 객체그룹 생성 시에 얻어진 객체와 서브객체그룹의 실행시간 정보와 클라이언트의 마감시간 정보를 저장한다. 그룹관리자는 클라이언트의 마감시간 정보를 스케줄러에게 전달하고, 이에 따라 스케줄러가 가지고 있는 객체나 서브객체그룹의 실행시간 정보를 이용하여 서비스 수행에 대한 마감시간으로 객체로부터 요청이 있을 경우에 스케줄링을 수행한다.



(그림 3) 스케줄러 내의 스케줄링 과정

위의 (그림 3)은 타이머 객체에 의해 동기화 과정 이후에 스케줄러 객체의 동작과정을 나타낸 것으로 전역 시간이 10:30:30.25 일 때, 각기 다른 3개의 시스템에서 요청이 있을 경우이다. 여기에서 시스템 C의 클라이언트 3이 마지막으로 서비스 수행을 요청하였더라

도, 마감시간이 가장 짧음으로 스케줄링의 EDF 기법에 의해 우선순위가 가장 높게 부여된다.

⑥ 객체(Object)

객체는 그룹관리자에 의해 관리되는 실질적인 서비스를 제공하는 객체이다. 객체그룹은 하나의 서비스를 제공하며, 서비스 요청은 서비스를 수행하는 일련의 객체들 중에서 첫 번째 객체가 그룹관리자로부터 서비스 수행 요청을 건네 받게되며, 스케줄러가 결정한 우선 순위 정보를 검색한 후에 우선 순위가 높은 요청 객체(즉, 클라이언트)에게 먼저 서비스를 제공한다. 따라서, 우선순위가 낮은 요청 객체는 우선 순위가 높은 요청 객체의 서비스 수행에 따른 초기작업이 끝날 때까지 대기하게 된다.

⑦ 서브객체그룹

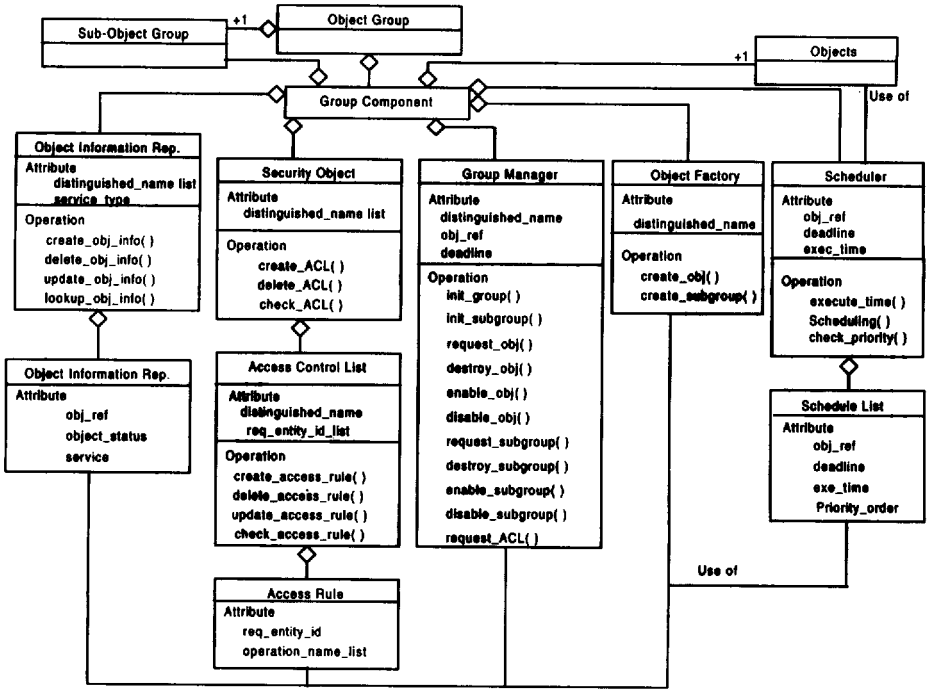
서브객체그룹은 앞에서 언급한 객체그룹의 특성을 그대로 상속받은 객체그룹이다. 따라서, 서브객체그룹을 구성하는 요소들이나 그들의 모든 기능이 객체그룹과 같다. 단지, 상위 객체그룹에서 서비스 수행 시에 필요한 서브객체그룹의 수행 요청은 상위의 그룹관리자만 서브객체그룹의 그룹관리자에게 요청할 수 있고, 서비스 수행 후 결과도 상위 그룹관리자에게 반환한다.

