A Study on Next Generation CORBA Architecture
(차세대 CORBA 구조 연구)

이세훈* 고희창** 손완기***
(Seihoon Lee) (Heechang Koh) (Wanki Son)

ABSTRACT

CORBA specification has improved continuously and CORBA 2.3 spec. is pronounced in 1998 by OMG. CORBA environment help user to develop application because of providing various services and to guarantee transparency about distributed and heterogeneous system. But, CORBA can't provide usability than COM and Active X, and is not suitable to realtime application, and has the difficulty of integrating JAVA and Internet and can't provide the interoperability among other's component models.

In this paper, we analyses many elements which next generation CORBA architecture has. Next generation CORBA architecture will need script-language for rapidly application development and usability, enhanced component model, and the additional function in realtime and embedded environment. Also, it will be the architecture to support for legacy systems and to integrate with internet harmoniously.

요 약

OMG에서 제안한 CORBA는 1998년에 CORBA 2.3 버전이 발표될 때까지 많은 발전을 해왔다. CORBA 환경은 사용자에게 시스템의 분산과 이질성에 대한 투명성을 보장, 응용 개발에 필요한 다양한 서비스를 제공함으로써, 응용개발을 보다 용이하게 한다. 그러나 기존의 CORBA는 다른 컴포넌트 표준인 COM, 액티브 X보다 사용자에게 편리성을 제공하지 못하는 점과 실시간 응용에 부적합하고 자바· 인터넷과의 통합이 어려우며 다른 객체 모델과의 호환에도 많은 문제점을 가지고 있다.

본 연구에서는 차세대 CORBA 구조가 갖추어야 할 여러 가지 요소들을 분석한다. 차세대 CORBA 구조는 사용자의 편리성, 빠른 응용 개발을 위한 스크립트언어의 도입과 확장된 컴포넌트의 도입이 필요하며 실시간 환경과, 내장 환경에 적합하도록 기능이 추가되어야 한다. 또한 기존의 시스템들을 지원하고 인터넷과 통합이 원활히 하기 위한 구조가 되어야 할 것이다.

1. 서론

시스템 환경의 변화 즉 다양한 기종의 컴퓨터 하드웨어와 소프트웨어의 결합, 그리고 서비스 등의 요구에 따라 소프트웨어 요구 사항도 높은 생산성, 신뢰성, 투명성 등을 요구받게 되었다[1]. 이러한 요구사항에 발맞춰 나온 것이 분산 컴퓨팅이다[2,3]. 분산 컴퓨팅은 OSF(Open Software Foundation) DCE(Distributed Computing Environment)의 출하[4]와 OMG(Object Management Group)에 의한 분산 객체 컴퓨팅 스펙의 수용 등에서 볼 수 있듯이 지난 몇 년 동안 주목할 만한 발전을 해왔다[3]. 특히 OMG의 CORBA(Common Object Request Broker Architecture)는 1991년에 CORBA 1.1버전이 발표된 후로 계속적인 발전을 거듭하여 1998년에 CORBA 2.3 버전이 발표되었다[5,6]. CORBA 환경은 사용자에게 시스템의 분산과 이질성에 대한 투명성을 보장하며,

---

※이 논문은 인하공업전문대학 교내연구비의 지원을 받아 연구되었음.

* 인하공업전문대학 전자계산기과 조교수
** 인하대학교 전자계산공학과 강사
*** 한전정보네트웍(주) 주임

응용 개발에 필요한 다양한 서비스를 제공함으로써, 응용개발을 보다 용이하게 한다[1,2,5,6].

그러나 CORBA는 COM이나 액티브 X와 달리 사용자에게 편리성을 제공하지 못하고 기존 시스템과 인터넷 및 자바와의 원활한 통합을 지원하지 못하고, 실시간 처리를 위한 QoS(Quality of Service)의 고려가 지원되지 않는 점등의 많은 취약한 부분을 내포하고 있다[7].

본 연구에서는 지금까지의 CORBA의 구조와 서비스 기능에 대해 고찰하고 그 취약점들에 대해 알아본다. 이와 더불어 차세대 CORBA가 갖추어야 할 기능들과 구조를 연구한다.

