□신기술해설□

실시간 CORBA

정 찬 균† 이 승 룡†† 이 병 호†††

◈ 목 차 ◈

1. 서 론

CORBA(Common Object Request Broker Architecture)는 OMG(Object Management Group)에서 발표한 분산 객체 환경으로서 Microsoft의 DCOM (Distributed Component Object Model)과 같은 다른 분산 환경 구조를 제치고 산업계의 표준으로 떠오르고 있다. CORBA는 클라이언트나 서버 객체의 동작환경에 관계없이 클라이언트가 구현 객체(object implementation)에서 제공하는 오퍼레이션을 자유롭게 요청해서 사용할 수 있는 분산 환경을 제공해준다[2].

그러나 CORBA는 시간 제약을 가지는 실시간 응용프로그램을 지원하기에는 제한사항들이 많기 때문에 OMG의 RT SIG(Real-Time Special Interest Group)에서는 표준 CORBA에 실시간 지원을 위한 기능을 추가하거나 확장하는 작업을 진행하고 있으며 이미 많은 기업체와 대학에서는 이에 적극 참여하고 있다. 이러한 관점에서, 본 논문에서는 표준 CORBA의 실시간 확장을 위한 요구사항들을 기술하고 개발 현황을 소개한다.

본 논문의 구성은 다음과 같다. 2장에서는 OMG의 표준 CORBA에 대하여 살펴보고, 3장에서는 실시간 CORBA의 요구사항과 제약 사항에 대하여 논의하며, 4장에서는 실시간 CORBA의 개발 현황과 특징을 소개하고, 5장에서 결론을 맺는다.

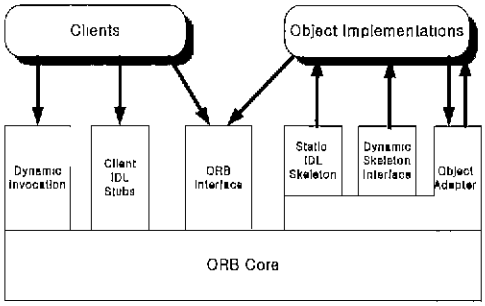
2. CORBA 배경

OMG의 목표는 분산, 이기종 환경에서 객체기반 응용 프로그램들의 재사용성, 이식성, 상호운용성(interoperability)을 위한 공통 프레임워크 환경을 제공하고 그에 대한 표준을 제정하는 것이다. OMG에서는 객체지향 기반의 분산 환경에서 응용 프로그램들을 서로 통합 할 수 있는 기술을 제안하였는데 이것이 객체 관리 구조(OMA: Object Management Architecture)이다. OMA는 응용프로그램 사이의 결합뿐만 아니라 객체의 생성, 소멸에서부터 저장, 트랜잭션 기능에 이르기까지 분산 객체 환경에 필요한 모든 서비스를 다루고 있다. OMA는 객체 요청 중개자(ORB: Object Request Broker), 객체 서비스(object services), 공통 기능요소(common facilities), 도메인 인터페이스(domain interfaces), 응용 객체(application objects)로 구성되

† 준회원 : 경희대학교 전자계산공학과 석사과정
†† 정회원 : 경희대학교 전자계산공학과 부교수
††† 정회원 : 쌍용투자증권 이사

어 있다. 그중 객체 서비스는 분산 객체로 구성된 CORBA 기반의 응용프로그램을 개발할 때 기본이 되는 서비스이다[1].

OMA에서 ORB 구현의 역할을 수행하는 CORBA는 응용 프로그램 사이에서 데이터가 전달되는 통로 즉, 소프트웨어 버스 역할을 한다[2]. CORBA는 (그림 1)과 같이 구성되어 있으며, 각각의 기능은 <표 1>에 나타나 있다.



(그림 1) CORBA의 구성요소

<표 1> CORBA 구성요소 및 기능

구성요소 이름	기 능
클라이언트	구현 객체의 오퍼레이션을 호출
구현 객체	객체 인스턴스를 위한 데이터와 객체의 오퍼레이션을 위한 코드를 정의함으로서 객체 내부 의미를 제공
클라이언트 IDL stubs/정적 IDL skeleton	클라이언트와 구현 객체가 통신을 하기 위한 프로그램 코드
동적 실행 인터페이스 (Dynamic Invocation Interface)	프로그램이 실행 중이더라도 실행하고자 하는 구현 객체의 오퍼레이션을 찾고 이를 실행할 수 있게 해주는 인터페이스
동적 skeleton 인터페이스	skeleton과 stubs을 가지고 있지 않은 다른 외부 객체의 오퍼레이션을 실행하려는 구현 객체들을 위해 제공되는 인터페이스
객체 아답터 (Object Adapter)	클라이언트의 요청을 받아 들여 적절한 구현객체로 디멀티플렉싱하고, 해당 오퍼레이션을 호출
ORB 인터페이스	직접 ORB에 접근하기 위하여 제공되는 인터페이스
ORB 코아 (ORB Core)	클라이언트의 요청을 구현 객체에 전달하고 그에 대한 응답 결과를 클라이언트로 돌려주는 역할을 담당

3. 실시간 CORBA

CORBA는 잘 설계된 분산 처리 환경임에도 불구하고, QoS(Quality of Service) 보장과 예측성 결여로 인하여 시간 제약을 가진 실시간 응용프로그램에 그대로 적용하기에는 적합하지 않다. 현재 실시간 응용 프로그램 지원을 위한 표준 CORBA의 제한 사항은 다음과 같다[6].