객체그룹 구성요소들의 기능은 위의 (그림 4)에 클래스 도표로 나타낸다. 클래스 도표에는 각 구성요소들과 구성요소에 속하는 속성과 오퍼레이션들을 기술한다. 특히, 보안 객체 내에는 ACL(Access Control List)로 접근 규칙에 따라 접근할 수 있도록 하였으며, 스케줄러 내에는 SL(Schedule List)를 포함하여 스케줄에 필요한 정보를 관리하도록 설계하였다.

4.2 기능 모델에 의한 분산 객체들의 관리 방안

본 절에서는 분산객체들을 관리하는 방안을 기능 모델링 방법으로 나타내었다. 관리방안으로는 서비스를 수행하는 객체의 위치에 따라 동일 객체그룹과 서로 다른 객체그룹으로 구분하며, 이 두 관리방안에는 각각 객체나 서브객체그룹의 생성, 삭제, 활성화, 비활성화 등이며 그룹관리자에 의해서 관리된다.

앞서 말한 분산 객체들의 관리 방안 중, 본 논문에서

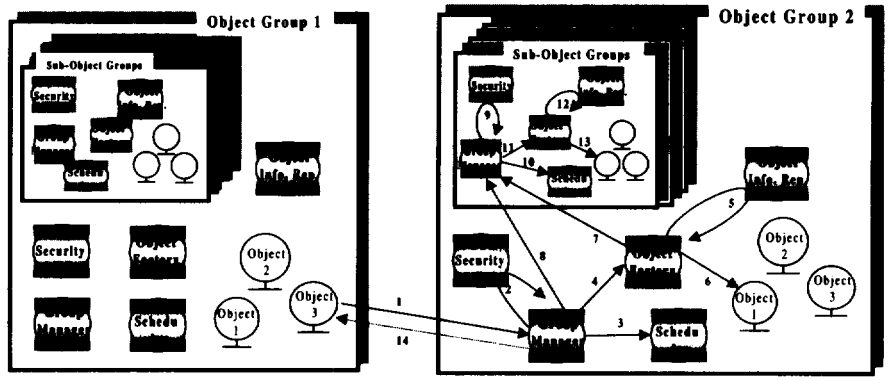


(그림 4) 객체그룹 내의 구성요소들의 클래스 도표

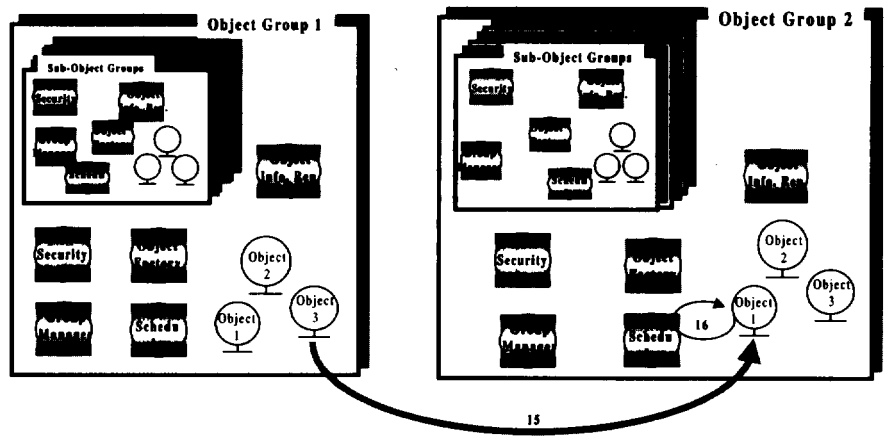
는 다른 객체그룹의 객체가 특정한 객체그룹의 서비스 수행을 요청하여 객체그룹 내의 객체 생성과 서비스 접속 등의 수행에 따른 일련의 과정을 기술한다. (그림 5)는 이러한 과정을 도식화한 것이다. 서비스 수행 요청에 앞서, 객체그룹 초기화에 의해 객체그룹의 서비스 수행 예측 시간이 이미 저장되어 있다고 전제한다.

