분산 실시간 서비스를 위한 TMO 객체그룹 모델의 구축

(A Construction of TMO Object Group Model for Distributed Real-Time Services)

신 창 선 † 김 명 희 †† 주 수 종 †††
(Chang-Sun Shin) (Myung-Hee Kim) (Su-Chong Joo)

요 약 본 논문에서는 분산 객체 컴퓨팅 환경에서 보장된 실시간 서비스를 지원하는 TMO 객체그룹(TMO Object Group) 모델을 설계·구축하고, 우리 모델의 정확한 분산 실시간 서비스 수행능력을 검증한다.

우리가 제안한 TMO 객체그룹은 TINA(Telecommunications Information Networking Architecture)의 객체그룹 개념을 기반으로, 실시간 특성을 가지는 TMO(Time-triggered Message-triggered Object) 객체들과 객체그룹 내의 객체 관리 서비스(Object Management Service), 실시간 스케쥴링 서비스(Real-Time Scheduling Service)를 지원하는 컴포넌트들로 구성된다. 또한, TMO 객체는 분산 시스템에 비중복 또는 중복으로 존재할 수 있다. 본 모델은 특정 ORB나 운영체제들의 제약 없이 COTS(Commercial Off-The-Shelf) 미들웨어 상에서 보장된 분산 실시간 서비스를 수행한다.

TMO 객체그룹을 구축하기 위해 TMO 객체의 개념과 TMO 객체그룹의 구조를 정의하였고, 객체그룹 내의 컴포넌트들의 기능과 그들간의 상호작용을 설계·구현하였다. TMO 객체그룹은 객체 관리 서비스와 실시간 스케쥴링 서비스 지원을 위해 동적바인더객체(Dynamic Binder Object)와 스케쥴러객체(Scheduler Object)를 각각 가진다. 동적바인더객체는 클라이언트들의 요청에 대해 중복 TMO 객체 중 적정 객체를 선정하는 동적 바인딩 서비스를 지원하고, 스케쥴러객체는 클라이언트들의 서비스 요청에 대해 TMO 객체가 수행해야 할 작업들의 우선순위를 정하는 실시간 스케쥴링 서비스를 지원한다. TMO 객체그룹의 수행 검증을 위해 이미 연구된 알고리즘을 확장한 동적 바인딩 서비스를 위한 바인딩 우선순위(Binding Priority) 알고리즘과 실시간 스케쥴링 서비스를 위한 EDF(Earliest Deadline First) 알고리즘을 적용하여 동적바인더객체와 스케쥴러객체를 구현했다. 마지막으로 수치 분석을 통해 TMO 객체그룹이 비중복/중복 TMO 객체의 동적 바인딩 서비스와 클라이언트들의 요청을 받는 임의의 TMO 객체에서 실시간 스케쥴링 서비스를 지원하는지 검증했다.

키워드 : 분산 실시간 서비스, 객체 관리, TMO, 객체그룹, 중복객체, 동적바인딩, 실시간 스케쥴링

Abstract In this paper, we design and construct a TMO object group that provides the guaranteed real-time services in the distributed object computing environments, and verify execution power of its model for the correct distributed real-time services.

The TMO object group we suggested is based on TINA's object group concept. This model consists of TMO objects having real-time properties and some components that support the object management service and the real-time scheduling service in the TMO object group. Also TMO objects can be duplicated or non-duplicated on distributed systems. Our model can execute the guaranteed distributed real-time service on COTS middlewares without restricting the specially ORB or the operating system. For achieving goals of our model, we defined the concepts of the TMO object and

· 본 연구는 정보통신부 정보통신연구진흥원에서 지원하는 정보통신기초기술연구지원사업(C1-2002-105-007-3)으로 수행되었음.
† 학생회원 : 원광대학교 전기·전자 및 정보공학부
csshin@wonkwang.ac.kr
†† 비 회 원 : 원광 디지털대학교 게임소프트웨어학과 교수
hee@wonkwang.ac.kr
††† 종신회원 : 원광대학교 전기·전자 및 정보공학부 교수
scjoo@wonkwang.ac.kr
논문접수 : 2002년 6월 24일
심사완료 : 2003년 3월 4일

the structure of the TMO object group. Also we designed and implemented the functions and interactions of components in the object group. The TMO object group includes the Dynamic Binder object and the Scheduler object for supporting the object management service and the real-time scheduling service, respectively. The Dynamic Binder object supports the dynamic binding service that selects the appropriate one out of the duplicated TMO objects for the clients' request. And the Scheduler object supports the real-time scheduling service that determines the priority of tasks executed by an arbitrary TMO object for the clients' service requests. And then, in order to verify the executions of our model, we implemented the Dynamic Binder object and the Scheduler object adopting the binding priority algorithm for the dynamic binding service and the EDF algorithm for the real-time scheduling service from extending the existing known algorithms. Finally, from the numerical analyzed results we are shown, we verified whether our TMO object group model could support dynamic binding service for duplicated or non-duplicated TMO objects, also real-time scheduling service for an arbitrary TMO object requested from clients.

Key words : Distributed Real-Time Service, Object Management, TMO, Object Group, Duplicated Object, Dynamic Binding, Real-Time Scheduling.

1. 서 론

현대의 컴퓨팅 환경은 분산된 객체에 실시간 서비스에 대한 요구가 증가하는 분산 실시간 객체 컴퓨팅 환경으로 변화하고 있다[1,2]. 이전의 실시간 어플리케이션은 단일 프로세스 환경에서 단순한 실시간 제약을 가지고 동작했다. 그러나, 현재의 항공전자공학이나 분산 방위시스템 등을 위한 실시간 어플리케이션은 분산 환경에서 복잡한 실시간 요구사항을 만족해야 한다. 즉, 분산 실시간 어플리케이션은 논리적으로 서로 연관된 하나 이상의 분산 객체들로 구성되며, 분산된 객체의 상호작용으로 클라이언트의 요청에 대한 정확한 결과의 산출과 실시간 요구사항을 만족하기 위한 운용의 적시성(timeliness)을 요구한다[3,4,5].

분산 어플리케이션에서 개별 객체들을 효과적으로 관리하기 위한 연구로 TINA-C (Telecommunications Information Networking Architecture-Consortium)에서는 TINA를 정의하였다[6,20]. TINA는 서비스나 어플리케이션을 위해 다수의 컴퓨팅 노드에 위치하는 연관된 객체들을 하나의 논리적인 집합으로 서비스를 수행한다. 이러한 객체 집합을 관리하기 위한 구조화 메카니즘으로 객체그룹(Object Group)을 정의하여 서비스 수행을 위한 각각의 개별 객체 대신에 하나의 단위로서 객체들을 관리하고자 했다. 객체그룹은 캡슐화 단위이고, 분산 객체들로 구성되는 단위객체이며, 적어도 객체그룹에 있는 객체들의 관리에 대한 책임자로서 한 객체를 포함한다. 또한, 객체그룹은 하위 객체그룹을 가지며 계층적인 객체들의 비결합(disjoint) 집합으로 구성된다. 우리 연구에서는 분산 객체들의 관리를 위해 TINA의 객체그룹 개념을 도입했다. 그러나, TINA에서의 객체그룹은 분산 환경을 위한 구성요소와 명세만을 정의했고, 또한 분산 환경에서 실시간 서비스를 지원하기 위한 방안도 정의하지 않았다.

TINA의 제약사항을 극복한 연구로 OMG(Object Management Group)는 분산 환경을 위한 표준 소프트웨어 규정인 CORBA(Common Object Request Broker Architecture)를 정의하였다[7]. CORBA는 분산 어플리케이션 구현을 위한 분산 객체의 유연성(flexibility), 확장성(scalability), 재사용성(reusability) 등을 개선하기 위해 사용되었지만, 분산 실시간 어플리케이션을 위한 실시간 요구사항은 지원하지 못했다. OMG는 실시간 요구사항의 지원 문제를 해결하기 위해 RT-SIG(Real-Time Special Interest Group)를 설립하여 실시간 확장성을 가지는 실시간 CORBA(Real-Time CORBA) 명세[8]를 개발했다. 그러나, 실시간 CORBA에서는 실시간 요구사항을 지원하기 위해, CORBA의 핵심이 되는 ORB(Object Request Broker)를 수정하거나 확장하여 특정 시스템이나 운영환경에 의존적인 분산 실시간 환경을 만들었다.