- QoS를 명시하는 API의 부재
  CORBA는 종단간의 QoS 요구를 명시하는 API들을 제공하지 않으므로, 클라이언트의 요청에 우선순위, 마감시간과 같은 조건을 표시할 수 없다.
- QoS를 강요하는 기능의 결여
  CORBA는 클라이언트의 요청을 스케줄링 하기 위해 FIFO 알고리즘을 사용하므로 무한정 우선순위 역전 현상이 일어날 수 있으며, ORB에서 구현 객체의 자원 과용 같은 잘못된 동작을 제어할 수 없다.
- 실시간 프로그래밍 기능의 결여
  현재의 CORBA는 실시간 프로그래밍을 지원하기 위해 필요한 핵심 기능을 정의하지 않고 있어, 실시간 특성을 지닌 응용프로그램을 개발하기가 어렵다.
- 성능 최적화 기능의 부족
  기존의 CORBA는 내부 구조의 불필요한 데이터 처리와 비효율적인 클라이언트 요청 디멀티플랙싱(demultiplexing)으로 인해 낮은 처리율과 높은 오버헤드를 가질 수 있다.

이러한 이유로, RT SIG는 1996년 표준 CORBA를 실시간 CORBA로 확장 개발할 때 필요한 기술과 개념을 정의하기 위하여 실시간 CORBA 백서 1.0을 발표하였다. 그리고, 실시간 CORBA 표준을 만들기 위해 현재 각 참여 회사 및 단체들에게 실시간 CORBA 1.0 RFP(Request For Proposal)를 발행

하여 표준화 작업을 진행하고 있다.

3.1 실시간 CORBA 백서(white paper)

실시간 CORBA 백서에서는 기본적으로 고려되어야 할 실시간 개념을 시간 제약, 스케줄링, QoS, 성능, 동기적/비동기적 분산 시스템 모델, 결함이 있을 때의 동작 등으로 규정하고 있다.

<표 2> 실시간 CORBA의 요구사항

네트워크, 운영체제 동작에 대한 요구사항	• 노드상에서 모든 클럭의 동기화 • 제한된 메시지 지연시간의 보장 • 선점형 우선순위 기반의 스케줄링의 지원 • 우선순위 계승기법의 구현
ORB와 객체 서비스에 대한 요구사항	• 절대적 시간과 상대적 시간의 표준 타입 명시 • 실시간 메소드 호출 정보의 전달 지원 • 전역 우선순위의 지원 • 모든 CORBA 개체 서비스에서의 우선순위 큐잉 지원 • 실시간 이벤트의 지원 • 실시간 예외 처리의 지원 • ORB에서의 QoS 보장

그리고, CORBA에서 고려해야 할 실시간 기술로는 CPU 스케줄링과 디스패칭 기술, 실시간 통신 프로토콜, 실시간 시스템에서 결함허용 소프트웨어의 구현, QoS 관리, 경량급의 CORBA, 클럭 동기화 등으로 규정하고 있다. 이와 같은 실시간 CORBA의 주요 요구사항을 정리하면 <표 2>와 같다.

한편, 실시간 CORBA가 응용될 수 있는 시장은 매우 다양하며, 현재 통신, 항공 및 국방, 제조업, 금융, 전자 상거래 등의 분야에서 이용되고 있다[4].

3.2 실시간 CORBA의 표준화 작업

실시간 CORBA RFP에서 진행하는 표준화 작업의 목적은 CORBA의 장점을 살리고 실시간 확장을 위한 기반을 제공하는 명세서를 만들어 내는데 있다. 이러한 목적들과 함께 RFP는 실시간 ORB를 위한 기술로 고정 우선순위 스케줄링과 종단간 예측성을 위한 ORB 자원 관리, 융통성을 가진 통신을 권고한다[5]. RFP의 요구는 필수적인 사항과 선택적인 사항으로 나누어지는데 이는 <표 3>에 나타나 있다.

<표 3> RFP의 요구사항

필수 요구 사항	• 기존의 표준 OMG 명세서에서 확장되어야 함 • 고정 우선순위로 스케줄 가능한 개체(schedulable entity)의 정의 • 서버로 클라이언트의 우선순위를 전달하기 위한 구조와 인터페이스의 명시 • 메소드 호출의 blocking, avoiding, bounding을 위한 구조와 인터페이스의 명시 • 자원 관리를 위한 자원들의 정의 • 종단간 자원의 할당과 재 할당을 관리하는 인터페이스의 명시 • 서버와 클라이언트를 위해 선택될 수 있는 전송 프로토콜과 상호작용 프로토콜을 위한 인터페이스의 명시
선택 요구 사항	• 클라이언트가 요청에 관련된 시간 제한 정보에 대해 공보할 수 있는 기능과 인터페이스 명시 • 사용자에 의해 제공된 전송 프로토콜의 설치를 위한 인터페이스 명시 • 실시간 ORB간의 상호운용을 제공하기 위한 실시간 상호작용 프르토콜 명시 • 실행 시에 스케줄 가능 개체의 특성을 판단하기 위한 인터페이스 정의

4. 실시간 CORBA 개발현황

이 장에서는 현재 대학, 연구소, 기업체에서 개발중인 실시간 CORBA의 개발현황과 특징에 대하여 소개한다.