수행과정은 먼저, 객체그룹 1(클라이언트)의 객체가 특정 서비스 수행의 요구(객체와 서브객체그룹의 생성)를 객체그룹 2의 그룹관리자에게 요청하면, 그룹관리자는 보안 객체에게 보안 검사를 의뢰하고 인증을 받으면, 스케줄러에게 요청한 클라이언트의 레퍼런스와 마감시간을 넘겨준다. 스케줄러는 그룹관리자로부터 전달받은 정보를 이용하여 클라이언트의 우선순위를 결정하고, 객체그룹이 초기화될 때 측정된 객체들의 수행 예측 시간을 유지한다. 그룹관리자는 객체팩토리에게 객체그룹의 서비스 수행 객체들과 서브객체그룹의 그룹관리자 생성을 요청한다. 객체팩토리는 객체정보 레포지토리의 템플릿을 참조하여 서비스 객체들과 서브그룹관리자를 생성하고, 서브객체그룹 내에 객체의 생성은 상위 그룹관리자가 서브그룹관리자에게 요청하여 이루어지며, 생성과정은 상위그룹과 같다. 객체팩토리 객체는 생성한 객체들의 정보를 객체정보 레포지토리에 저장하고, 생성된 객체들의 레퍼런스 중 서비스 수행 시 처음에 호출되는 첫 번째 객체의 레퍼런스를 그룹관리자에게 반환하며, 그룹관리자는 다시 서비스를 요청한 객체에게 이 레퍼런스를 반환한다. 여기까지의 과정(1-14)은 실질적인 서비스 수행에 앞선 객체그룹의 생성단계이다.

아래의 (그림 6)은 객체그룹 서비스 수행 요청에 따



(그림 5) 객체 생성 단계



(그림 6) 객체 서비스 단계

른 실질적인 어플리케이션 서비스 과정을 나타낸다. 서비스 수행을 요청한 객체는 그룹관리자로부터 반환받은 레퍼런스를 이용하여 서비스를 수행하는 객체와 접속을 한다. 객체는 서비스 수행에 앞서, 스케줄러가 가지고 있는 정보를 이용하여 클라이언트의 우선순위에 따라 서비스를 수행한다. 이때, 우선 순위가 낮은 서비스 요청 객체는 높은 우선순위 객체들의 서비스 접속이 이루어질 때까지 대기하게된다.

