□신기술해설□

분산 객체 시스템

이 동 현† 황 승 구††

◈ 목 차 ◈

요 약

컴퓨팅 환경은 쉴새 없이 변하고 발전한다. 빠른 속도를 가지는 개인용 컴퓨터들의 보편화와 이들을 광범위하게 연결하고 빠른 속도로 상호간에 통신할 수 있도록 하는 네트워크의 보편화 경향은 이러한 환경을 기반으로 하는 새로운 소프트웨어를 요구한다. 멀티미디어나 그룹웨어, 분산 가상 환경 등, 새롭게 요구되는 많은 소프트웨어들은 광범위한 지역에서 여러 사용자들 간에 컴퓨터를 통한 공동 작업을 가능하게 하고, 멀티미디어 등 여러 형식의 다양한 대용량의 정보를 빠르게 주고 받을 수 있게 한다고 약속한다. 그러나 아직은 이러한 소프트웨어의 개발과 이를 수행할 수 있는 환경이 충분하지 않다 이러한 환경을 위하여 분산 시스템 소프트웨어 등에 대한 연구가 진행되고 있지만 컴퓨터간의 통신 처리에 중점을 두는 기존의 기술은 아직 효과적이고 효율적으로 이를 관리하고 그 위에서 수행될 분산 응용 프로그램의 개발을 위하여 편리하게 사용할 수 있는 개발 도구를 충분히 지원하지 못하고 있다

분산 객체 시스템은 이러한 점들을 극복하기 위한 분산 시스템 소프트웨어 개발의 새로운 방향으로서 통신 처리의 복잡성을 객체 중심적인 방법을 통하여 해결하여 객체 중심적인 개념이 제공하는 개발 및 관리의 용이성, 확장성, 재사용성 등의 장점을 분산 환경의 관리에서도 얻고자 하는데 기본적인 목적이 있다.

본 고에서는 분산 객체 시스템에 대한 소개와 그 기술적인 특성, 표준화와 관련한 현재의 기술 추세 등에 대하여 설명하며, 전자통신연구소에서 개발 중인 복제 공유 객체 시스템에 대하여 소개하고 향후의 발전 방향과 기술적인 문제점 들에 대하여 논의 한다.

1. 서 론

자원을 효과적이고 저렴하게 사용할 수 있도록 발전하여 왔다. 시분할 처리 방식은 한정적인 컴퓨터 자원을 여러 사람이 동시에 사용할 수 있도록 개발되었으며, 개인용 컴퓨터의 발전은 일반인의 컴퓨터 사용을 보편화 시켰다. 이러한

† 정회원 : 한국전자통신연구소 선임연구원
†† 정회원 · 한국전자통신연구소 책임연구원

추세는 네트워크의 발전과 더불어 네트워크로 연결된 컴퓨터 자원의 공용을 통하여 여러 사람이 개별적인 컴퓨터들을 더욱 효과적으로 사용할 수 있도록 하였다. 네트워크로 연결된 컴퓨팅 환경의 초기에는 개별 컴퓨터에서 취급하는 데이타 양에 비하여 현저히 느린 속도로 인하여 컴퓨터 자원의 공유 정도가 미약하였으나 현재와 같이 빠른 네트워크의 실현은 점차 충분한 공유를 가능하게 하고 있으며 앞으로의 발전 추세에 비추어 보면 네트워크를 통한 컴퓨터 자원의 공유는 보편화할 것으로 보인다.

컴퓨팅 환경의 변화와 더불어 사용자들이 처리하고자 하는 작업의 성격도 변화하여, 일괄 처리 방식에서 주로 행해지던 계산 결과만을 필요로 하던 정보처리 작업들은 시분할 방식에서 컴퓨터와의 상호 작용을 필요로 하는 정보가공 작업들로, 그리고 개인용 컴퓨터 환경에서는 보다 나은 사용자 인터페이스를 통한 개인들의 정보 생산 작업들로 변화하게 되었다. 고속의 네트워크 환경으로의 변화는 이제 정보의 공유 및 분배, 재사용 작업이라는 새로운 성격의 작업들이 가능하도록 하고 있으며 이에 따른 새로운 소프트웨어에 대한 요구들을 발생시키고 있다.