4.1 TAO(The Ace Orb)

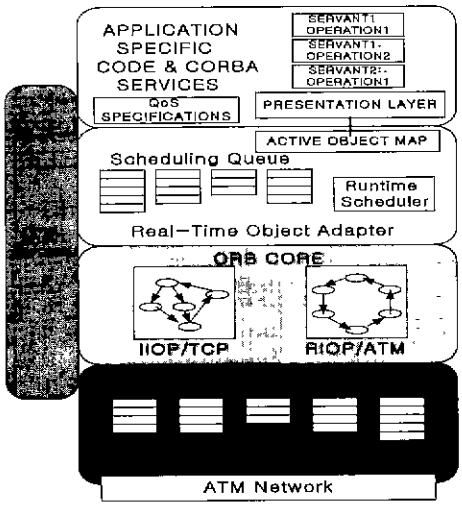
4.1.1 개요

TAO는 미국 워싱턴 대학에서 개발한 고성능의 표준 CORBA와 호환되는 실시간 ORB로서, VxWorks, Chorus, Solaris 2.x, Windows NT와 같은 다양한 운영체제 환경에서 동작한다[6], [7], [8]. TAO는 C++ 언어로 작성된 플랫폼을 지원하는 미들웨어 프레임워크인 ACE (Adaptive Communication Environment)를 사용하여 개발되었다. TAO의 목표는 결정적(deterministic)이고 통계적(statistical)

인 QoS를 보장하기 위해 요구되는 기능을 제공하고, 미들웨어와 분산 응용프로그램의 종단간 대역폭과 안정성을 보장할 수 있는 최적화된 실시간 I/O 서브 시스템을 가진 ORB를 제공하는데 있다. 또한, 확장이 용이한 실시간 ORB를 개발하고, OMG의 CORBA 표준에서 응용프로그램이 QoS 요구의 명시를 가능케 하는데 있다.

4.1.2 구성요소 및 기능

TAO는 (그림 2)와 같이 기가비트 실시간 I/O 서브 시스템, ORB 코아, 실시간 객체 아답터, 표현계층의 요소로 구성된다.



(그림 2) TAO의 시스템 구성요소

ORB간의 통신을 위해 IIOP(Internet Inter-ORB Protocols)뿐 아니라 RIOP(Realtime Inter-ORB Protocol)도 제공한다. ORB가 ATM 망에서 동작할 때에는 RIOP가 사용되고 TCP/IP망에서는 IIOP가 사용된다. 실시간 객체 아답터는 서번트 최하위 계층의 기가비트 실시간 I/O 서브 시스템은 ORB와 응용프로그램이 하위수준 네트워크와 운영체제 자원에 접근하도록 해주며, ATM 망이나 TCP/IP 망에 연결된다. 그 위에 있는 ORB 코아는 TAO 시스템의 핵심으로서 클라이언트의 요구를 객체 아답터에 전달하고 그에 대한 응답을 클라이언트로 중계해주는 역할을 담당하며, ORB간의 통신을 위해 IIOP(Internet Inter-ORB Protocols)뿐 아니라 RIOP(Realtime Inter-ORB Protocol)도 제공한다. ORB가 ATM 망에서 동작할 때에는 RIOP가 사용되고 TCP/IP망에서는 IIOP가 사용된다. 실시간 객체 아답터는 서번트(Servant: CORBA에서의 구현 객체)에 대한 클라이언트의 요청을 디멀티플랙싱하고 그 서번트에 있는 적절한 오퍼레이션을 디스패칭하는 일을 수행한다. 표현계층은 바이트 정렬, 데이터 형 크기와 같이 각 시스템마다 차이가 있는 부분을 공통적인 표현으로 변형(marshaling) 또는 역변형(demarshaling)을 수행한다. Zero Copy Buffer는 ORB에서의 동적 메모리 할당과 데이터 복사로 인한 오버헤드를 축소시킨다[6].

4.1.3 실시간 스케줄러

TAO에서 실시간 스케줄러는 객체 아답터에 위치하여 반드시 마감시간을 만족해야 하는 결정적 QoS 요구와 그렇지 않는 통계적 QoS 요구를 지원하도록 설계되었다. 그리고 런타임과 오프라인 스케줄링 방법을 지원하도록 설계되었다(현재는 오프라인 스케줄링만 구현되어 있다).

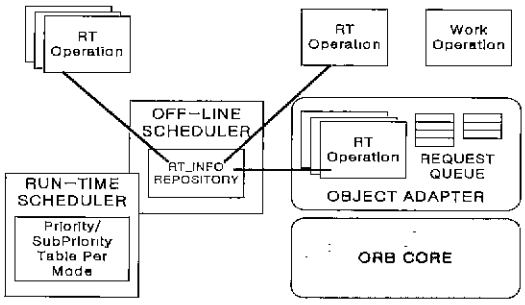
- 오프라인 스케줄링 서비스

오프라인 스케줄링 서비스는 현재 RM(Rate Monotonic) 스케줄링을 사용하기 때문에 오퍼레이션의 빈도에 따라 우선순위가 할당되며, 저장소(repository)에 있는 각 *RT_Operation*들은 다음의 규칙에 의해 우선순위가 할당된다. 오퍼레이션의 *RT_Info::period*가 0이 아니면, 그 값이 우선순위를 매핑 하는데 사용되고, *RT_Info::period*가 0 이면 *RT_Info* 구조체에 저장된 *operation_ dependencies* 필드로부터 빈도를 알아내어 우선순위에 매핑 시

킨다.