TINA와 CORBA의 구현과 실시간 서비스 지원 문제를 해결하기 위해, CORBA를 기반으로 독립적인 플랫폼 상에서 객체 관리 서비스와 실시간 서비스를 지원하는 실시간 객체그룹(Real-Time Object Group, RTOG)에 대한 연구가 진행되었다[2,9,20,22]. 실시간 객체그룹은 표준 CORBA 상에 ORB의 수정 없이 TINA의 객체그룹 개념을 기반으로 분산 실시간 서비스를 제공하는 모델로, 분산된 객체들의 체계적인 관리와 실시간 서비스 지원을 위해 개별 객체들에게 과중되는 관리절차

를 간소화하였다. 그러나, 실시간 객체그룹은 동일한 서비스를 제공하는 중복 객체 중 서비스 수행을 위해 적정 조건을 갖는 객체를 선정하는 동적 바인딩 서비스를 지원하지 못했다.

실시간 특성을 만족하는 객체 모델에 관한 연구로 UCI의 DREAM 연구소에서는 TMO(Time-triggered Message-triggered Object) 객체 스키마를 제안했다[11,12,13]. TMO 객체는 실시간 특성을 자체적으로 갖는 객체로 기존 서비스 객체의 구조를 확장한 메카니즘이다. 그러나, TMO 객체는 객체에 할당된 작업들에 대한 실시간 스케줄링 서비스나 중복 TMO 객체 중 적정 객체에 바인딩하기 위해 동적으로 객체를 선정하는 서비스를 지원하지 못했다.

위의 연구들에서 나타난 제약사항을 극복하기 위해 본 논문에서는 분산된 TMO 객체들을 COTS(Commercial Off-The-Shelf) 미들웨어 상에서 그룹으로 관리하고, 보장된 분산 실시간 서비스 수행능력을 제공하는 TMO 객체그룹(TMO Object Group) 모델을 제안한다. TMO 객체그룹을 구축하기 위해 TMO 객체의 개념과 TMO 객체그룹의 구조를 정의하고, 객체그룹 내의 컴포넌트들의 기능과 그들간의 상호작용을 설계·구현한다. 본 모델의 구축을 위해 클라이언트들의 요청에 대해 중복 TMO 객체 중 적정 객체를 선정하는 동적 바인딩 서비스와 임의의 서비스 TMO 객체에 대해 클라이언트들의 요청을 처리하기 위한 실시간 스케줄링 서비스 관점을 고려하여 객체그룹 내 동적바인더객체와 스케줄러객체를 설계하고 구현하였다. TMO 객체그룹의 수행 검증을 위해 이미 연구된 알고리즘을 확장하여 동적 바인딩 서비스를 위한 바인딩 우선순위(Binding Priority) 알고리즘과 실시간 서비스를 위한 EDF(Earliest Deadline First) 알고리즘을 동적바인더객체와 스케줄러객체에 적용했다. 마지막으로 수치 분석을 통하여 TMO 객체그룹이 비중복/중복 TMO 객체에 대한 동적 바인딩 서비스와 클라이언트들의 요청을 받는 임의의 TMO 객체에서 실시간 스케줄링 서비스를 정확하게 지원하는지 검증한다.

본 논문의 구성은 다음과 같다. 2장에서 TMO 객체의 개념과 TMO 객체그룹 모델의 구조를 정의하고, 객체그룹 내 컴포넌트의 기능과 상호작용을 설계한다. 3장과 4장에서 TMO 객체그룹의 분산 실시간 서비스를 위한 동적바인더객체와 스케줄러객체를 바인딩 우선순위 알고리즘과 EDF 스케줄링 알고리즘을 적용하여 구현하고, 위 서비스들의 수행결과를 수치 분석하여 우리 모델이 비중복/중복 객체에 대한 동적 바인딩 서비스와 클라이언트로부터의 서비스 요청들에 대한 실시간 스케줄링 서비스를 지원하는지 검증한다. 마지막으로, 본 논문에 기술된 연구에 대한 결론과 향후 연구내용 및 방향을 5장에서 기술한다.

2. TMO 객체그룹 모델

본 장에서는 TINA의 객체그룹 정의를 기반으로 실시간 객체인 TMO 객체들을 COTS 미들웨어 상에서 하나의 논리적인 그룹으로 관리하고, 이들간의 실시간 요구사항을 만족하도록 하는 TMO 객체그룹 모델을 정의하고, TMO 객체그룹 내의 실시간 서비스를 책임지는 TMO 객체의 개념과 제안하는 모델의 구조 및 그룹 내부 컴포넌트들의 기능과 그들간의 상호작용을 기술한다.

2.1 TMO 객체의 개념

TMO 객체는 기존 객체 모델에 대한 확장으로, 실시간 특성을 자체적으로 가지는 실시간 객체이다. 기존 객체는 클라이언트의 서비스 요청 메시지에 의해서만 동작하고 자체적인 실시간 특성을 가지고 있지 못했다. 그러나, TMO 객체는 기존 객체 모델이 가지고 있는 클라이언트의 서비스 요청에 의해 동작되는 메소드인 SvM(Service Method)과 함께 객체에 정의된 시간에 자발적으로 동작하는 새로운 메소드인 SpM(Spontaneous Method)을 추가적으로 가진다. 그리고, 분산된 TMO 객체들은 원격 메소드 호출(remote method call)을 통하여 상호 동작한다. TMO 객체는 UCI의 DREAM 연구소에서 개발되었으며, 기본 구조는 그림 1과 같고 다음의 5부분으로 구성된다[11,12,13].

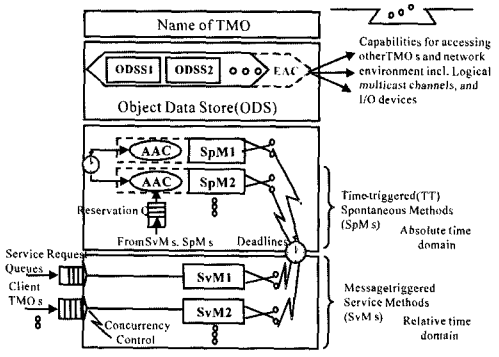

그림 1 TMO 객체 구조

① ODS(Object Data Store) : 객체의 상태나 속성 정보 저장을 위한 공통 정보 저장소.

② EAC(Environment Access Capability) : 원격 객체

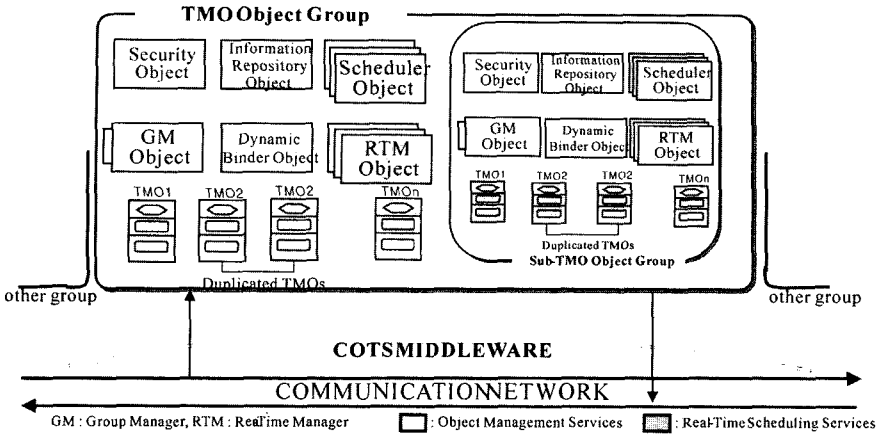

그림 2 TMO 객체그룹 모델의 구조

메소드, 통신채널, I/O 장치 인터페이스에 호출 경로를 제공하는 게이트(gate) 리스트.