쉽게 사용이 가능하고 충분한 데이타의 교환이 가능한 네트워크 환경은 원거리 사용자들 간에 정보의 교환이 쉽게 이루어지도록 하여, 사용자들은 이를 바탕으로 컴퓨터를 통하여 그룹의 사용자들 간에 공동 작업이 가능한 환경을 요구하게 되었으며, 원거리의 빠른 정보 이동은 이를 통하여 원거리 교육 및 발표, 정보검색 등의 응용을 요구하게 되었다. 또한 원거리의 정보나 소프트웨어에 대한 접근의 용이성으로 이의 재사용성이 더욱 중요하여 졌다. 이에 따라 서로 다른 시스템에서 같은 정보를 사용할 수 있게 하는 일이나 서로 다른 시스템에서도 같은 소프트웨어가 수행 되도록 하는 일 등, 정보의 교환에 대한 표준화와 소프트웨어의 이식성 및 상호 운용성 등이 매우 중요하게 부각되고 있다.

분산 시스템 소프트웨어는 이러한 새로운 소프트웨어에 대한 요구를 만족 시키는 기본적인 틀을 제공하기 위하여 연구되어 왔다. 분산 시스템 소프트웨어는 분산 응용 프로그램들의 기본적인 수행 환경을 제공할 뿐만 아니라 분산 응용 프로그램 개발 도구의 기본적인 틀의 역할도 하여 분산 응용 프로그램 개발 도구들은 바탕이 되는 분산 시스템의 구조에 제한을 받게 된다. 이러한 분산 시스템 소프트웨어로서 지금까지 주로 연구되었던 부분은 두 시스템간에 안정된 통신 처리 부분들과 이에 대한 인터페이스 제공, 망의 관리, 시스템 간의 동기화, 분산 자원의 관리 등이었다. 이러한 부분들은 분산 시스템을 위한 기본적인 기술들을 많이 해결하였으며 기본적인 분산 응용 프로그램의 개발이 가능하게 되었으나 아직도 손쉬운 개발과 그 관리 및 확장, 개선 등은 매우 어려운 일로 여겨진다. 그 주된 이유는 개발 도구의 부족과 개발 도구의 바탕이 되는 분산 시스템 소프트웨어의 체계성 및 유연성, 확장성의 부족 등으로 볼 수 있다[6].

분산 응용 프로그램의 개발에 있어서 중요한 것은 분산 응용 프로그램의 모델과 이의 수행을 지원하는 분산 시스템, 그리고 모델에 준하는 구성 재료(building block)들과 이 구성 재료들을 적절히 조합하여 응용 프로그램을 개발할 수 있도록 하는 도구들이다.

분산 시스템 상에서 수행되는 분산 응용 프로그램들은 각 응용 프로그램의 특성에 따라 다양한 형태가 존재하는데 두 응용 프로그램 간의 통신 형태에 따른 기본적인 모델은, 대표적으로 요구-지원 모델(client-server model)과 대등 관계 모델(peer-to-peer)을 들 수 있다[6]. 이러한 모델들은 두 응용 프로그램이 서로 간의 통신에서 어떠한 역할을 하는가에 따라 구분해 볼 수 있는 모델이다. 이러한 기본적인 모델들은 그룹의 응용 프로그램들 간의 통신 형태에 서로 다른

형태로 응용이 가능하다.

분산 객체 시스템은 객체를 중심으로 분산 환경을 지원하는 분산 시스템으로서, 다양한 형태의 분산 응용 프로그램 모델을 이해하기 쉽게 지원할 수 있으며, 분산 객체를 구성 재료로 제공하여 이를 이용한 분산 응용 프로그램 개발 도구를 쉽게 지원할 수 있다. 또한 객체 중심 시스템이 제공하는 관리의 용이성, 시스템의 확장성 및 구성 재료의 재사용성 등의 장점을 그대로 가지므로 분산 객체 시스템은 분산 시스템의 향후 발전 방향으로 주목되고 있다.

2. 분산 객체 컴퓨팅(Distributed Object Computing)

서로 다른 컴퓨터들과 이들간의 네트워크로 구성된 분산 환경을 분산 객체를 통하여 하나의 체계적인 환경으로 제공하고, 이 환경에서 응용 프로그램의 개발과 수행 등을 할 수 있도록 하는 것이 분산 객체 컴퓨팅 환경이다. 이러한 환경에서 분산 객체를 이용하여 필요한 컴퓨터 작업을 구성하고 수행하는 것들을 분산 객체 컴퓨팅이라 한다. 이러한 환경은 구체적으로 분산 객체 시스템을 통하여 이루어 진다.