● 런타임 스케줄링 서비스

TAO의 런타임 스케줄링 모델은(그림 3)과 같은 구성요소를 포함한다. *Work_Operation*은 응용프로그램 수준에서 처리나 상호간의 통신을 수행하는 하나의 작업 단위이다. 즉, 입력을 받거나 결과를 출력하거나 물리적 단위를 변환하는 것들이 *Work_Operation*이다. *RT_Operation*은 시간 제약을 가진 *Work_Operation*중에 하나의 형식이며, 각 *RT_Operation*은 자신의 QoS 정보를 가진 일종의 오퍼레이션(CORBA에서 객체에 정의된 메소드)으로 생각되어 진다. 쓰레드는 동시에 실행되는 단위를 말하고, 객체 아답터 디스패처는 다음에 CPU에서 실행될 쓰레드를 선택하기 위해 클라이언트 요청의 우선순위를 이용한다. 그리고 런타임 스케줄러는 실행시 클라이언트의 요구를 스케줄링하는 역할을 한다. 이때 *RT_Info*는 QoS 정보를 저장하는 구조체이다.



(그림 3) TAO 런타임 스케줄링 구성요소

그리고 런타임 스케줄러는 실행시 클라이언트의 요구를 스케줄링하는 역할을 한다. 이때 *RT_Info*는 QoS 정보를 저장하는 구조체이다.

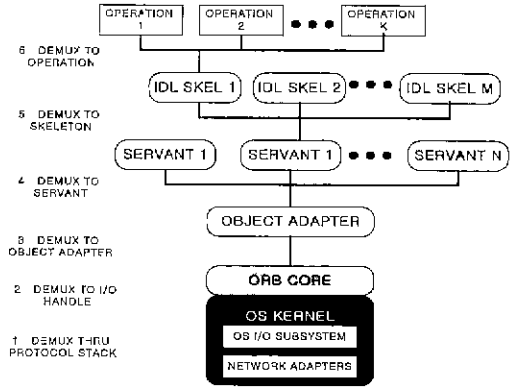
4.1.4 디멀티플랙싱

● 표준 CORBA의 디멀티플랙싱

표준 CORBA 객체 아답터에서는 클라이언트의 요청을 <표 4>와 같은 순서에 의해 디멀티플랙싱한다 (그림 4) 참조.

<표 4> CORBA의 계층적 디멀티플랙싱 순서

단계 1, 2	단계 1, 2에서는 운영체제의 프로토콜 스택에서 들어오는 클라이언트의 요청이 디멀티플랙싱 된다. 즉 데이터 링크, 네트워크, 트랜스포트 계층을 거쳐 사용자와 커널의 경계와 ORB 코아로 디멀티플랙싱 된다.
단계 3, 4, 5	ORB 코아는 적절한 객체 아답터, 서번트, IDL 오퍼레이션의 skeleton을 위치시키기 위해 클라이언트의 객체 키(object key)안의 주소정보를 이용한다
단계 6	IDL skeleton은 클라이언트의 요청을 적절한 오퍼레이션으로 위치시키며, 요청버퍼를 오퍼레이션 파라미터로 변환하고(demarshal), 오퍼레이션 업콜(upcall)을 수행한다.



(그림 4) 계층적 CORBA 요청 디멀티플랙싱

이러한 계층적(layered) 디멀티플랙싱은 오버헤드가 많기 때문에 실시간 응용프로그램에게는 적합하지 않다.

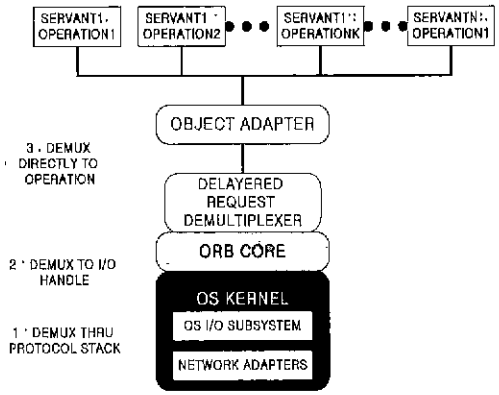
● TAO에서 디멀티플랙싱

TAO의 객체 아답터는 delayered 디멀티플랙싱을 통해 오버헤드를 감소시키도록 설계되었다 (그림 5 참조).

TAO에서는 CORBA의 다단계 멀티플랙싱의 오버헤드를 축소하기 위해 두 가지 방법을 사용하였는데, 먼저 퍼펙트 해싱은 두 단계의 디멀티플랙싱 방법으로서, 서번트를 찾기 위해 자동으로

생성된 퍼펙트 해싱 함수를 사용하며, 오퍼레이션을 찾기 위해 두 번째 퍼펙트 해싱 함수가 사용된다.

다른 하나의 방법은 액티브 디멀티플랙싱으로서, 이 방법에서는 클라이언트가 활성 오브젝트 맵과 CORBA 요청 헤더안의 오퍼레이션 테이블 안에 오브젝트에 대한 처리를 포함하며, 수신측의 객체 아답터는 객체와 그와 연관된 오퍼레이션을 찾기 위해 CORBA 요청 헤더안의 처리를 이용한다. 이 때 디멀티플랙싱은 한 단계로 진행된다 [6], [7].