본 논문의 구성은 다음과 같다. 2장에서는 OMG에서 제안한 CORBA에 대해 살펴본다. 3장에서는 기존의 CORBA의 미흡한 부분에 대해 차세대 CORBA에 추가되어야 할 기능들에 대해 알아본다.

2. CORBA의 구조

본 장에서는 분산 객체 컴퓨팅과 분산객체 프레임워크에 대해 알아보고 CORBA의 구조와 서비스 및 기능에 대해 알아본다.

2.1 분산 객체 컴퓨팅 표준

고성능 컴퓨터와 높은 대역폭의 네트워크에 의해 분산 컴퓨팅이 가능하게 되었고 이에 따라 기업 정보 시스템은 다양한 이기종 컴퓨터 환경에서 데이터와 응용프로그램을 분산 운용하면서 프로그램의 상호 운용성과 사용자에게 분산 투명성을 제공할 수 있게 되었다[3]. 또한 하드웨어 발달과 소프트웨어 발달의 불균형이 심화되면서 소프트웨어 개발의 새로운 전환이 요구됨에 따라 1980년대 객체 지향 기술이 대두되었다. 객체지향 기술이 제공하는 추상화, 캡슐화, 상속성, 다형성 등의 특성들이 컴포넌트 프로그래밍을 가능하게 하였으며, 소프트웨어 생산성을 보장하는 개발 방법론으로 성장하였다. 객체들은 데이터와 데이터를 조작할 수 있는 메소드로 구성되어 있으며, 프로그램은 메시지에 의한 객체간 상호작용을 기술한 것으로 객체간 메시지 교환 문제를 해결한다[3].

분산 객체기술은 이러한 두 가지 기술의 장점을 효과적으로 통합하여 주는 기술이며, 개발자에게 응용 개발의 생산성 향상을 제공하고, 사용자에게 분산 환경에 투명하게 통합된 정보를 제공하여 준다[1].

이와 같은 분산 객체 기술로는 DCE의 RPC(Remote Procedure Call)와 COM의 OLE(Object Linking Embedding) 등이 있으나, RPC는 구조적 패러다임을 기반으로 하기 때문에 복잡하고 거대한 규모의 시스템을 구축하는데 한계가 있으며, OLE는 MS의 윈도우 객체간의 호환을 위한 표준으로 근본적인 문제를 해결하는데 한계가 있다[8].

- DCE

DCE는 1990년 OSF에 의해 발표된 분산 컴퓨팅 환경에서 각종 응용 프로그램의 개발, 사용 및 유지를 시켜주는 서비스이다[4]. DCE는 통신용 RPC와 셀 디렉토리 분산형 파일 서비스, 분산형 시간 서비스, 보안 서비스 그리고 멀티스레드 소프트웨어 응용을 만들어 주는 스레드로 이루어져 있다.

- COM/DCOM

COM/DCOM은 마이크로소프트사의 바이너리 표준으로서, 객체가 어떻게 상호 작용할 것인지를 정의한다. DCOM은 COM이 네트워크에 걸쳐 분산되어 있는 것을 말한다. DCOM은 COM에서 단순히 객체를 분산시키는 것뿐만 아니라 보안과 멀티스레딩도 지원한다

2.2 CORBA 구조

OMG는 컴포넌트의 재사용, 상호운용성(interoperability), 이식성(portability)을 실현하는 분산 컴퓨팅 환경에서의 응용 통합을 위한 객체 기술에 기반을 둔 표준 프레임워크를 제안하였는데, 이것이 바로 OMA(Object Management Architecture)이다[5,6,7].

OMA는 그림 1과 같은 네 개의 컴포넌트로 구성되어있다. 소프트웨어 컴포넌트 환경으로 클라이언트의 요구를 목표 객체에 전달하고, 그

처리 결과를 클라이언트에게 보내주는 역할을 하는 ORB와 모든 컴포넌트들에 의해서 요구되어지는 시스템 수준의 서비스 집합인 CORBA 객체 공통 서비스(service), 응용 객체들에게 직접 서비스를 제공하기 위한 컴포넌트들의 집합인 CORBA 공통 기능(facility), 그리고 사용자 응용에서 기술하는 컴포넌트인 응용 객체로 이루어진다[6,8].