③ AAC(Autonomous Activation Condition) : SpM의 주기적인 동작을 위한 시간 정의.

④ SpM(Spontaneous Method) : 주기적으로 실시간 동작하는 시간 트리거 메소드.

⑤ SvM(Service Method) : 외부의 서비스 요청에 응답하는 메시지 트리거 메소드.

ODS는 SpM와 SvM에 의해 접근될 수 있는 공통 정보 저장소로 SpM과 SvM이 동시에 접근할 수 없으며, SpM이 SvM보다 ODS 접근에 대해 우선권을 갖는다. 즉, TMO 객체의 메소드는 ODS에 접근 시 BCC(Basic Concurrency Constraint)를 준수한다. EAC는 다른 TMO 객체에 접근하기 위해 네트워크 상의 통신 채널이나 I/O 장치의 인터페이스 호출을 책임진다. SpM과 SvM은 메소드들의 리스트로 TMO 객체는 여러 개의 SpM과 SvM을 가질 수 있다. TMO 객체는 SpM의 처음에 위치하는 AAC에 SpM의 동작 시간을 명세하여 기존 객체와 구별되는 실시간 객체로 구현한다.

TMO 객체 모델은 실시간 요구사항을 만족하는 실시간 객체 모델로 군사 어플리케이션이나 운송 어플리케이션과 같은 실시간 시뮬레이션 분야에 적용되어 활발한 연구가 진행 중에 있다[13,14]. 그러나, TMO 객체 모델은 객체 접근에 대한 보안 검사와 동일한 서비스를 수행하는 중복된 TMO 객체들 중 서비스 수행을 위한 적정 TMO 객체를 선정하는 동적 바인딩 방안 및 다수의 클라이언트로부터 요청되는 서비스에 대한 실시간 스케쥴링 방안을 제공하지 않는다. 이러한 제약사항을 TMO 객체그룹 모델에서 해결한다.

2.2 TMO 객체그룹의 구조

본 논문에 제안하는 TMO 객체그룹은 TINA에서 제안한 객체그룹 정의를 COTS 미들웨어 상에서 적용할 수 있도록 한 모델로 단순한 객체들의 모임이 아닌 특정 실시간 서비스를 수행하기 위한 일련 객체들의 집합으로 논리적인 객체그룹이다. 이러한 TMO 객체그룹 설계를 위한 요구사항으로, TMO 객체그룹의 객체 관리 서비스를 지원하기 위해 그룹관리자객체(Group Manager Object, GM), 보안객체(Security Object), 정보저장소객체(Information Repository Object), 동적바인더객체(Dynamic Binder Object)를 포함하며, 실시간 스케쥴링 서비스를 지원하기 위해 TMO 객체, 실시간관리자객체(Real-Time Manager Object, RTM), 스케쥴러객체(Scheduler Object)를 가진다. TMO 객체는 동일한 서비스를 제공하는 중복 TMO 객체(Duplicated TMO objects)로 존재할 수 있다. 또한, 하위 TMO 객체그룹(Sub-TMO Object Group)을 포함할 수 있으며, 하부 TMO 객체그룹은 상위 TMO 객체그룹의 구조와 기능적 절차, 접속 순서가 같다. 앞서 제시한 요구사항을 바탕으로 분산 실시간 서비스를 지원하기 위한 TMO 객체그룹의 구조는 그림 2와 같다.

본 논문에서 제안하는 분산 실시간 서비스를 지원하는 TMO 객체그룹 모델을 구축하기 위해 다음 사항들을 가정한다.

1) TMO 객체그룹은 논리적인 객체그룹이다. TMO 객체들은 논리적인 도메인 상에 물리적 네트워크로 연결된 개별 시스템에 위치한다.

2) 우리 모델에서 TMO 객체그룹은 비중복 또는 중

복 TMO 객체를 포함한다. 중복 TMO 객체는 동일한 속성을 갖는 둘 이상의 객체를 의미한다. 클라이언트는 중복 TMO 객체 중, 하나를 선정하여 바인딩해야 하기 때문에 동적 바인딩 서비스가 필요하다.

3) 우리 모델은 객체 관리 서비스와 실시간 스케쥴링 서비스를 제공한다. 객체 관리 서비스는 객체그룹 내의 객체들의 관리를 위한 기반 서비스와 동적 바인딩 서비스를 지원하고, 실시간 스케쥴링 서비스는 클라이언트들의 서비스 요청에 대해 우선순위를 기반으로 분산 실시간 스케쥴링 서비스를 지원한다.
4) TMO 객체그룹에서 수행하는 클라이언트의 서비스 요청은 주기적으로 발생하며, 마감시간은 클라이언트가 서비스 요청 시 계산된다[14].

TMO 객체그룹의 객체 관리 서비스를 지원하기 위한 컴포넌트로, 그룹관리자객체는 TMO 객체그룹 내의 객체들에 대한 전반적인 관리를 책임진다. TMO 객체를 특정 TMO 객체그룹에 소속시키거나 탈퇴시키는 기능을 가지며, 보안객체에게 클라이언트의 서비스 요청에 대해 객체 접근권한 검사를 요청하고, 접근이 허가된 요청에 대해 정보저장소객체에 서비스를 수행 할 TMO 객체 레퍼런스를 요청한다. 정보저장소객체로부터 반환된 TMO 객체 레퍼런스를 클라이언트에 반환한다.

보안객체는 TMO 객체그룹에서 발생하는 클라이언트의 서비스 요청에 대한 그룹 내의 서비스 TMO 객체 접근 권한을 검사한다. 접근 권한 검사는 접근제어리스트(Access Control List, ACL)를 참조하여 이루어지며, 접근제어리스트는 클라이언트명과 서비스명을 속성으로 가진다. 그룹관리자객체로부터 새로운 TMO 객체의 그룹 소속이나 탈퇴요청을 받으면 접근제어리스트를 갱신한다.

정보저장소객체는 서비스를 수행할 TMO 객체들의 객체리스트를 가지며, 객체리스트는 서비스명과 TMO 객체 레퍼런스를 속성으로 가진다. 그룹관리자객체로부터 새로운 TMO 객체의 그룹 소속이나 탈퇴 요청을 받아 객체리스트의 정보를 갱신한다. 서비스를 수행할 TMO 객체가 비중복되어 존재할 경우 해당 객체의 레퍼런스를 정보저장소객체가 반환하며, 동일한 속성을 갖는 TMO 객체가 중복되어 존재할 경우 동적바인더객체에 부하를 고려한 적정 조건의 TMO 객체를 선정하도록 요청한다.

동적바인더객체에는 정보저장소객체에 존재하는 중복 TMO 객체에 대해 각각의 바인딩 우선순위 리스트를 가진다. 서비스를 수행할 TMO 객체가 위치하는 서버 시스템의 부하정보와 클라이언트의 서비스 요청에 대한 결과 반환 시간인 요청마감시간 정보를 가지고 적정 조건의 TMO 객체 선정 작업을 수행한다. 객체 선정 작업은 바인딩 우선순위 알고리즘을 적용하여 이루어지며, 성능이 우수한 다른 알고리즘으로 대체 가능하다.

실시간 스케쥴링 서비스를 지원하는 컴포넌트로, 2.1절에서 설명한 TMO 객체와 실시간관리자객체, 스케쥴러객체를 포함한다. 실시간관리자객체는 TMO 객체로부터 클라이언트의 요청마감시간 정보를 전달받아 계산된 서비스마감시간으로 스케쥴러객체에 실시간 스케쥴링을 요청한다. 클라이언트의 서비스 요청에서 결과를 반환받기 위한 단계로 서비스 요청단계, 서비스 처리단계, 결과 반환단계로 수행되며, 다섯 가지의 실시간 제약조건을 가진다. 실시간 제약조건은 2.3절에서 세부적으로 정의한다.