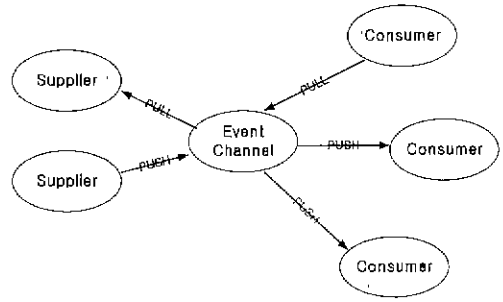
(그림 5) delayered TAO 요청 디멀티플랙싱

4.1.5 이벤트 서비스

- CORBA 이벤트 서비스

표준 CORBA 호출 모델의 제한사항들을 완화시키기 위해 설계된 것이 CORBA 객체 서비스 중의 하나인 이벤트 서비스이다. COS (CORBA Object Service) 이벤트 서비스는 비동기적 메시지 전송을 지원하며, 하나 이상의 공급자가 하나 이상의 소비자에게 메시지를 전송하는 것을 허용한다[3].

CORBA 이벤트 서비스는 공급자와 소비자, 이벤트 채널의 요소로 이루어져 있는데(그림 6), 소비자는 공급자에서 발생하는 이벤트를 받는 개체이며, 활성 푸쉬(push) 공급자는 수동적 푸쉬 소비자에게 이벤트를 밀어 보낸다. 마찬가지로, 수동적 풀(pull) 공급자는 활성 풀 소비자가 자신으로부터의 이벤트를 끌어 받기를 기다린다. 이벤트 채널은 공급자와 소비자 사이에서 매개체 역할은 수행하는 CORBA 이벤트 서비스의 핵심이다. 공급자는 소비자에게 이벤트를 밀어 보내기 위해 이벤트 채널을 사용한다.



(그림 6) CORBA 이벤트 서비스 구성도

그러나, 표준 CORBA 이벤트 서비스는 실시간을 지원하기 위해 필요한 실시간 이벤트 디스패칭과 스케줄링의 보장, 중앙 집중화된 이벤트 필터링과 상관성을 위한 명시, 주기적 처리 지원 등의 기능이 결여되어 있어 이를 지원하는 실시간 이벤트 서비스가 요구된다.

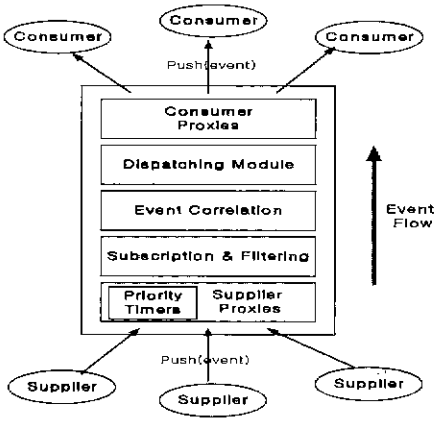
- TAO 실시간 이벤트 서비스 구조

실시간 이벤트 서비스는 (그림 7)과 같이 이벤트 채널, 소비자 프록시 모듈, 공급자 프록시 모듈, 전달신청(subscription) 및 필터링, 우선순위 타이머 프록시, 이벤트 상관성, 디스패칭의 구성요소 및 기능으로 이루어져 있다.

실시간 이벤트 서비스 모델에서 이벤트 채널은 기존의 CORBA 이벤트 서비스에서의 기능과 같은 역할을 수행한다. 소비자 프록시 모듈은 소비자가 채널에 접속하거나 접속을 해제하는데 사용된다. 공급자 프록시 모듈은 공급자가 채널에 접

속하거나 접속을 해제하는데 사용된다. CORBA 이벤트 서비스에서 소비자가 단지 공급자로부터의 이벤트중 일부분에만 관심이 있다면, 불필요한 이벤트를 받지 않게 하는 자신만의 이벤트 필터링을 구현해야 한다. 이러한 단점을 극복하기 위해 TAO의 실시간 이벤트 서비스는 소비자가 특정 이벤트들의 집합만을 신청할 수 있게 전달신청 및 필터링 기능이 추가되었다.

공급자 프록시 모델은 채널에 등록된 모든 타이머를 관리하는 특별한 목적의 우선순위 타이머 프록시를 포함하며, 우선순위 타이머 프록시는 소비자가 종료시간을 등록할 때 종료시간이 소비자의 우선순위에 따라 분배되도록 하는 런타임 스케줄러와 함께 동작한다. TAO의 실시간 이벤트 서비스에서 이벤트 상관성을 이용하기 위해, 소비자는 필터링 요구사항을 등록할 때 결합적(AND) 혹은 분리적(OR) 의미를 명시할 수 있다. 결합적 의미는 명시된 모든 이벤트 의존성이 만족되었다는 것을 채널이 소비자에게 알리도록 지시하며, 분리적 의미는 어느 명시된 이벤트 의존성이 만족되었을 때 채널이 소비자에게 알리도록 지시한다. 디스패칭 모듈은 이벤트가 언제 소비자에게 배달될지를 결정하며, 시스템의 스케줄링 서비스와 공동으로 작업을 수행한다[8].



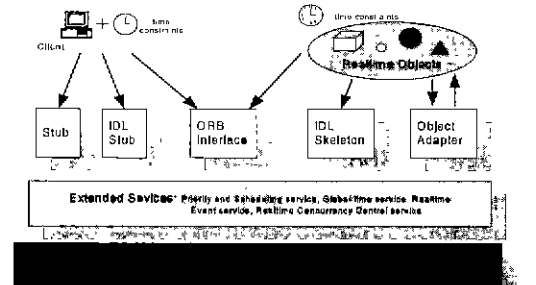
(그림 7) 실시간 이벤트 서비스 구조

4.2 NRaD 실시간 CORBA

4.2.1 개요

NRaD(US Navy Research and Development Laboratories) 실시간 CORBA는 Rhode Island 대학과 미 해군 NRaD 연구소, 그리고 MITRE사의 공동 작업으로 개발된 시스템으로, 다중쓰레드를 지원하는 Iona의 Orbix_2.0.1 CORBA를 사용하였으며 POSIX(Portable Operationg System Interface) 쓰레드를 갖는 Solrais2.5에서 구현되었다. 이는 CORBA 시스템 내의 동적 종단간 시간제약을 표현하고 강요할 수 있도록 설계되어있으며, 클라이언트나 서버가 추가 또는 삭제되고, 시간제약이 변경될 수 있는 환경을 제공한다[9], [10].