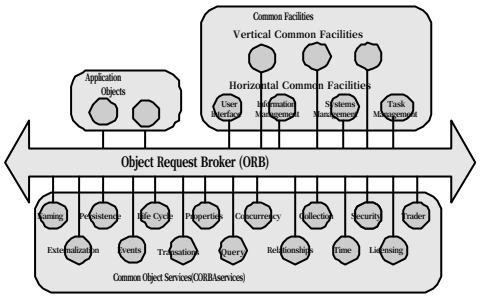

그림 1. OMA 컴포넌트

■ ORB

ORB(Object Request Broker)는 CORBA 객체간의 상호 작용을 위한 객체 버스로써, 클라이언트의 요구를 서버 객체에게 전달하고, 서버 객체의 실행 결과를 다시 클라이언트에게 전달한다. 이때 사용자에 객체 위치, 시스템 실행 환경, 통신 메커니즘 등에 대한 투명성을 제공한다. 그림 2는 ORB 구조이다.

ORB의 기능은 정적/동적 메소드 호출, 고수준 언어 바인딩, 메타 데이터 제공, 위치 투명성 등을 제공한다[2].

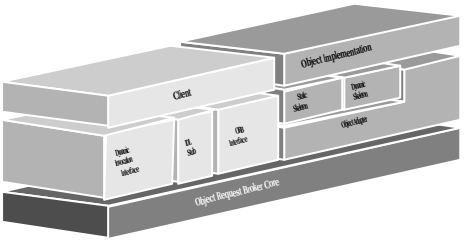

그림 2. ORB의 구조

2.3 CORBA 서비스 및 공통 기능

CORBA 하위 계층에 관련한 서비스를 제공하는 공통 객체 서비스와 상위 계층에 관련한 여러 기능을 제공하는 CORBA 공통 기능은 CORBA 환경에서 응용을 개발하는데 있어서 필요한 여러 기능을 제공하는 서비스들의 집합체이다[9].

■ CORBA 서비스

공통 객체 서비스는 분산 객체 시스템을 구축하는데 있어서 필요한 기본적인 서비스들의 집합이다. 공통 객체 서비스는 이미 채택된 서비스와 현재 준비중인 서비스를 포함하여 총15개의 서비스로 구성되어 있다. 객체 서비스들은 분산 객체 시스템을 위한 기본적인 설계 단위로써 분산 컴포넌트를 생성하기 위해 사용된다. 공통 객체 서비스는 다른 응용 객체와 마찬가지로 ORB에 접속되고, 객체 서비스의 각 구성요소도 OMG IDL에서 기술된 인터페이스를 갖고 있으며, ORB를 통해서 클라이언트에 서비스를 제공한다. CORBA서비스에는 네이밍 서비스, 트레이더 서비스, 이벤트 서비스, 보안 서비스, 동시성 제어 서비스, 특성 서비스, 트랜잭션 서비스, 관계 서비스, 질의 서비스, 지속성 서비스, 생명주기 서비스, 라이센스 서비스 등이 있다[8,9].

■ CORBA 공통 기능

CORBA 공통 기능은 응용의 기능을 지원하기 위한 소프트웨어들의 집합이다. 따라서, 응용 분야에서 공통적으로 사용하는 상위 차원의 서비스들로서, 응용 프로그램 차원의 상호호환 기능에 중점을 두고 있다[2].

CORBA 공통 기능에는 사용자 인터페이스 기능을 강조하는 복합 표현 기능(Compound Presentation Facility), 시스템 관리 기능(System Management Facility)이 있고, 응용 프로그램 지원 기능 부분을 강조하는 데이터 교환 기능(Data Interchange Facility), 복합문서 교환 기능(Compound Document Exchange Facility)이 있다.

그리고, CORBA 공통 기능에 대해서 OMG

는 모든 도메인에 필요한 기능을 제공하는 수평 기능과 특정 분야의 응용 시스템을 구현하는데 필요한 기능들을 제공하는 수직 기능으로 나누고 있다.

3. 차세대 CORBA 구조

본 장에서는 기존 CORBA에서 미흡했던 점을 해결하기 위해 차세대 CORBA에 추가되어야 할 기능들에 대해 분석한다.