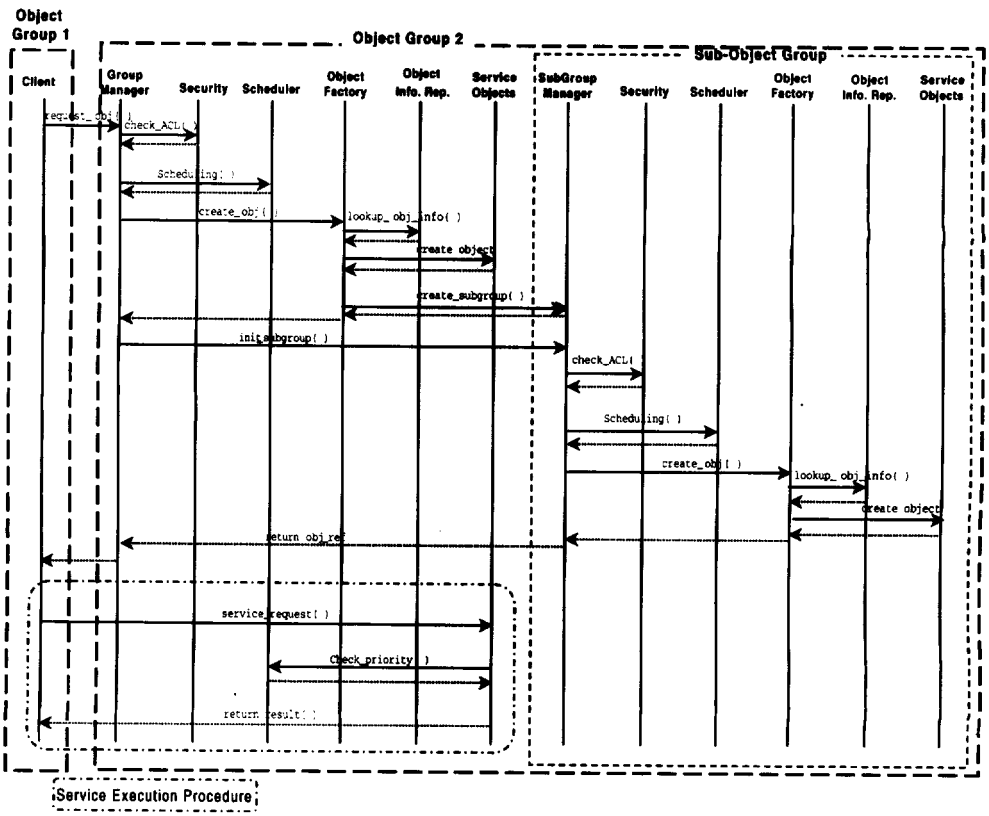
4.2 동적 모델에 의한 분산 객체들의 관리 방안

본 절은 동적 모델링 방법으로써, 앞에서 살펴본 객체그룹 내에서 객체 생성과 서비스 수행을 위한 접속 과정을 (그림 7)에서 ETD로 나타낸다.

5. 객체그룹의 IDL 설계

본 장은 객체그룹의 구성객체들 중 객체그룹의 여러 구성요소들을 관리하는 그룹관리자와 스케줄러에 대한 IDL(Interface Definition Language) 설계를 기술한다.

다음의 (그림 8)에서 그룹관리자는 객체그룹 내에서 객체들의 관리적인 측면으로 오퍼레이션을 수행한다. 즉, 객체그룹 내의 객체에게 서비스를 요청하도록 객체그룹을 초기화하는 init_group( )과 서비스 요구를 위한 request_obj( ) 오퍼레이션을, 삭제, 활성화, 비활성화를 위해서 각각 destroy_obj( ), enable_obj( ), disable_obj( ) 오퍼레이션을 제공한다. 또한 서브객체그룹을 위해서 init_subgroup( ), request_subgroup( ), destroy_subgroup( ), enable_subgroup( ), disable_subgroup( ) 오퍼레이션을 제공하며, 보안 검사를 위한 request_ACL( ) 오퍼레이션을 제공한다. 여기에서 init_group( ) 오퍼레이션이 객체그룹의 서비스 객체들을 미리 수행시켜 실시간의 예견성에 따른 객체그룹의 서비스 실행 시간을 얻어낸다.



(그림 7) 객체 생성 및 서비스 요청에 대한 ETD

```
/*group_manager.idl */
interface Group_Manager_Obj {
        typedef Object                          Object_Reference;
        typedef string                          DN;
        typedef sequence<DN>                    DN_List;
        typedef string                          deadline;
        typedef string                          Request_Entity_Id;
        typedef string                          Template_Name;
        typedef string                          Operation_Name;
        typedef sequence<Operation_Name>        Operation_Name_List;
        enum Object_Status {initial, valid, invalid};
        typedef sequence<Object_Information> Object_Information_List;
        typedef sequence<Attr>          Attr_List;
        typedef sequence<Security_Tag> Security_Tag_List;

        // 중간 생략 ...

        readonly    attribute DN                    super_grp_name;
        readonly    attribute DN                    grp_name;
        readonly    attribute DN                    self_id;
        readonly  attribute Object_Reference        obj_ref;

        void init_group(in Template_Name group_template_name);
        Object_Reference request_obj(in Template_Name obj_template_name,
                                        in Request_Entity_Id entity_id, in deadline dl);
         void destroy_obj(in Object_Reference obj_ref, in Request_Entity_Id entity_id);
         void enable_obj(in Object_Reference obj_ref);
         void disable_obj(in Object_Reference obj_ref);

        boolean init_subgroup(in Template_Name subgroup_template_name);
         boolean request_subgroup(in Template_Name group_template_name,
                                in Request_Entity_Id entity_id, in deadline dl);
         void destroy_subgroup(in Object_Reference obj_ref);
         void enable_subgroup(in Object_Reference obj_ref);
         void disable_subgroup(in Object_Reference obj_ref);
       boolean request_ACL(in Object_Reference obj_ref);
};
```