NRaD 동적 실시간 CORBA 시스템의 구성요소는 실시간 운영체제에서 동작하는 실시간 데몬 프로세스, 타입 및 IDL 정의, 클라이언트와 서버 코드의 연결을 위한 실시간 라이브러리로 이루어지며, CORBA의 기본 ORB와 달리 클럭 동기화를 위한 전역 시간 서비스(Global Time Service)와 제한된 메시지 지연을 보장하기 위한 지연 서비스(latency service)를 포함한다.



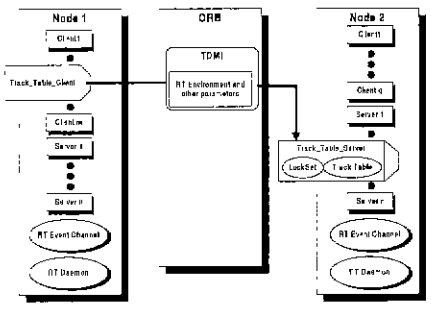
(그림 9) 동적 실시간 CORBA 시스템

4.2.2 TDMI

NRaD 실시간 CORBA는 전역 우선순위, 실시

간 이벤트, 그리고 병행제어 서비스가 새로이 추가되었으며, 실시간 파라메터를 표현할 수 있는 몇몇의 새로운 IDL 타입을 포함하고 있다. 그리고 이러한 구성 요소들은 실시간 제약이 CORBA 시스템에 의해서 강요되고 클라이언트의 CORBA 메소드 호출에 의해 표현될 수 있는 TDMI(Timed Distributed Method Invocations)를 지원한다. TDMI는 마감시간, 중요성, QoS 등과 같은 실시간 제약 정보를 갖는 서버 객체에 대한 클라이언트의 요청이다. NRaD 실시간 CORBA는 TDMI를 지원하기 위해 시간 제약을 표현하는 실시간 라이브러리를 포함하고, ORB와 객체 서비스를 확장하였다. 확장된 객체 서비스에는 EDF(Earliest Deadline First) 스케줄링을 사용하는 전역 우선순위 서비스(global priority service), 우선순위 계승의 잠금 형태를 통해 서버 객체의 일관성을 보장하는 CORBA 실시간 병행 제어 서비스(real-time concurrency control service), 실시간 이벤트 기반의 동기화와 제약을 허용하는 실시간 이벤트 서비스(real-time event service)가 있다[13].

TDMI를 통해 클라이언트는 실시간 제약을 표현할 수 있으며, TDMI는 중요성, 마감시간, 주기 등과 같은 속성을 표현하기 위해 *RT_Environment* 구조체와 *RT_Manager* 클래스를 사용한다. 이러한 구조를 클라이언트의 요청에 덧붙임으로써 종단간의 모든 곳에서 TDMI 요구사항을 강요할 수 있다[10], [11].



(그림 10) NRaD 실시간 CORBA의 TDMI

4.2.3 전역 우선순위 서비스와 실시간 스케줄링

동적 실시간 스케줄링은 실시간 운영체제의 스케줄러와 연동해서 수행된다. 각 클라이언트는 전역 우선순위 서비스와 스케줄링 파라미터를 참조하여 전역 우선순위를 차례대로 할당받는다. 이러한 우선순위는 실행 중에도 변화될 수 있어 일시적 우선순위(transient priority)라고 하며 *RT_Environment*의 정보로부터 얻어진 정수값을 사용한다.

전역 우선순위 서비스는 일시적 우선순위가 시스템내의 다른 일시적 우선순위에 대해 상대적 의미를 가지게 한다. 이는 지역 시스템에서 우선순위가 할당되는 것처럼 전체 실시간 CORBA 도메인 내에서 우선순위가 할당되는 것을 뜻한다. 한편 각 시스템에 있는 실시간 데몬은 일시적 우선순위를 실시간 데몬이 관리하는 다수의 스케줄 가능한 개체들로 매핑 한다. 예를 들어, Solaris 시스템상의 실시간 데몬은 클라이언트의 일시적 우선순위를 60 여개의 실시간 우선순위중 하나로 매핑한다[9], [11].

4.2.4 실시간 병행 제어 서비스

표준 CORBA는 클라이언트의 서버에 대한 일관된 접근을 유지하기 위해 병행 제어 서비스를 제공하며, NRaD 동적 실시간 CORBA 시스템은 우선순위 계승을 포함한 실시간 병행 제어 서비스를 제공한다. 실시간 병행 제어 서비스는 TDMI 요청을 통해 이루어지며 동작 과정은 다음과 같다.