3.1 차세대 CORBA 구조 개요

객체 컴포넌트 표준으로 자리잡고 있는 CORBA에 대해 OMG에서는 차세대 CORBA를 위한 스펙 작업이 진행되고 있다.

기존의 CORBA는 다른 컴포넌트 표준인 COM, 액티브 X보다 사용자에게 사용의 편리성을 제공하지 못하는 점과 실시간 응용에 부적합하고 자바와 인터넷과의 통합이 어려우며 다른 객체 모델과의 호환에도 많은 문제점을 가지고 있다. 뿐만 아니라 플러그 앤 플레이 기능을 지원하지 못한다. 이러한 기존의 CORBA의 문제점를 보완하고 더욱 분산 객체 처리에 맞는 표준을 만들기 위해 차세대 CORBA 구조에 대한 연구가 활발히 진행되고 있다.

차세대 CORBA는 다음과 같은 기능이 필요하다. 사용자에게 편리성을 주기 위한 스크립트 언어의 개발이나 더욱더 확장된 CORBA 컴포넌트의 도입이 요구된다. 이것은 쉬운 툴과 인터페이스를 제공해 빠른 응용 개발을 지원할 수 있다. 또한 CORBA와 비동기식의 메시징, 그리고 QoS, 메시징 시스템을 도입하여 실시간 처리에 적합하도록 하는 기능이 필요하다. 뿐만 아니라 기존의 환경들을 위해 지원하고 완벽하게 인터넷과 통합이 가능할 수 있는 다른 기술이 필요할 것이다.

3.2 분산된 컴포넌트의 지원

기존의 CORBA는 분산된 객체를 지원하기에는 매우 유용하다. 그러나 CORBA는 사회의 흐름에 따라 더욱 사용하기 편리한 확장된 컴포넌트를 필요하고 이러한 분산된 컴포넌트를 지원하기 위해서는 다음과 같은 기능들이 제시되어야 한다.

▪ 다중 인터페이스 및 복합 기능

기존의 CORBA는 동일한 객체 내에 여러 개의 서비스를 묶을 수 없고 하나의 인터페이스는 버전관리의 기능을 지원하지 못하는 단점이 있다. 이로 인해 프로그램의 유연성을 제한하고 프로그래머가 효율적으로 작업하지 못하며 프로그램의 유지보수가 어렵다.

분산된 객체의 버전이나 프로그램의 유연성을 제공하기 위해서는 여러 개의 IDL인터페이스를 하나의 컴포넌트로 구성하는 다중 인터페이스의 개념이 필요하다. 다중 인터페이스는 버전기능이나 COM 객체 모델을 효과적으로 지원가능하며, 생성된 객체를 지원하는 합성 기능의 생성이 가능하게 한다.

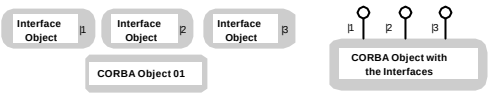

그림 3. 단일 인터페이스와 다중 인터페이스

즉, 그림 3과 같이 다중 인터페이스 정의를 사용함으로써 객체가 논리적으로 구별되는 서비스로 구성되기 위한 방법을 제공할 수 있으며, 인터페이스 정의, 연산의 리스트 등을 기반으로 하여 객체 함수에 대한 접근과 투명성을 제어 할 수 있다. 또한 프로그래머의 효율성과 프로그램의 유지보수가 향상될 수 있다.

▪ 값에 의한 객체 전달

응용 개발은 데이터의 집합을 중심으로 이루어지며, 또한 클라이언트/서버 환경의 응용 또한 데이터의 전달에 의해 동작하게 된다. 그러나 이러한 데이터 집합의 전달은 네트워크를 기반으로 이루어지기 때문에 네트워크의 성능에 상당히 의존하게 된다