마지막으로, 스케쥴러객체는 작업 우선순위 리스트를 가지며, 클라이언트들의 서비스 요청에 대한 작업 우선순위를 클라이언트 정보와 서비스마감시간 정보를 이용하여 실시간 스케쥴링 한다. 스케쥴러객체가 실시간 스케쥴링 서비스를 지원하는지 보이기 위해 EDF 알고리즘을 적용한다. 동적바인더객체와 같이 스케쥴러객체의 EDF 알고리즘도 성능이 우수한 다른 알고리즘으로 대체 가능하다. 서비스마감시간이 가장 짧은 요청에 가장 높은 우선순위를 부여하며, 스케쥴링 후 산출되는 결과를 기반으로 클라이언트의 요청에 대한 마감시간 위반을 예측할 수 있다.

2.3 실시간 제약조건

TMO 객체그룹의 객체 관리 서비스와 실시간 스케쥴링 서비스를 지원하기 위해, 다음의 다섯 가지 시간 제약조건을 정의한다. 클라이언트의 TMO 객체나 그룹관리자 객체에 대한 서비스 요청시간(Invocation Time, IT), TMO 객체에서 본 서비스를 수행하는 서비스 수행시간(Service Time, ST), 클라이언트의 서비스 요청에 대한 결과 반환 시간인 요청마감시간(Request Deadline, RD), 실시간관리자에서 계산된 TMO 객체의 서비스마감시간(Service Deadline, SD), 클라이언트가 TMO 객체에게 서비스 요청 메시지를 보내는데 필요한 전송시간(Transfer Time, TT)을 가진다. 요청시간은 클라이언트의 서비스 요청시간(Client Invocation Time, CIT), 그룹관리자객체가 바인딩 서비스 요청을 받은 시간(GM Invoked Time, GMIT), TMO 객체가 실시간 서비스 요청을 받은 시간(Service TMO Invoked Time, SIT)으로 나눈다. 본 논문에서 사용되는 시간들은 전역시간

임을 전제로 하며, 분산되어 있는 각 시스템들이 전역시간을 획득하는 방법에 대해서는 고려하지 않는다. 그림 3은 시간 제약조건의 정의를 보여준다.

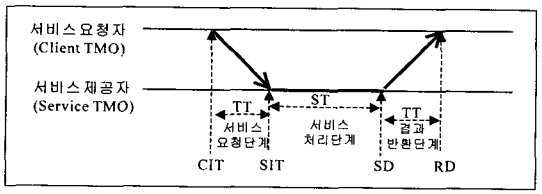

그림 3 시간 제약조건의 time-window

위에서 정의한 시간 제약조건으로 다음과 같은 식을 얻을 수 있다.

$$TT = SIT - CIT, \qquad ST = SD - SIT$$
$$RD \geq CIT + ST + (2 * TT) + slacktime \qquad (1)$$
$$SD \leq RD - TT$$

slacktime : 적정 요청마감시간을 결정하기 위한 상수값

2.4 TMO 객체그룹의 장점

본 논문에서 제안한 TMO 객체그룹의 장점은 다음과 같다. 첫째, 분산 실시간 환경에 대한 객체 관리 서비스와 실시간 스케쥴링 서비스를 제공한다. 서비스 객체에 대한 접근권한 검사와 중복 객체에 대한 동적 바인딩 서비스 및 서비스 요청에 대한 작업 우선순위 기반 실시간 스케쥴링 서비스를 지원한다. 둘째, 실시간 객체인 TMO 객체 모델을 적용하여 실시간 어플리케이션의 서비스 수행 능력을 향상시킬 수 있다. 셋째, TMO 객체 모델에 대해 작업 우선순위 스케쥴링을 기반으로 추가적인 실시간 서비스를 제공한다. 넷째, 광범위한 적용분야를 가지는 실시간 어플리케이션의 구현 비용을 줄일 수 있다. 다섯째, 컴포넌트로 TMO 객체그룹 구성요소들을 구현하여 실시간 어플리케이션 확장과 유지관리의 편리성을 제공한다. 여섯째, 실시간 어플리케이션을 위한 예측 가능한 타이밍 수행능력의 평가 방법을 제공한다.

3. 객체 관리 서비스 지원

본 장에서는 TMO 객체그룹이 제공하는 객체 관리 서비스 중 동적 바인딩 서비스를 중심으로 기술한다. TMO 객체그룹의 객체 관리 서비스는 비중복 또는 중복된 TMO 객체에 대해 적정 조건을 갖는 TMO 객체를 선정하는 바인딩 우선순위 알고리즘을 수행하여 동적 바인딩 서비스를 지원한다. 동적 바인딩 서비스의 수행결과로서 바인딩 우선순위 리스트를 참조하여 서비스

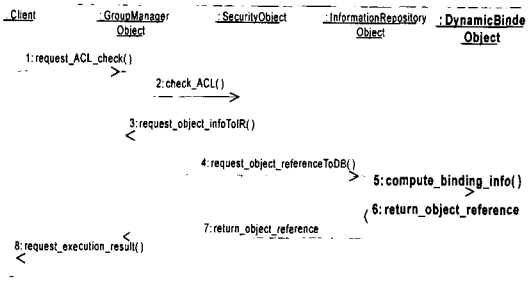

그림 4 TMO 객체그룹의 객체 관리 서비스 절차

를 수행할 적정 TMO 객체 레퍼런스를 획득하게 된다.

TMO 객체그룹의 동적 바이딩 서비스 절차는 다음과 같다. 클라이언트는 네임 서버를 통하여 그룹관리자객체의 레퍼런스를 획득한 후, 그룹관리자객체에게 바인딩할 임의의 TMO 객체의 레퍼런스를 요청한다. 서비스 요청 시 클라이언트는 클라이언트명(client_name), 서비스명(service_name), 요청마감시간(request_deadline), 클라이언트 요청시간(client_invocation_time) 정보를 그룹관리자객체에게 제공한다. 그룹관리자객체는 보안객체에게 클라이언트가 요청한 서비스에 대해 객체 접근 권한 검사를 요청하고, 접근 권한이 인정되면 정보저장소객체에 TMO 객체 레퍼런스를 요청한다. 정보저장소객체는 객체리스트를 검색하여 TMO 객체 레퍼런스를 반환하며, 만일 레퍼런스가 중복되어 존재할 경우 동적바인더 객체에 적정 조건을 갖는 객체를 선정하도록 요청한다. 동적바인더객체는 중복된 TMO 객체들 중에서 각각의 TMO 객체가 존재하는 서버 시스템의 부하정보와 클라이언트가 서비스 요청 시 제공하는 요청마감시간 정보를 2.3절의 실시간 제약조건에 적용하여 서비스 수행 우선순위를 예측, 적정 TMO 객체의 레퍼런스를 선정한다. 마지막으로 그룹관리자객체는 TMO 객체 레퍼런스를 클라이언트에게 반환한다. 아래 그림 4는 TMO 객체그룹 내의 동적 바인딩 서비스를 위한 컴포넌트 사이의 상호작용을 보여주는 ETD(Event Trace Diagram)이다. 본 장에서는 그림 4의 선정 절차에 준하여 비중복 TMO 객체의 선정과 동일한 서비스를 수행하는 중복 TMO 객체들 중 적정 조건을 갖는 TMO 객체를 선정하는 과정들을 기술한다.

3.1 비중복 TMO 객체에서의 객체 레퍼런스 선정

TMO 객체가 객체그룹에 비중복되어 존재할 경우, 동적바인더객체의 동적 바인딩 서비스는 수행되지 않는다. 클라이언트의 TMO 객체 레퍼런스 요청이 그룹관리자객체에게 들어오고, 해당 TMO 객체의 접근 권한 검

사가 이루어진 후 정보저장소객체에서 클라이언트가 서비스 요청 시 제공한 서비스명으로 TMO 객체 레퍼런스를 검색한다. 검색된 TMO 객체 레퍼런스의 중복성을 확인하여, 비중복 TMO 객체인 경우 해당 TMO 객체 레퍼런스를 그룹관리자객체에게 즉시 반환한다.