TDMI 요청이 실시간 동시성 제어 서비스를 통해 자원에 대한 잠금(lock)을 요청했을 때, 요청한 TDMI의 실행 우선순위는 그 자원에 대한 잠금을 가진 모든 TDMI의 우선순위들과 비교되어, 잠금을 가진 TDMI의 우선순위가 요청된 TDMI의 우선순위보다 더 낮은 경우 잠금을 가진 TDMI의 우선순위는 요청된 TDMI의 우선순위로 높여지고 요청된 TDMI는 실행이 중단된다. 그 후 잠금이

해제될 때마다, 해제되는 TDMI는 블럭된 TDMI의 우선순위중 가장 높은 우선순위를 자신의 우선순위로 재 설정하며, 더 이상 더 높은 우선순위의 TDMI가 블럭되어 있지 않다면, 원래 자신의 우선순위를 회복한다. 그런 다음 지금 실행 가능한 가장 높은 우선순위의 TDMI가 해당하는 자원에 잠금을 걸고 실행을 계속한다[9], [10].

4.2.5 실시간 이벤트 서비스

실시간 제약의 표현 및 강요, 그리고 동기화를 제공할 수 있게 된 것은 실시간 이벤트의 사용이다. CORBA 2.0의 이벤트 서비스는 CORBA 시스템에서의 네이밍 이벤트의 교환을 허용한다. 예를 들면, 클라이언트는 CORBA 이벤트를 최초로 생성한 다른 클라이언트를 기다림으로써 다른 클라이언트와 동기화를 한다. NRaD의 실시간 CORBA는 이벤트의 전달을 우선순위화하고 이벤트가 발생되는 시간을 전달하는 실시간 이벤트 서비스를 제공한다. 이벤트의 우선순위는 전역 우선순위 서비스에 의해 설정된 공급자와 소비자의 전역 우선순위에 바탕을 두었으며, 이벤트 발생 시간은 이벤트와 연관된 시간 제약의 표현을 허용하기 위해 사용된다.

NRaD 실시간 CORBA의 실시간 이벤트 서비스는 IP 멀티캐스팅과 지역 운영체제내의 멀티쓰레딩을 이용해 구현되었다. 각 시스템은 실시간 데몬안에 실시간 이벤트 채널을 가지며, 각 실시간 이벤트는 멀티캐스트 그룹을 위한 IP 주소로 매핑되는 단일의 이벤트 ID를 가진다. 공급자는 멀티캐스팅을 이용해 실시간 이벤트를 해당하는 이벤트 채널로 전송하며, 이벤트 소비자는 실시간 이벤트의 전달을 기다리거나 이벤트를 가져오기 위해 지역 실시간 이벤트 채널을 호출할 수 있다. 실시간 이벤트 채널은 입력되는 이벤트들을 우선순위에 따라 정렬하여 소비자가 높은 우선순위의 이벤트를 먼저 받도록 해준다[9], [10], [11].

4.3 기타 실시간 CORBA

4.3.1 실시간 시스템을 위한 Orbix

실시간 시스템을 위한 Orbix는 Iona 사에 의해 개발된 Orbix ORB 시스템을 실시간 운영체제 위에 탑재하는 방법이다. 그러나 이는 실시간 운영체제 위에 비 실시간 ORB 시스템을 탑재하였기 때문에 실시간 운영체제의 실시간 특성들을 활용하지 못 할 뿐만 아니라, 분산시스템의 종단간 시간 제약을 충분히 강요할 수 없다[12].

4.3.2 Fast CORBA

Lockheed Martin사는 현존하는 ORB를 수정하여 보다 빨리 수행되고, 보다 빠른 반응 시간을 가지는 CORBA를 설계하였다. 특히 CORBA에서 동적 호출 인터페이스와 같은 기능을 제거하고 클라이언트와 서버 사이의 고정된 점대점 연결을 허용하는 것과 같은 특별한 프로토콜을 수용 하므로써 성능을 향상시켰다. 그러나, 이로 인하여 CORBA가 가지는 장점인 유연성은 감소되었으며, 또한 실시간 CORBA가 지녀야할 종단간 시간제약을 강요할 수 있는 수단도 부족하다 [11].

4.3.3 CHROUS/COOL 실시간 CORBA

CHORUS/COOL 실시간 CORBA는 Chorus 사에 의해 개발되고 있는 융통성 있는 실시간 ORB 시스템이다. 이는 자원관리 정책과 메카니즘을 철저히 분리하였으며, 응용프로그램이 운영체제 수준의 자원을 제어할 수 있는 방법을 제공한다. CHORUS/COOL ORB는 융통성 있는 바인딩 구조를 제공하는데 이는 응용프로그래머가 커스터마이즈(customized) 또는 동적 객체 바인딩을 정의할 수 있도록 허용한다. CHORUS/COOL ORB는 독립적인 실시간 CORBA 시스템이 아니며, CHORUS 운영체제에 많은 부분을 의존하고 있다. 따라서 ORB 시스템은 운영체제의 바로 위에서 최소한의 오버헤드만을 가진다[11].

4.3.4 New Attack Submarine 실시간 CORBA

NUWC(Naval Undersea Warfare Center)와 Lockheed Martin 사는 미군의 신형 공격형 잠수함 시스템(US Navy's New Attack Submarine : NSSN) C3I를 위하여 실시간CORBA 기술을 개발하였다. 이는 지연(latency)의 진원지를 규명하는 것 이외에도 실시간 CORBA 동작 시에 요구되는 지연평가를 수행할 수 있는 CORBA 지연 서버를 제안하였다 [13].