일반적으로 CORBA에서는 클라이언트 측과 서버 측이 서로 다른 주소 공간에 존재하기 때문에 이러한 데이터의 전송을 위해 레퍼런스

전달(pass-by-reference)방식을 사용한다. 레퍼런스 전달 방식은 클라이언트가 어떤 객체내의 연산이나 데이터에 접근하려면 클라이언트가 전달받은 객체 참조를 사용하여 작업을 하게 된다. 이러한 객체에 대한 접근이 네트워크를 통해서 이루어지기 때문에 만약 연속데이터를 전달받게 된다면 네트워크의 성능에 상당한 제약을 받게 된다. 이러한 레퍼런스 전달 방식의 문제점은 값에 의한 객체 전달 방식을 사용함으로써 해결이 가능할 것이다. 값에 의한 객체 전달 방식으로 객체를 전달한다는 것은 서버측이 클라이언트 측으로 객체의 복사본을 전달한다는 의미이다. 수신자는 자신의 연산 공간의 객체를 가지고 작업할 수 있는 것이다. 객체는 수신자 측의 프로세서에 의해 생성되며 객체는 서버의 응용에 의해 제어 받게 된다. 그림 4는 값에 의한 객체 전달의 동작 과정이다

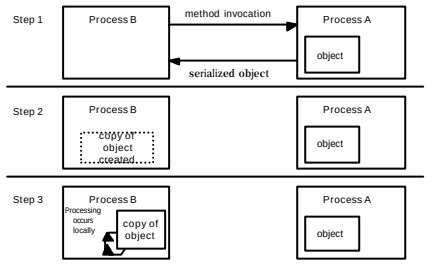

그림 4. 값에 의한 객체 전달

이로써 객체에 대한 접근이 클라이언트 측에서 일어나기 때문에 빠르고 간단하며 네트워크 트래픽은 고려하지 않아도 된다. 또한 이러한 값에 의한 객체 전달은 프로그래머가 CORBA를 더욱 유연하게 자바 같은 프로그래밍언어로 통합되는 것을 도와줄 수 있다.

■ CORBA 컴포넌트 모델

일반적인 응용은 다양한 방법으로 설계가 가능하며 이러한 응용은 하나의 큰 블록이 된다. 이것은 모듈 단위의 결합도가 높아 모듈 단위로 분리하기가 매우 어려우며 이것은 코드의 재사용이나 확장에 많은 제약을 가하게 된다. 이러한 제약을 해결하기 위해 CORBA는 확장된 컴포넌트 개념의 도입이 필요하다. 또한 컴포넌트는 Plug-and-play CORBA 객체의 개발을 위한 프레임워크를 지정하여 CORBA 기반 컴포넌트 모델과 이러한 컴포넌트를 다른 컴포넌트 모델과의 매핑을 위해 인터페이스와 메커니즘을 제공해야 한다. 이로써 객체 지향 소프트웨어 요소를 표현하는 기능과 이것들을 응용으로 만들기 위한 더욱 완벽한 메커니즘을 제공할 수 있다.

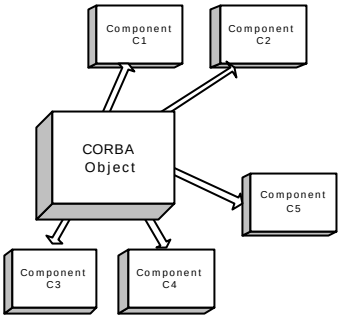

그림 5. CORBA 컴포넌트 환경

그림 5는 5개의 다른 CORBA 컴포넌트와 상호 작용를 하는 CORBA 객체를 나타내고 있다.

또한 CORBA 컴포넌트 모델은 자바 빈스(JavaBeans)와 액티브 X 컨트롤을 포함한 다른 컴포넌트 기술과의 강력한 통합이 가능하게 하여 시스템의 확장성과 재사용성이 증가할 수 있다.

■ CORBA 스크립트 언어

지금의 많은 사용자는 프로그래밍 쉽고 빠른 프로그래밍을 원한다. 이를 위해 산업계는 사용자의 요구를 충족시키는 많은 개발 툴을 내놓고 있다. 그러나 현재의 CORBA는 프로그래밍 과정이 복잡하고 컴포넌트화 하기가 매우 어렵기 때문에 많은 사용자의 요구에 충족시키지 못하고 있다. 이를 해결하기 위해서 CORBA는 RAD(Rapid Application Development environment)에 적합한 CORBA 스크립트 언어를 제공하여야 한다. CORBA 스크립트 언어는 스크립트를 사용하여 보다 쉽게 CORBA 컴포

넌트를 구성할 수 있게 하며, 또한 스크립트 언어의 사용은 복잡성을 제거하기 때문에 프로그래머가 사용하기가 간편하고 이해하기가 쉬울 것이다.