(그림 8) 그룹관리자의 IDL

```
/* Scheduler */
 interface Scheduler{
        typedef Object                          Object_Reference;
        typedef string                          Deadline;
        typedef string                          Execution_time;

        attribute Execution_time            exec_time;
        attribute Deadline                      dead_line;
        readonly  attribute Object_Reference    obj_ref;

        void execute_time(in Object_Reference obj_ref, in Execution_time exec_time);
        boolean Scheduling(in Object_Reference obj_ref, in Deadline dl);
        boolean Check_priority(in Object_Reference obj_ref);
 };
```

(그림 9) 스케줄러의 IDL

앞의 (그림 9)는 스케줄러의 IDL을 나타낸다. 객체 그룹이 생성 초기에 객체들을 실행한 후에 얻은 실행 시간을 저장하기 위한 execute_time( ) 오퍼레이션을 제공하며, 스케줄러에서 클라이언트의 레퍼런스와 마감시간 정보를 저장하고, 이를 이용하여 요청 객체들의 우선순위를 스케줄하는 Scheduling( ) 오퍼레이션과 객체가 실제적인 외부의 서비스의 요청시 우선순위에 맞는지 검사할 수 있는 Check_priority( ) 오퍼레이션을 제공한다.

6. 결 론

분산 컴퓨팅 환경은 통신망 관리의 복잡성을 해소하고 어플리케이션 개발자가 분산 어플리케이션을 설계할 때의 어려움을 해결하기 위해, 분산 환경에서 객체를 설계하고 구현하는데 TINA, CORBA 등의 표준화 방법을 사용하고 있다. 이러한 분산 환경 하에서 복잡한 분산 소프트웨어의 개발 및 관리의 복잡성을 줄이고 분산된 객체들을 효율적으로 관리하면서 서비스 수행을 요청한 객체들에게 실시간 서비스를 제공하기 위한 객체들의 접속 기능들을 지원하는 객체들의 집합체적인 객체그룹의 정의가 필요하다. TINA에서 이미 객체그룹에 대한 정의가 제안되었으나, 객체그룹에 대한 구조나 구성요소들의 상세한 기능정립이 아닌 단순한 정의에 대한 기술이었다. 따라서, 본 논문은 TINA에서 제안한 객체그룹의 정의를 도입하여 현재 분산 시스템의 표준으로 사용하는 CORBA 기반에서 객체그룹의 요구사항과 James Rumbaugh의 모델링 방법에 의해서 객체그룹내의 구성요소들에 대한 기능을 정립하고 실시간 지원 객체그룹의 모델 즉, 구조를 제시하였다. 또한, 실시간 서비스 수행을 위해 스케줄러를 객체그룹 내 구성요소로 구조화하였다. 본 논문에서는 실시간 서비스를 수행하기 위해, 서비스 수행을 요청하는 객체와 서비스를 수행하는 객체그룹간의 관리방안과 접속과정을 ETD로 구성요소들의 기능에 중점을 두어 기술하였다.

앞으로의 연구로는 기능 정립 및 설계된 객체그룹의 구조를 실시간 서비스 수행 지원 객체그룹 플랫폼을 구현하고, 구현된 객체그룹의 구조를 분산 어플리케이션에 응용하여 클라이언트(서비스 수행 요청 객체)에게 실시간적으로 응답할 수 있는지를 검증하고자 한다.