5. 결 론

본 논문에서는 OMG RT CORBA SIG의 실시간 CORBA에 대한 요구사항과 각 대학, 연구소, 기업의 연구 개발 동향을 살펴보았다. 실시간 CORBA에 확장되거나 또는 추가되는 기능들은 주로 사용자 수준의 QoS 명세화, 종단간 QoS 강요, 전역 우선순위, 실시간 스케줄링, 실시간 이벤트, 실시간 병행 수행제어, 실시간 통신프로토콜 등이 있다.

워싱턴 대학의 TAO는 고성능의 이식성이 좋은 실시간 CORBA를 목표로 하고있으며, 구성요소로는 기가비트 I/O 서브시스템, RIOP, QoS 요구를 명시하는 기능, 실시간 스케줄링 서비스, 실시간 이벤트 서비스를 가진 실시간 객체 아답터를 가지고 있다. NRaD 실시간 CORBA는 종단간 QoS 보장을 위해 TDMI를 사용하며, 이러한 TDMI를 지원하기 위해 실시간 이벤트 서비스, 실시간 전역 우선순위 서비스 및 실시간 스케줄링, 실시간 병행 제어 서비스가 추가되었다.

일반적으로 실시간 시스템은 개발과 유지에 많은 비용이 들며, 특정 분야에만 적용되는 한계를 가지고 있었다. 이러한 점에서 실시간 CORBA가 제공하는 융통성과 적응성은 저렴한 비용과 짧은 기간 내에 실시간 시스템 개발을 가능케 할 것으로 예측되고 있다. 그리고, 향후 실시간 CORBA는 결함 허용, 보안, 모니터링, 데이터베이스 연동 기능 등이 확장 또는 추가되는 방향으로 연구가 진행될 것이다.

참고문헌

[1] Object Management Group, A Discussion of The Object Management Architecture, http://www.omg.org/library/omaindx.htm, January, 1997.

[2] Object Management Group, The Common Object Request Broker: Architecture and Specification Revision 2.2, http://www.omg.org /corba/c2indx.htm, February, 1998.

[3] Object Management Group, CORBA Services : Common Object Services Specification, http://www.omg.org/corba/csindx.htm, November, 1997.

[4] OMG Realtime Platform SIG, White Paper on Realtime CORBA, http://www.omg.org/realtime/real-time_whitepapers.html, December, 1996.

[5] OMG Realtime Platform SIG, Realtime CORBA 1.0 Request For Proposal, http:// www.omg.org/library/schedule/Realtime_CORBA_1.0_RFP.htm, September, 1997.

[6] D. Schmidt, D. Levine, S. Mungee, "The Design of the TAO Real-Time Object Request Broker", Computer Communications Journal, http://www.cs.wustl.edu/~schmidt/corba-research-realtime.html, October, 1997.

[7] A. Gokhale, D. Schmidt, "Evaluating the Performance of Demultiplexing Strategies for Real-time CORBA", In Proceedings of GLOBECOM '97, http://www.cs.wustl.edu/ ~schmidt/corba-research-realtime.html, November, 1997.

[8] T. Harrison, D. Levine, D. Schmidt, "The Design and Performance of a Real-time

CORBA Event Service", In Proceedings of OOPSLA '97, http://www.cs.wustl.edu/ ~schmidt/ corba-research-realtime.html, October, 1997.

[9] V. Wolfe, L. DiPippo, R. Ginis, M. Squadrito, S. Wohlever, I. Zykh, R. Johnston, "Expressing and Enforcing Timing Constraints in a Dynamic Real-Time CORBA System", http://www.cs. uri.edu/rtsorac/publications.html, June, 1997.

[10] G. Cooper, L. DiPippo, L. Esibov, R. Ginis, R. Johnston, P. Kortman, P. Krupp, J. Mauer, M. Squadrito, B. Thuraisingham, S. Wohlever, V. Wolfe, "Real-Time CORBA Developement at MITRE, NRaD, Tri-Pacific and URI", In Proceedings of IEEE Workshop on Middleware for Distributed Real-time Systems and Services", pp. 69-74, December, 1997.

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

[12] Iona Technologies, Orbix for Real-Time Systems White Paper, http://www.iona.com/ support/whitepapers/orbixrt/index.html, August, 1997.

[13] R. Ginis, V. Wolfe, "The Design of an Open System with Distributed Real-Time Requirements", In Proceedings of the second IEEE Real-time Technology and Applications Symposium, pp. 82-90, June, 1996.

정 찬 균

1998년 경희대학교 전자계산공학과 (공학사)
1998년-현재 경희대학교 전자계산공학과 (석사과정)
관심분야 : 실시간 운영체제, 실시간 CORBA

이 승 룡

1978년 고려대학교 재료공학과 (공학사)
1986년 Illinois Institute of Technology 전산학과 (석사)
1991년 Illinois Institute of Technology 전산학과 (박사)
1992년-1993년 Governors State University, Illinois 조교수
1993년-현재 경희대학교 전자계산공학과 부교수
관심분야 : 실시간 시스템, 실시간 고장허용시스템, 멀티미디어 시스템, 실시간 CORBA

이 병 호

1980년 서울대학교 자원공학과 (이학사)
1980년-1981년 동력자원연구소 (연구원)
1984년 서울대학교 자원공학과 (이학석사)
1987년-1994년 GTRI 연구원
1992년 Georgia Institute of Technology 전산학 박사
1994년-1995 동양 SHL 부서장
1995년-현재 쌍용투자증권 이사
관심분야 : Distributed Computing, Fault-tolerant Systems, Operating Systems, Multimedia Systems