스크립트 도구는 편집하는 단계가 필요 없기 때문에 사용자가 곧바로 실행 가능한 코드를 얻을 수 있다. 또한 응용 설계자가 실행할 때에 새로운 스크립트를 생성가능 하도록 지원할 수 있다. 이와 같은 스크립트 언어의 도입은 응용 프로그래밍과 프로토타입핑의 속도를 향상시킬 수 있다.

3.3 메세징 서비스와 QoS

CORBA는 기존의 메시징 방식에는 많은 문제점을 가지고 있으며, QoS를 제공하지 못하는 단점이 있다. 이를 해결하기 위해서 차세대 CORBA는 기존의 메시징 방식, ORB등에 대한 많은 연구가 필요하다.

■ 메시징 서비스(Messaging Service)

전형적인 클라이언트/서버 응용은 실제적인 내부 구조에서 메시징에 의존한다. 이러한 메시징은 분산된 응용에서 중요한 요소이다.

CORBA는 서비스 내의 객체와 클라이언트, 그리고 서버간의 조정된 메시징 방식을 사용하고 있다. 기존의 CORBA는 메시징 시스템 상에서, 메시징 모듈은 서비스를 이용하는 모든 이용자에게 분배한다. 그러나 이 방법은 메시징 서비스의 하부 레벨의 내부적인 작업을 드러낼 뿐만 아니라 시스템을 통해서 메시징 모듈을 분배하는 것은 좋은 설계 방식이라 말할 수 없다. 이를 해결하기 위해서 메시징 서비스로의 확장이 필요하다. 이는 자신의 프로세싱 커널을 가지는 서비스가 되고 이를 요구하는 다른 서비스에 대여하는 방식이다.

또한 메시징 서비스는 안정된 실행 환경과 신뢰성을 제공하는 방법이 필요하다. 클라이언트와 서버간의 연결이 네트워크를 통해 이루어지기 때문에 네트워크가 느리거나 네트워크 서버가 고장이 발생한다면 서버 응용을 이용할 수 없다. 이와 같은 경우를 위해 메시지를 관리하고 서버에 요구를 전송하는 담당하는 응답 관리 인터페이스를 사용하는 응답 관리하는 서비스가 필요할 것이다. 이 서비스는 네트워크 트래픽으로 인해 응답이 도착하기 전에 종료될 수 있는 클라이언트도 관리한다.

마지막으로 메시징 서비스는 요구를 생성하고 요구에 응답할 때 QoS를 지정할 수 있도록 클라이언트와 서버에 기능을 제공할 수 있으며 마지막으로 전달되는 요구의 순서를 제어하기 위한 기능도 제공한다. 이러한 QoS를 위한 기능을 제공함으로써 더욱 신뢰성 있는 메시징 서비스를 할 수 있다. 메시징 서비스는 QoS를 위해 Acknowledgment level, Time-to-Live, Priority, Cost, Delivery reliability, Routing, Ordering 등의 사항들을 지정할 수 있게 하여야 한다.

■ 최소 CORBA

기존의 CORBA는 내장(embedded) 환경에서 사용되기에는 코드의 크기가 크고 불필요한 기능도 많이 가지고 있다. 그러므로 차세대 CORBA에서는 코드의 크기를 줄이고 불필요한 기능을 제거한 최소(Minimum) CORBA의 개념이 필요하다. 이것은 내장 환경에서 사용되기 위한 최소한의 CORBA 스펙으로써 내장 시스템 시장에서 CORBA의 사용을 확산시킬 것이다.

■ 실시간 CORBA

현재는 다양한 멀티미디어와 급속한 네트워크의 발전으로 인해 실시간 서비스에 대한 요구가 증가되고 있다. 그러나 실시간 서비스를 위해 CORBA를 사용하기에는 자원 관리나 우선 순위 스케줄링에 관한 기능 등의 많은 제약사항을 가지고 있다. 이러한 실시간 서비스를 위해서는 확장된 ORB의 정의가 필요하다. 확장된 ORB는 종단간의 원활하고 효율적인 통신을 위해 ORB 자원을 제어하고 고정된 우선 순위 스케줄링을 포함하여야 한다.