일반적으로 분산 객체란 논리적으로 하나의 객체가 분산 환경의 독립적인 여러 응용 프로그램 간에 공유되어 사용되는 객체를 말한다.

여러 응용 프로그램에 공유되는 분산 객체의 종류는 그 구현 형태에 따라 원격 객체(remote object), 복제 객체(replicated object)로 구분해 볼 수 있다[9]. 원격 객체는 한 응용 프로그램 내에 실제의 객체가 존재하면서 다른 응용 프로그램에게 그 인터페이스를 제공한다. 다른 응용 프로그램에서는 대리 객체(proxy object)를 통하여 원격 객체에 접근한다. 이 경우 객체의 정보는 하나의 객체에서 관리되므로 객체의 일관성 유지 등에 장점이 있다. 복제 객체는 동일한 정보를 가지는 객체가 각 응용 프로그램에 복제되어 유지 되는 것을 말한다. 이 경우 각 객체에 대한 접근이 빠르고 객체의 중복성으로 내고장성(fault tolerency)을 가질 수 있다. 그러나 객체의 일관성 유지가 어려운 문제가 된다.

성 객체는 필요에 따라 그 실존 위치를 옮길 수 있으므로 분산된 전체 시스템에 적절한 부하 조정을 할 수 있으며 빠른 접근성을 제공할 수 있다. 분산 객체는 이와 같은 기본적인 형태들의 조합으로 구성될 수도 있다.

분산 객체가 가져야 할 기본적인 속성으로는 영구성(persistency)와 동시성(concurrency)이 있다[7,9]. 분산 객체는 여러 개의 응용 프로그램 혹은 시스템에서 공유되는 것이므로 전체 생존 기간이 개별 응용 프로그램에 반드시 독립적이어야 한다. 또한 서로 독립된 컴퓨터에서 동시에 접근하므로 동시 호출(concurrent invocation)이 발생하고 이를 효과적으로 지원하려면 동시성 제어가 필요하다. 객체가 스스로 동작하는 제어 흐름을 가지는 경우 이를 능동 객체(active object)라고 하고 외부에서의 호출에 의하여만 동작하는 경우 이를 피동 객체(passive object)라고 한다.

분산 객체 컴퓨팅은 구체적으로 분산 객체 시스템을 통하여 이루어 진다. 분산 객체 시스템은 분산 객체 기반 시스템(Distributed Object Based System)과 분산 객체 중심 시스템(Distributed Object-Oriented System)으로 나누어 볼 수 있다[9]. 이 둘 간의 구분은 다루는 객체가 상속성이 있는가 없는가에 따라 나누어 진다. 다루고 있는 분산 객체가 상속성을 가질 경우 분산 객체 중심 시스템이라 한다

분산 객체 시스템의 일반적인 기능은 분산 객체의 생성 및 소멸과 저장 및 복원 등의 분산 객체 존재 관리와 분산 객체의 위치 투명성 제공을 위한 분산 객체 이름 관리, 분산 객체의 수정과 이에 따른 일관성 유지, 분산 객체의 소유권 관리와 이에 따른 접근 제어 등으로 볼 수 있다. 이와 같은 기능을 위하여 분산 객체 시스

템은 크게 두 가지 부분으로 나누어 볼 수 있다. 분산 객체 관리 부분과 객체 간의 통신 지원 부분이다[2].

분산 객체 시스템에서 분산 객체 관리 부분의 기능으로 객체의 존재를 관리하고 객체의 참조와 그 일관성 유지, 접근 제어 등의 기능들은 객체간의 통신 상에서 이루어지는 것이므로 객체 간의 통신을 위한 부분이 필요하다. 이 부분이 객체 간의 통신 지원 부분으로, 분산 객체 관리 부분과 독립적으로 다루어 질 수 있다. 두 부분 간의 인터페이스에서 중요한 것은 객체 간의 통신들을 응용 프로그램들 간의 통신 속에서 어떻게 다루는가 하는 점이다.