3.2 중복 TMO 객체에서의 객체 레퍼런스 선정

TMO 객체그룹에 동일한 서비스를 지원하는 TMO 객체가 두 개 이상 중복되어 존재할 경우, 동적바인더객체는 적정 조건을 갖는 TMO 객체를 선정하기 위해 바인딩 우선순위 알고리즘을 적용한 동적 바인딩 서비스를 수행한다. 동적바인더객체는 서비스를 수행하는 TMO 객체가 위치하는 서버 시스템의 부하정보와 클라이언트가 제공한 요청마감시간 정보를 이용하여 TMO 객체를 선정한다. 여기서 적정 조건의 객체란, 여러 서버 시스템들 상에 동일한 특성을 갖는 객체들(중복객체라 부름)이 존재할 때, 클라이언트의 서비스 요청에 대해 가장 빠른 서비스를 제공할 수 있는 객체를 의미한다. TMO 객체그룹은 클라이언트가 서비스 요청 시 제공한 요청마감시간으로, 선정된 TMO 객체에 바인딩 후 서비스를 받기 위해 스케쥴러객체에 대기해야 하는 시간을 예측할 수 있다. 본 모델에서 부하를 고려한 동적 바인딩 알고리즘을 정의하고 수행결과를 보이기 위해 다음과 같은 환경을 가정한다.

- 동일한 특성을 가지는 TMO 객체들은 서버 시스템들 상에 두개 이상 중복되어 존재한다.
- 동적바인더객체는 중복된 TMO 객체들(TMO1, TMO2) 각각에 대해 객체 선정을 위한 바인딩 우선순위 리스트를 가진다.
- 부하정보로 각 서버 시스템의 CPU 이용률을 이용하며, 이 정보는 서버 시스템의 상태에 따라 변경된다.
- 바인딩 우선순위 리스트의 정보는 클라이언트의 요청을 수행하는 TMO 객체의 서비스 수행이 완료되면 갱신된다.

클라이언트의 TMO 객체 레퍼런스 요청이 그룹관리자객체에 들어오고, 보안객체에서 접근 권한 검사 후 정보저장소객체에 클라이언트의 요청을 수행할 TMO 객체 레퍼런스가 요청된다. 정보저장소객체는 중복된 TMO1과 TMO2에 대해, 동적바인더객체에게 서버 시스템의 부하정보와 요청마감시간 정보를 이용한 TMO 객체 선정 작업을 요청하게 되고, 동적바인더객체는 바인딩 우선순위 알고리즘을 적용하여 TMO 객체의 레퍼런스를 선정한다. 마지막으로, 선정된 TMO 객체 레퍼런스가 클라이언트에게 반환한다.

3.3 바인딩 우선순위 알고리즘

동적바인더객체는 중복 TMO 객체에 대해, TMO 객체들이 존재하는 서버 시스템의 부하정보와 클라이언트에서 제공한 요청마감시간 정보를 바인딩 우선순위 알고리즘에 적용하여 서비스 요청을 수행할 TMO 객체를 선정한다. 바인딩 우선순위 알고리즘은 다음의 정보들과 수식을 이용한다.

- *CPU_utilization* : TMO 객체가 존재하는 서버 시스템의 CPU 이용률로 시스템의 상태에 따라 변한다.
- *request_deadline* : 클라이언트의 서비스 요청 시 제공하는 요청마감시간을 나타낸다.
- *anticipative_service_time* : 클라이언트의 요청에 대해 TMO 객체에서 서비스를 수행하는 시간을 산출하기 위한 예측서비스시간(anticipative_service_time, AST)으로 클라이언트가 서비스 요청 시 제공하는 클라이언트 요청시간(client_invocation_time)과 그룹관리자객체가 존재하는 시스템의 시스템시간(GM_ system_ time)을 이용하여 다음의 식(2)로 예측서비스시간을 산출한다.

$$anticipative\_service\_time = request\_deadline - (2*transfer\_time) - client\_invocation\_time \quad (2)$$

식(1)에서 *"GM_invoked_time - client_invocation_time"*으로 클라이언트와 그룹관리자객체 사이의 통신을 위한 전송시간(transfer_time)을 산출한다.

- $\sum_{i=0}^{k} anticipative\_service\_time_i$: 예측서비스시간(AST)의 합으로 객체 선정 후 해당 TMO 객체에 바인딩 시 서비스를 요청한 클라이언트의 요청이 스케줄러에 대기해야 할 시간을 예측할 수 있다.
- *binding_priority* : 클라이언트의 TMO 객체에 대한 바인딩 우선순위이며, 서비스 요청에 대한 binding_priority는 [4,5]의 정의를 확장하여 다음의 수식을 통해 얻는다.

$$binding\_priority_k = \frac{1}{\sum_{i=0}^{k} anticipative\_service\_time_i + \frac{c}{CPU\_utilization}} \quad (3)$$

binding_priority : 바인딩우선순위, k : 클라이언트 수, *anticipative_service_time* : TMO 객체의 예측서비스시간, *CPU_utilization* : CPU 이용률, c : 비율상수(0.01)

위의 정보와 수식을 적용하여, 서버 시스템의 CPU 이용률과 클라이언트의 요청마감시간을 고려한 바인딩 우선순위 알고리즘은 그림 5와 같다.

Binding Priority Algorithm

new request$_k$ arrives on GroupManager with(client_name, service_name, request_deadline, client_invocation_time)

/* 바인딩 우선순위 리스트에 클라이언트의 요청 정보 삽입 */

insert request$_k$ information in each TMO object's binding priority list

/* 바인딩 우선순위 계산 */

compute the binding_priority$_k$ with

$$binding\_priority_k = \frac{1}{\sum_{i=0}^{n} anticipative\_service\_time_i} + \frac{c}{CPU\_utilization}$$

insert the binding_priority$_k$ in each TMO object's binding priority list

/* 바인딩 우선순위 비교하여 적정 TMO 객체 선정, 우선순위가 높은 TMO 레퍼런스 반환 */

if(TMO1's binding_priority$_k$ ≥ TMO2's binding_priority$_k$)

return TMO1's reference

else

return TMO2's reference

/* 바인딩 우선순위 리스트에서 선정되지 않은 TMO의 클라이언트 정보 삭제 */

delete request$_k$ information in TMO's binding priority list that didn't return reference

그림 5 중복 TMO 객체에 대한 바인딩 우선순위 알고리즘

3.4 동적 바인딩 서비스 수행 결과

그림 5의 바인딩 우선순위 알고리즘을 기반으로 중복 TMO 객체에 대해 부하를 고려한 적정 조건을 갖는 TMO 객체 선정과정을 보이기 위해, TMO 객체그룹이 클라이언트의 서비스 요청을 받은 후, 중복 객체인 TMO1과 TMO2에 대해 적정 조건의 객체를 선정하는 과정을 위에서 제시한 알고리즘을 통해서 살펴본다. 초기에 TMO1과 TMO2의 바인딩 우선순위 리스트는 비어있으며, TMO1과 TMO2의 CPU 이용률은 각각 10%와 11%이다. 클라이언트로부터 그룹관리자객체에 객체관리 서비스 요청 시 소요되는 전송시간(Transfer Time, TT)은 2sec이며 1sec을 주기로 클라이언트의 요청이 발생한다. 클라이언트1(c1)의 서비스 요청(CIT=10:00:00sec)이 10:00:09sec의 요청마감시간을 가지고 그룹관리자객체에 서비스 요청되면, 동적바인더객체의 TMO 객체 바인딩 우선순위 리스트에 클라이언트 정보가 각각 삽입되고 바인딩 우선순위 알고리즘에 따라 우선순위를 계산한다. 바인딩 우선순위 계산식인 식(3)에 적용한 TMO1의 바인딩 우선순위는 0.300이고, TMO2의 바인딩 우선순위는 0.291이다. 바인딩 우선순위 리스트의 바인딩 우선순위를 비교하면, TMO1의 바인딩 우선순위가 TMO2보다 높기 때문에 동적바인더객체는 TMO1을 선정하여 그룹관리자객체에게 TMO1의 레퍼런스를 반환하고, TMO2의 바인딩 우선순위 리스트에서 c1의 정보를 삭제한다. c1의 서비스 수행 중 클라이언트2(c2)의 서비스 요청(CIT=10:00:01)이 10:00:14의 요청마감시간을 가지고 들어오면, 위와 같은 방법으로 바인딩 우선순위를 계산한다. TMO1의 바인딩 우선순위는 0.171이고, TMO2의 바인딩 우선순위는 0.202이다. TMO2의 바인딩 우선순위가 TMO1보다 높기 때문에 동적바인더객체는 TMO2를 선정하여 그룹관리자객체에게 TMO2의 레퍼런스를 반환하고, TMO1의 바인딩 우선순위 리스트에서 c2의 정보를 삭제한다. 표 1은 c1과 c2의 요청에 대해 적정 조건을 갖는 TMO 객체의 레퍼런스를 선정하기 위한 바인딩 우선순위 리스트 정보이다.