■ 비동기 메시징

현재의 CORBA는 동기화, 연기된 동기화,

one-way 메시징 방식을 지원한다. 그러나 기존의 메시징 방식은 클라이언트에게 비동기 메시징 방식을 제공하지 못하므로 순서화와 QoS를 처리하기 위한 비동기 메시징 방식의 채택이 필요하다. 이는 두 가지 요소로 구분할 수 있다. 먼저 QoS 동의를 위한 레벨과 비동기 메소드 호출방식을 지원하기 위한 IDL의 변경과 클라이언트와 ORB가 객체상의 메소드를 비동기적으로 호출하는 메소드를 정의하여야 하며, 분산된 객체 시스템에서 순서화와 QoS 요구들을 포함한 비동기 메시지를 관리하여야 한다. 이것은 향상된 통신과 워크플로우의 효율성을 제공할 수 있다.

3.4 그 밖의 다른 기술들

여기서는 자바와 인터넷 통합 그리고 기존 시스템들을 지원하기 위해 제시되어야 하는 것들에 대해 알아본다.

■ 영속 상태 서비스

기존의 영속 객체 서비스(Persistent Object Service)는 확장성이 떨어지고 벤더들이 서비스를 이용하여 개발하기가 힘들고, 영속 상태 서비스(Persistent State Service)는 상호 운용성을 무시한 설계 상의 한계 때문에 전세계적으로 인정받지 않고 있다.

영속 상태는 CORBA 객체 상태의 영속성을 제공하는 영속 상태 서비스를 사용하는 CORBA 객체이다. 영속 상태를 구현하기 위하여 객체의 상태를 기술하는 방법이 필요하다. 상태는 실제 객체뿐만 아니라 부수적으로 많은 속성들도 저장할 필요가 있다. 영속 상태 서비스는 객체의 활성화된 상태를 저장하기 때문에 오류 허용의 기반을 제공할 수 있다. 이와 같은 영속 상태 서비스를 위해서는 영속 상태를 구현하기 위한 인터페이스가 필요하고, 인터페이스에는 하나이상의 자료저장소를 제공되어야 한다. 또한 인터페이스들은 객체의 구현과 영속성 관리를 하는데 자료저장시스템의 선택에 영향을 받지 않아야 하고 CORBA를 통해 많은 수의 작은 객체들을 위한 효과적인 접근 방법을 제공해야 할 것이다.

■ POA(Portable Object Adapter)

CORBA 1.0에서 정의한 BOA(Basic Object Adapter)는 ORB 환경에서는 매우 중요한 역할을 한다. 객체 어댑터는 객체 요구를 받아들이고, 서버 객체를 위치시키고, 활성화하며, 객체 인스턴스를 생성한다. 그러나 BOA는 설계에 있어서 너무나 한정성을 가지고 있고, 많은 약점을 가지고 있다. 이는 이식성과 다른 상업용 ORB와의 상호연동을 지원하는 방향으로 확장되어야 한다. 따라서 OMG는 새로운 객체 어댑터인 POA를 제시하고 있다. POA는 이식성이 있고, BOA가 가지고 있는 모든 설계 규약을 고려하여 만들었다. POA를 가지고, 상업용 ORB로 작성된 클라이언트는 다른 상업용 ORB에 의해 작성된 것과 쉽게 동작할 수 있다. 또한 모든 기업에서 정의한 명세서는 POA에 의해 적절하게 처리될 수 있다. 즉, POA가 제공하는 가장 중요한 특징은 투명한 객체 활성화 모델이다.

■ 자바의 IDL 매핑

CORBA가 컴포넌트를 지원한다고 하더라도 기존의 자바 언어로 된 컴포넌트는 사용하지 못했다. 자바의 IDL 매핑 기능의 추가는 자바 언어로부터 자동적으로 OMG IDL을 생성하도록 하여 IDL로 기술되어지지 않은 자바 컴포넌트를 통합 가능하게 할 수 있을 것이다.