객체 간의 통신 지원 부분에서 주된 기능은 동기적 통신과 비동기적 통신을 지원하는 기능이다. 둘 이상의 구성원을 갖는 그룹의 분산 응용 프로그램을 개발하는 경우는 그룹 통신을 지원하여야 한다. 그룹 통신을 위한 중요 기능은 전송 메시지의 분배와 전송된 메시지들의 수신 순서 관리 등이다. 수신 순서 관리는 특히 복제 객체의 경우 그 일관성 유지를 위하여 반드시 보장되어야 할 부분으로 중앙 집중식 순서 관리와 토큰을 사용한 순서 관리 방법들이 주로 사용된다.

3. 분산 응용 프로그램의 모델

분산 응용 프로그램의 모델과 이 응용 프로그램을 지원하는 분산 시스템의 모델과는 밀접한 관계가 있다. 서로 간의 모델이 유사할 경우 프로그램의 개발 및 개발 도구의 제공이 용이하게 이루어질 수 있다.

분산 응용 프로그램의 모델은, 먼저 여러 응용 프로그램에서 공유하는 것이 어느 것인가에 따라 데이타 중심적(data-oriented)인 모델과 프로세스 중심적(process-oriented)인 모델로 구분할 수 있다. 데이타 중심적인 모델은 여러 응용 프로그램들의 협조(coordination)가 공유되는 데이타를 중심으로 설계되는 것을 말하며, 프로세스 중심적인 모델은 협조가 동일한 모듈의 수행에 관련하여 이루어지도록 설계되는 것을 말한다. 프로세스 중심적인 모델이 보다 세세한 동기화 과정을 설계하기 유리하다.

분산 응용 프로그램간의 통신 형태에 따라 응용 프로그램 모델을 분류하면 요구-지원 모델과 대등 관계 모델을 볼 수 있다. 요구-지원 모델에서 응용 프로그램 간의 통신은 동기적으로 이루어 진다. 통신의 시작은 요구 응용 프로그램만이 할 수 있으며 지원 응용 프로그램은 요구 응용 프로그램이 시작한 통신(서비스 요구)에 대하여 응답한다. 요구 응용 프로그램의 존재가 지원 응용 프로그램에게 알려져 있지 않으므로 지원 응용 프로그램이 먼저 통신을 요청할 수가 없다. 양쪽간의 통신은 항상 통신 순서에 준하여 수신 측이 수신 준비를 하고 있어야 가능하므로 요구-지원 모델은 서로 간의 정보 교환을 위한 통신에 관련한 설계에 주의하여야 한다. 대등 관계 모델에서는 통신 측면에서 응용 프로그램들 간의 역할 구분이 없다. 각 응용 프로그램은 대등한 관계에 있으므로 아무 때나 필요한 정보를 상대에게 전달하고 정보를 수신할 수 있다. 그러므로 대등 관계에서의 통신은 비동기적으로 이루어질 수 있다 그러므로 통신에 관련된 설계가 상대적으로 간단하고 쉽다. 그러나 요구-지원 모델은 동기적인 통신으로 가능하므로 각 응용 프로그램이 하나의 제어 흐름으로도 설계 가능하나 대등 관계 모델은 비동기적인 통신을 지원하기 위하여 적어도 두개 이상의 제어 흐름을 필요로 한다 또한 한 응용 프로그램 내의 두개의 제어 흐름간의 정보 교환을 위한 설계가 필요하다.

응용 프로그램의 모델은 분산 객체의 모델에서도 적용될 수 있다. 분산 객체는 하나의 개체(entity) 안에 데이타와 그에 대한 오퍼레이션이

같이 묶여 있으므로 분산 객체를 통하여 데이타 중심적인 모델과 프로세스 중심적인 모델은 하나의 형태로 합쳐질 수 있다.