동적 바인딩 서비스의 정확한 수행과정을 검증하기 위해, TMO1과 TMO2에서 c1, c2의 서비스 수행이 완료되지 않은 상태에서 클라이언트들(c3, c4, c5, c6, c7, c8)의 서비스 요청이 주기적으로 그룹관리자객체에 계속해서 도착하고, 동적바인더객체에서 바인딩 우선순위를 계산하여 바인딩 우선순위 리스트를 작성한다. 표 2는 TMO 객체그룹에 들어오는 클라이언트들의 서비스 요청에 따라 작성된 바인딩 우선순위 리스트 정보를 보여준다.

본 연구에서는 아래 표 2로부터 클라이언트의 요청을 수행 할 적정 조건을 가지는 TMO 객체의 선정 결과를 확인할 수 있다. 표 1에서 클라이언트1(c1)과 클라이언트2(c2)의 요청에 대해 TMO1과 TMO2를 선정하여 레퍼런스를 각각 반환했다. 이후 클라이언트3(c3) 요청에 대해서도 c1과 c2에서와 같이 식(3)을 적용, TMO1을 선정하여 레퍼런스를 클라이언트에게 반환하는 결과를

표 1 바인딩 우선순위 리스트에서 클라이언트2의 상태

TMO1

CN	*RD*	*AST*	*CU*	*BP*
c1	***10:00:09***	***5***	***0.10***	***0.300***
~~c2~~	~~10:00:14~~	~~9~~	~~0.10~~	~~0.171~~

TMO2

CN	*RD*	*AST*	*CU*	*BP*
~~c1~~	~~10:00:09~~	~~5~~	~~0.11~~	~~0.291~~
c2	***10:00:14***	***9***	***0.11***	***0.202***

CN:client_name, RD:request_deadline, AST:anticipative_service_time, CU:CPU_utilization, BP:binding_priority

표 2 TMO1과 TMO2의 바인딩 우선순위 리스트 정보

TMO1

CN	RD	AST	CU	BP
c1	10:00:09	5	0.10	0.300
c2	10:00:14	9	0.10	0.171
c3	10:00:12	6	0.10	0.191
c4	10:00:14	7	0.10	0.156
c5	10:00:16	8	0.10	0.138
c6	10:00:20	11	0.10	0.134
c7	10:00:18	8	0.10	0.127
c8	10:00:19	8	0.10	0.127

TMO2

CN	RD	AST	CU	BP
c1	10:00:09	5	0.11	0.291
c2	10:00:14	9	0.11	0.202
c3	10:00:12	6	0.11	0.158
c4	10:00:14	7	0.11	0.153
c5	10:00:16	8	0.11	0.150
c6	10:00:20	11	0.11	0.127
c7	10:00:18	8	0.11	0.131
c8	10:00:19	8	0.11	0.121

확인할 수 있다. 최종적으로 클라이언트 c1, c3, c4, c6, c8의 요청에 대해서는 TMO1을 선정하고, 클라이언트 c2, c5, c7의 요청에 대해서는 TMO2를 선정하여 클라이언트에게 TMO 객체 레퍼런스를 반환한다. 클라이언트가 선정된 TMO 객체에 바인딩 후 TMO 객체의 서비스 수행이 완료되면, 해당 클라이언트 정보를 바인딩 우선순위 리스트에서 삭제하고 리스트를 재작성한다. CPU 이용률의 변경은 해당 TMO 객체에 바인딩 된 클라이언트들의 요청에 대한 서비스 수행 성능에 전반적인 영향을 미치기 때문에 바인딩 우선순위 리스트의 CPU 이용률을 모두 변경하고 바인딩 우선순위를 재작성해야 한다. 우리가 제안한 모델은 클라이언트의 서비스 요청에 대해 비중복 또는 중복 객체들 중, 바인딩 할 적정 객체의 레퍼런스를 선정하는 동적 바인딩 서비스를 지원하여 요청마감시간 내에 선정된 TMO 객체로부터 서비스를 수행 받는 것을 보장할 수 있다.

4. 실시간 스케줄링 서비스 지원

TMO 객체그룹은 클라이언트의 요청 작업들에 대한 실시간 스케줄링 서비스를 제공한다. 실시간 스케줄링 서비스를 위해 EDF(Earliest Deadline First) 알고리즘을 적용하고, 수행결과로서 작업 우선순위 리스트를 얻게된다.

TMO 객체그룹에서 클라이언트의 서비스 요청에 대한 마감시간을 보장하는 실시간 서비스 수행을 위해서, 클라이언트가 서비스 요청 시 제공하는 요청마감시간과 클라이언트 정보를 이용하여 작업 우선순위를 실시간 스케줄링한다. TMO 객체는 실시간관리자객체에게 클라이언트명과 요청마감시간 정보를 전달한다. 실시간관리자객체는 계산된 서비스마감시간과 클라이언트명을 스케줄러객체에게 전달하여 EDF 알고리즘을 적용한 작업 우선순위를 스케줄링한다. 서비스 수행을 위한 TMO 객체들간 상호작용 후 결과를 클라이언트에게 반환하고 대기중인 우선순위가 가장 높은 작업을 수행한다. 다음 그림 6은 TMO 객체그룹에서 실시간 스케줄링을 위한 객체간 상호동작을 나타내는 ETD이다. 본 절에서 중점적으로 고려한 사항은 제시한 모델을 통해 임의의 실시간 스케줄링 전략을 채택하여 실시간 서비스의 지원이 가능한지 검증하는데 있으며, 주어진 스케줄링 전략의 우수함을 보이지는 않는다.

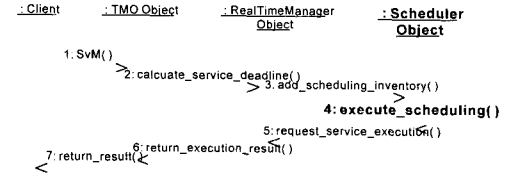

그림 6 TMO 객체그룹의 실시간 스케줄링 서비스 절차

4.1 실시간 스케줄링 서비스 방안

스케줄러객체는 TMO 객체가 이전에 요청된 서비스

수행 없이 대기 상태로 존재할 경우, 제일 먼저 들어오는 요청에 대해 즉시 서비스를 수행하도록 TMO 객체에 요청한다. 이후 들어오는 요청은 현재 TMO 객체가 수행 중인 서비스 요청이 완료되지 않았을 경우, 스케쥴러객체의 작업 우선순위 리스트에 적재되고 서비스 수행을 끝날 때까지 대기하게 된다. 다른 클라이언트로부터의 서비스 요청이 계속해서 들어오면, EDF 알고리즘을 적용한 실시간 스케쥴링으로 우선순위를 부여하여 작업 우선순위 리스트에 등록한다. 실시간 스케쥴링에서 우선순위가 가장 높은 작업은 서비스마감시간(SD)의 값이 가장 작은 작업이다. 본 연구에서는 현재 서비스 중인 작업에 대해서는 수행 완료를 보장하는 비선점형 스케쥴링을 하며, 작업 우선순위 리스트에 적재된 작업에 대해서만 EDF 알고리즘을 적용한다. 작업 우선순위는 작업의 도착이나 작업의 완료와 같은 이벤트가 발생할 경우 재 작성된다.