■ DCE/CORBA 인터워킹

기존의 DCE 응용을 CORBA 환경으로의 통합을 지원하기 위한 기능이다. DCE 서버와 함께 CORBA 클라이언트간의 상호작용과 CORBA 서버와 함께 DCE 클라이언트간의 상호작용, DCE 서비스를 사용해 구현된 CORBA 서비스를 사용할 수 있다.

4. 결론

OMG CORBA는 1991년에 CORBA 1.1버전이 발표된 후로 1998년 CORBA 2.3 버전이 발

표될 때까지 많은 발전을 해왔다. CORBA 환경은 사용자에게 시스템의 분산과 이질성에 대한 투명성을 보장하며, 응용 개발에 필요한 다양한 서비스를 제공함으로써, 응용개발을 보다 용이하게 한다.

그러나 기존의 CORBA는 다른 컴포넌트 표준인 COM, 액티브 X보다 사용자에게 사용의 편리성을 제공하지 못하는 점과 실시간 응용에 부적합하고 자바와 인터넷과의 통합이 어려우며 다른 객체 모델과의 호환에도 많은 문제점을 가지고 있었다. 뿐만 아니라 플러그 앤 플레이 기능을 지원하지 못했었다. 이러한 이유로 CORBA표준의 개발 속도보다 사용자의 기술 습득 속도가 현저히 떨어지게 되었다. 이러한 많은 문제점으로 인해 차세대 CORBA에 대한 연구가 활발히 진행 중이다.

본 연구에서는 차세대 CORBA 구조에 대해 연구하였다. 차세대 CORBA는 사용자의 편리성, 빠른 응용 개발을 위한 스크립트언어의 도입이 필요하며, 또한 확장된 컴포넌트의 도입, 실시간 환경과, 내장 환경에 적합하도록 기능이 추가되어야 한다. 또한 기존의 시스템들을 지원하고 인터넷과 통합이 원활히 하기 위한 기능이 필요하다.

※ 참고 문헌

[1]Vinoski,S.," CORBA : Integrating Diverse Application Within Distributed Heterogeneous Enviroment," IEEE Communications Magazine, Vol. 14, No. 2, 1997

[2]Orfali,R. and Harkey,D., *Client/Server Programming with JAVA and CORBA*, John Wiley & Sons, 1997.

[3]왕창종, 이세훈, 분산 객체 컴퓨팅 기술 CORBA 프로그래밍, 도서출판 대림,1998.

[4]Rosenberry,W.,Kenney,D., and Fischer,G., *Understanding DCE*, OReilly and Associates, Inc., 1992.

[5]OMG, *The Common Object Service Specification*, OMG Document, 1997.

[6]OMG, *The Common Object Service : Architecture and Specification Revision 2.3*, OMG Document, 1998.

[7]OMG,"History of CORBA," http://www.omg.org/news/pr98/corbhist.htm, 1998.

[8]Hoque,Reaz., *CORBA 3*, IDG Books Worldwide, 1998.

[9]OMG, *The Common Object Service Specification*, OMG Document, 1997.

이세훈

1985년 인하대학교 전자계산학과 졸업(이학사)
1987년 인하대학교 대학원 전자계산학과 졸업(이학석사)
1996년 인하대학교 대학원 전자계산공학과 졸업(공학박사)
1987~1990년 해병대 전산실 분석장교
1991~1993년 비트컴퓨터 기술연구소 선임연구원
1993~현재 인하공업전문대학 전자계산기과 조교수
관심분야 : 분산객체컴퓨팅, 멀티미디어, 소프트웨어공학, 원격교육

고희창

1986년 인하대학교 전자계산학과 졸업(이학사)
1992년 인하대학교 대학원 전자계산학과 졸업(이학석사)
1996~현재 인하대학교 대학원 전자계산공학과 박사과정
1992~1997년 현대전자산업(주) 통신연구소
1996~현재 인하대학교, 인하공전 강사
관심분야 : 멀티미디어 응용, 통신망관리, 원격교육

손완기

1993년 인하대학교 전자계산공학과(공학사)
1995년 인하대학교 대학원 전자계산공학과(공학석사)
1995년~현재 한전정보네트웍(주) 주임