참 고 문 헌

[1] OMG Realtime Platform SIG, "Realtime CORBA A White Paper - Issue 1.0," http://www.omg.org/realtime/real-time_whitepapers.html, 1996.

[2] John. K. Black, et al. "Real-time Method Invocations in Distributed Environments," University of Rhode Island Department of Computer Science and Statics, Technical Reports TR95-244, 1995.

[3] Victor Fay Wolfe, et al., "Expressing and Enforcing Timing Constraints in a Dynamic Real-Time CORBA System," http://www.cs.uri.edu/rtsorac/publication.ht- ml, 1997.

[4] Victor Fay Wolfe, et al., "Real-Time CORBA," In proceedings of the third IEEE Real-time Technology and Applications Symposium, 1997.

[5] G. P. A. Fernandes and I. A. Utting, "An Architecture for Scheduling of Services in a Distributed System"

[6] OMG, "The Common Object Request Broker : Architecture and Specification revision 2.2," http://www.omg.org/corba/corbaCB.htm, 1998.

[7] OMG, "CORBA Services : Common Object Services Specification," http://www.omg.org/corba/sectran1.htm, 1997.

[8] Nguyen Duy Hoa, "Distributed Object Computing with TINA and CORBA," Technical Report Nr. 97/7, http://nenya.ms.mff.cuni.cz/thegroup/, 1997.

[9] "Scheduling in Real-Time Systems," http://www.realtime-os.com/sched_o3.html, 1996.

[10] OMG, "Realtime CORBA 1.0 RFP," OMG Document : orbos/97-09-31, 1997.

[11] OMG TC, "Realtime CORBA Extensions : Joint Initial Submission," OMG TC, Document orbos/98-01-09, 1998.

[12] James Rumbaugh, et al. "Object-Oriented Modeling and Design," Prentice Hall, 1991.

[13] 주수종, 한국전자통신연구원, "분산처리환경에서 객체그룹 모델링 및 성능분석에 관한 연구" 최종보고서, 1997.

[14] 고창록 외 3명, "개방형 분산시스템에서 객체그룹

모델링에 관한 연구", 한국정보과학회 학술지, 1997.
[15] 신경민 외 2명, "CORBA 기반 분산환경에서 실시간 서비스 지원을 위한 객체그룹화에 대한 연구", 한국정보과학회, 1998.
[16] 정창균 외 2명, "실시간 CORBA", 한국정보처리학회지, 제5권 4호, pp.88-98, 1998.

신 경 민

e-mail : gmshin@wonnms.wonkwang.ac.kr
1997년 원광대학교 공과대학 컴퓨터공학과(공학사)
1999년 원광대학교 대학원 컴퓨터공학과(공학석사)
관심분야 : 분산 실시간 컴퓨팅, 분산 객체 시스템, 객체지향 데이터베이스

김 명 회

e-mail : hee@wonnms.wonkwang.ac.kr
1993년 원광대학교 컴퓨터공학과(공학사)
1996년 원광대학교 대학원 컴퓨터공학과(공학석사)
1996년~현재 원광대학교 대학원 컴퓨터공학과 박사과정 중
관심분야 : 멀티미디어 데이터베이스, 분산 실시간 컴퓨팅, 시스템 최적화, 운영체제

주 수 종

e-mail : scjoo@wonnms.wonkwang.ac.kr
1986년 원광대학교 전자계산공학과(공학사)
1988년 중앙대학교 대학원 전자계산학과(공학석사)
1992년 중앙대학교 대학원 전자계산학과(공학박사)
1993년~1994년 미국 Univ. of Massachusetts at Amherst, 전기 및 컴퓨터공학과, Post-Doc.
1990년~현재 원광대학교 공과대학 컴퓨터공학과 부교수
관심분야 : 멀티미디어 데이터베이스, 분산 실시간 컴퓨팅, 분산객체모델, 시스템 최적화