4.2 실시간 스케쥴링 서비스 수행 결과

본 장에서는 3.4절의 바인딩 우선순위 리스트에서 TMO1을 적정 조건의 바인딩 객체로 선정한 클라이언트들의 실시간 스케쥴링 서비스 결과를 보인다. 클라이언트1(c1)의 요청이 TMO1에 접수되고, 실시간관리자객체, 스케쥴러객체와 통신 후 현재 TMO1이 다른 클라이언트의 서비스를 수행하고 있을 경우 스케쥴러객체의 작업 우선순위 리스트에서 대기하게된다. 만일, TMO1이 수행중인 작업이 없을 경우 c1의 서비스 요청은 즉시 수행된다. c1의 수행 중 다른 클라이언트의 서비스 요청이 들어오면, c1에서의 객체간 통신과 동일한 스케쥴링 작업 수행 후 작업 우선순위 리스트에 등록된다. 이후 들어오는 요청에 대해서도 동일한 작업을 수행, 서비스마감시간을 비교하여 마감시간이 짧은 클라이언트 요청에 높은 우선순위를 부여하고 작업 우선순위 리스트의 상단에 등록한다. TMO1이 수행 중인 작업을 완료하고 대기상태가 되면, 작업 우선순위 리스트의 최상단에 등록된 대기중인 작업을 수행하도록 하고, 리스트에서 해당 정보를 삭제한 후 대기중인 작업들을 한 단계씩 위로 이동시킨다. 다음 표 3은 TMO1에 들어오는 클라이언트들의 요청들로, 실시간관리자에서 실시간 제약사항(식1, SD=RD-TT)을 적용하여 서비스마감시간이 계산된 결과를 보여준다. 전송시간(TT)은 2sec으로 가정한다.

표 3 클라이언트들의 요청에 대한 서비스마감시간

CN	*RD*	*TT*	*SD*
c1	10:00:09	2	10:00:07
c3	10:00:12	2	10:00:10
c4	10:00:14	2	10:00:12
c6	10:00:20	2	10:00:18
c8	10:00:19	2	10:00:17

CN:client_name, RD:request_deadline, TT:transfer_time, SD:service_deadline

표 3에서, c1의 요청이 10:00:07의 서비스마감시간으로 스케쥴러객체에 등록되면 작업 우선순위를 '1'로 하여 작업 우선순위 리스트의 최상단에 등록시킨다. TMO1이 현재 다른 클라이언트로부터의 요청에 대해 서비스를 수행중인 경우 스케쥴러객체의 작업 우선순위 리스트에서 대기 상태로 존재한다. c3의 요청이 TMO1에 들어오면, c1과 c3의 서비스마감시간을 비교하여 마감시간이 짧은 요청을 작업 우선순위 리스트의 상단에 등록한다. c1의 서비스마감시간이 10:00:07이고 c3의 서비스마감시간이 10:00:10이기 때문에 c1의 우선순위는 '1'을 가지고 c3의 우선순위는 '2'를 갖는다. c4, c6, c8에 대해서 c1, c3에서와 같은 절차를 수행하여 작업 우선순위 리스트에 등록한다. 아래 표 4는 TMO1에 클라이언트의 서비스 요청 c1, c3, c4, c6, c8이 스케쥴러객체에서 실시간 스케쥴링 후 작업 우선순위 리스트를 보여준다.

c6의 서비스마감시간이 10:00:18이고 c8의 서비스마감시간이 10:00:17으로 c6이 c8보다 작업우선순위가 낮기 때문에 c8을 c6보다 작업 우선순위 리스트의 상단에 위치시킨다. TMO1의 수행 중인 작업이 완료되면 작업 우선순위 리스트에서 우선순위가 가장 높은 c1에 대한 서비스를 수행하고, c1의 정보를 우선순위 리스트에서 삭제하여, 작업 우선순위 리스트를 재작성한다.

본 모델은 하나의 TMO 객체에 들어오는 여러 클라이언트들의 요청에 대해 마감시간 내에 실시간 서비스를 보장하는 실시간 스케쥴링 서비스를 제공하며, 또한 클라이언트가 서비스 요청 시 제공하는 요청마감시간을 시간 제약조건에 적용하여 서버 시스템에서 서비스 수행 시 발행할 수 있는 서비스마감시간 위반을 예측할 수 있다. 즉, 실시간관리자객체는 요청마감시간과 전송시간으로 서비스마감시간 계산 후 서비스마감시간과 서

표 4 스케쥴러객체의 작업 우선순위 리스트

CN	*RD*	*TT*	*SD*	*TP*
c1	10:00:09	2	10:00:07	1
c3	10:00:12	2	10:00:10	2
c4	10:00:14	2	10:00:12	3
c8	***10:00:19***	***2***	***10:00:17***	***4***
c6	***10:00:20***	***2***	***10:00:18***	***5***

TP:task_priority

버 시스템의 전역시간을 비교하여 해당 요청에 대한 마감시간을 위반을 미리 확인하고 처리할 수 있다.

5. 결 론

분산 실시간 객체 컴퓨팅은 분산된 객체들이 실시간 서비스에 대한 요구를 만족시킬 수 있는 환경을 기반으로 하며, 특히 분산 객체를 관리하는 측면에서 중복되어 존재하는 분산 객체에 바인딩 시 적정 조건을 가지는 객체를 선정할 수 있는 동적 바인딩 방안과 실시간 서비스 측면에서 여러 클라이언트들의 서비스 요청에 대한 실시간 스케쥴링 방안이 지원되어야 한다.

따라서, 본 논문에서 제안한 TMO 객체그룹은 분산 실시간 어플리케이션을 위해 실시간 CORBA나 실시간 운영체제를 고려하지 않고, 실시간 특성을 자체적으로 가지는 TMO 객체를 그룹으로 하여 COTS 미들웨어 상에서 보장된 분산 실시간 서비스를 제공하는 모델이다. 본 모델의 구축을 위해, TMO 객체의 개념과 모델의 구조, 객체그룹 컴포넌트의 기능과 상호작용과 같은 TMO 객체그룹의 전반사항을 살펴봤고, TMO 객체그룹이 지원하는 객체 관리 서비스와 실시간 스케쥴링 서비스를 위한 컴포넌트를 구현했다. 객체 관리 서비스 관점에서, 외부 클라이언트로부터 TMO 객체그룹 내의 비중복 또는 중복 TMO 객체에 서비스 요청 시, TMO 객체가 존재하는 서버 시스템의 부하정보와 클라이언트의 요청마감시간 정보를 이용하여 적정 조건을 가지는 TMO 객체를 선정하는 동적 바인딩 방안을 정의했고, 동적 바인딩 알고리즘을 적용한 동적바인더객체를 구현하여 수행과정과 선정결과를 보였다. 위의 전략은 TMO 객체가 존재하는 서버 시스템들 사이의 부하를 고려한 전략으로, 선정된 TMO 객체에 바인딩 시 스케쥴러객체에 대기해야 하는 시간을 예측하여, TMO 객체들이 수행하는 전체 분산 실시간 어플리케이션의 서비스 수행 속도를 향상시킬 수 있다. 그리고, 실시간 스케쥴링 서비스 관점에서, 선정된 객체에 바인딩 시 여러 클라이언트로부터의 요청 작업들에 대해 EDF 알고리즘을 적용한 실시간 스케쥴링 전략을 정의하고, 스케쥴러객체를 구현하여 실시간 스케쥴링 결과를 보였다. 이는 클라이언트들의 요청에 대해 마감시간 내에 실시간 서비스를 보장하는 실시간 작업 스케쥴링을 제공하고, 클라이언트가 서비스 요청 시 제공하는 요청마감시간으로 서버 시스템에서 실제 서비스 수행 시 발행할 수 있는 마감시간 위반을 예측할 수 있다. 마지막으로 수치 분석을 통하여 우리 모델이 클라이언트의 요청으로부터 비중복/중복 TMO에 대한 동적 바인딩 서비스와 임의의 TMO 객체를 위한 실시간 스케쥴링 서비스를 지원하는지 검증했다. 우리의 연구는 분산환경에서 TMO 객체들을 그룹으로 관리하여 분산 실시간 서비스를 제공하는 TMO 객체그룹 플랫폼 구현을 목적으로 하기 때문에, 위에서 제시한 전략들은 성능이 우수한 다른 연구 전략으로 대체 가능하다.

본 연구에서의 실시간 스케쥴링 알고리즘은 중복 객체의 선정시 마감시간을 기반으로 작업 우선순위를 예측하여 서비스하는 모델로, 분산 방위 시스템과 같은 분산 실시간 응용 서비스들에 적용될 수 있는 모델이다. TMO 객체그룹은 객체 존재 범위가 결정되지 않은 분산 환경에 적용될 수 있는 모델로, 이러한 분산 중복 객체들을 그룹으로하여 객체 관리를 위한 일관된 방안을 제공한다. 실시간 시스템에서 스케쥴링을 위해 본 연구에 적용한 예측 가능성만큼이나 중요한 요소는 분산 객체에 대한 작업이나 객체의 중요도를 고려한 스케쥴링으로 본 연구에서 추후 고려해야할 사항이다.

이후 연구로 우리의 모델을 이용하여, 새로운 동적 바인딩 서비스와 실시간 스케쥴링 서비스 정책을 다양하게 적용할 수 있는 기본 플랫폼을 구현하고, 분산 실시간 시뮬레이션을 통해 우리 모델이 제공하는 분산 실시간 서비스 수행능력의 우수성을 보이고자 한다.

참 고 문 헌

[1] M. Takemoto., *"Fault-Tolerant Object on Network-wide Distributed Object-Oriented Systems for Future Telecommunications Applications", In IEEE PRFTS,* pp.139-146, 1997.

[2] W.J. Lee, C.W. Jeong, M.H. Kim, and S.C. Joo., *"Design and Implementation of An Object Group in Distributed Computing Enviroments", Journal of Electronics & Computer Science,* Vol.2, No.1, 2000.

[3] E.D. Jensen, C.D. Locky, and H. Tokuda., *"A Time-Driven Scheduling Model for Real-Time Operating Systems", In Proc. 6th IEEE Real-Time System Symposium,* pp.112-122, 1985.

[4] V. Kalogeraki, P.M. Melliar-Smith, and L.E. Moser., *"Dynamic Scheduling for Soft Real-Time Distributed Object Systems", In Proc. IEEE 3rd Int'l Symp. on Object-Oriented Real-Time Distributed Computing,* pp. 114-121, 2000.

[5] P.M. Melliar-Smith, L.E. Moser, and P. Narasimhan., *"Consistent object replication in the Eternal System", Theory and Practice of Object System,* Vol.4, No.2, pp.81-92, 1998.

[6] 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
[7] Object Management Group, *"The Common Object Request Broker: Architecture and Specification 2.2"*, http://www.omg.org/corba/corbaCB.htm, 1998.
[8] OMG Real-time Platform SIG., *"Real-time CORBA A White Paper-Issue 1.0"*, http://www.omg.org /realtime/real-time_whitepapers. html, 1996.
[9] C.S. Shin, M.H. Kim, Y.S. Jeong, S.K. Han, S.C. Joo, *"Construction of CORBA Based Object Group Platform for Distributed Real-Time Services", In Proc. 7th IEEE Int'l Workshop on Object-oriented Real-time Dependable Systems(WORDS'02)*, pp.229 -302, 2002.
[10] C.S. Shin, M.S. Kang, Y.S. Jeong, S.K. Han, S.C Joo, *"TMO-Based Object Group Model for Distributed Real-Time Services", In Proc. IASTED Int'l Conference Networks, Parallel and Distributed Processing, and Applications(NPDPA'02)*, pp.178- 183, 2002.
[11] K.H. Kim., *"Object-Oriented Real-Time Distributed Programming and Support Middleware", In Proc. 7th Int'l Conf. on Parallel & Distributed System*, pp. 10-20, 2000.
[12] K.H. Kim, Seok-Joong Kang, and Yuqing Li., *"GUI Approach to Generation of Code-Framworks of TMO", In Proc. 7th IEEE Int'l Workshop on Object-oriented Real-time Dependable Systems(WORDS)*, pp.17-25, 2002.
[13] Eltefaat Shokri, Patrick Crane, and K.H. Kim., *"An Implementation Model for Time-Triggered Message-Triggered Object Support Mechanism in CORBA-Compliant COTS Platforms", In Proc. IEEE 1st Int'l Symp. on Object-oriented Real-time dependable Computing(ISORC)*, pp. 12-21, 1998.
[14] John A. Stankovic, Marco Spuri, Krithi Ramamrithm, Giorgio C. Buttazzo., *Deadline Scheduling for Real-Time Systems.*, p.31, Kluwer Academic Publishers, 2002.
[15] K.H. Kim, Juqiang Liu,Masaki Ishida, and Inho Kim., *"Distributed Object_Oriented Real-Time Simulation of Ground Transportation Network with TMO Structure Scheme", In Proc. IEEE CS 23rd Int'l Computer Software & Application Conference*, pp.130-138, 1999.
[16] Victor Fay Wolfe, et al., *"Expressing and Enforcing Timing Constraints in a Dynamic Real-Time CORBA System"*, http://www.cs.uri.edu/rtsorac/publication.htm, 1997.
[17] C.L. Liu., *"Fundamentals of real-Time Scheduling (Extended Abstract)", In Proc. NATO Advances Study Institute on Real Time Computing*, pp.1-7, 1992.
[18] K.H. Kim, D. Beck, J.Q. Liu, H. Miyazaki, and E.H Shokri., *"A CORBA Service Enabling Programming-Friendly Object-Oriented Real-Time Distributed Computing", In Proc. 5th IEEE Int'l Workshop on Object-oriented Real-time Dependable Systems(WORDS)*, pp.101-107, 1999.
[19] E. Damiani., *"A Fuzzy Stateless Approach to Load Distribution for Object-Oriented Distributed Environments", International Journal of Knowledge-Based Intelligent Engineering Systems*, Vol.3, No.4, pp.240-253, 1999.
[20] 주수종, "분산처리환경에서 객체그룹 모델링 및 성능분석에 관한 연구", 한국전자통신연구원 최종보고서, 1997.
[21] 신창선, 강명석, 김명희, 주수종, "TMO 기반 분산 실시간 객체그룹 관리방안에 대한 연구", 한국인터넷정보학회 추계 학술발표논문집, Vol.2, No.2, pp.123-126, 2001.
[22] 김명희, 주수종, "분산 실시간 서비스를 위한 CORBA 객체그룹 플랫폼의 구축", 정보과학회 논문지, Vol.7, No.6, 2001.

신 창 선
1996년 우석대학교 전산학과 학사.
1999년 한양대학교 컴퓨터교육과 석사.
2000년~현재 원광대학교 컴퓨터공학과 박사과정. 관심분야는 분산 실시간 컴퓨팅, 분산 알고리즘, 실시간 객체 모델, 실시간 시뮬레이션

김 명 희
1993년 원광대학교 컴퓨터공학과 학사.
1996년 원광대학교 컴퓨터공학과 공학석사.
2001년 원광대학교 컴퓨터공학과 공학박사.
2002년~현재 원광 디지털대학교 전임강사.
관심분야는 분산 실시간 컴퓨팅, 시스템 최적화, 운영체제

주 수 종
1986년 원광대학교 전자계산공학과 학사.
1988년 중앙대학교 컴퓨터공학과 공학석사.
1992년 중앙대학교 컴퓨터공학과 공학박사.
1993년 미국 University of Massachusetts at Amherst, Post-Doc. 1990년~현재 원광대학교 컴퓨터공학과 교수 2003년~현재 미국 University of California at Irvine, Visiting Professor.
관심분야는 분산 실시간 컴퓨팅, 분산객체모델, 시스템 최적화, 멀티미디어 데이타베이스