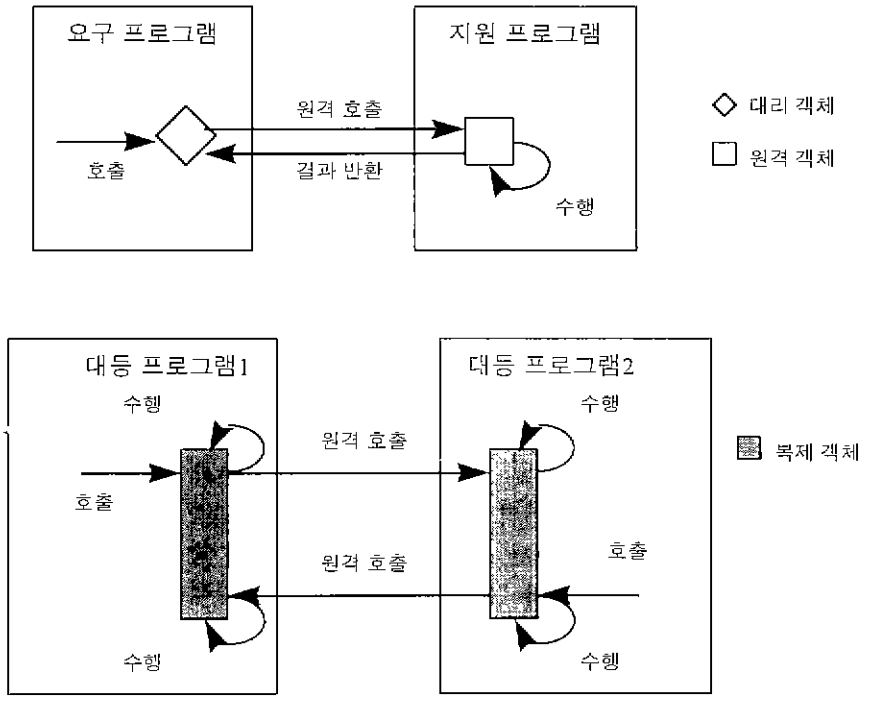
원격 객체와 대리 객체의 개념은 요구-지원 모델의 응용 프로그램을 개발하는데 적절하며 복제 객체의 개념은 대등관계 모델을 자연스럽게 지원할 수 있다[2]. 원격 객체는 대리 객체의 존재에 독립적이므로 이를 이용하여 지원 응용 프로그램을 설계하고 대리 객체는 원격 객체의 참조 및 수정을 대신하므로 이를 이용하여 쉽게 요구 응용 프로그램을 설계할 수 있다. 복제 객체들 간에 비동기적이고 대등한 통신은 대등 관계 응용 프로그램에 그대로 사용될 수 있다. 원격 객체와 복제 객체를 사용한 응용 프로그램의 구성에 대한 예는 (그림 1)과 같다.

4. RSOS(Replicated Shared Object System)

RSOS는 전자 통신 연구소에서 수행하고 있는 GIANT 과제[1] 중 그룹웨어 응용 프로그램의 수행 및 개발 환경의 제공을 목적으로 개발된 분산 객체 중심 시스템으로 복제 객체 및 원격 객체를 기본적인 분산 객체로서 제공한다[2,3,4,5]

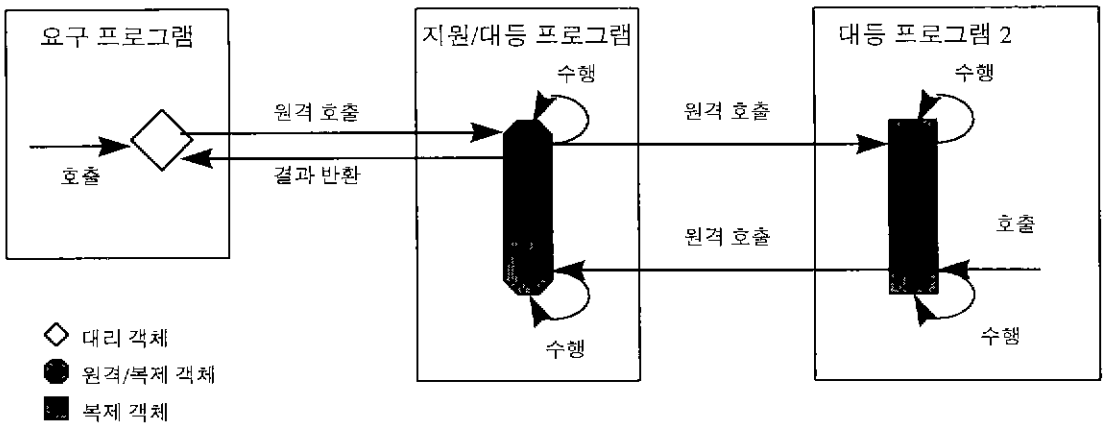
RSOS는 원격 객체와 복제 객체를 잘 조합된 형태로 제공하여 요구-지원 모델과 대등 관계 모델의 분산 응용 프로그램의 개발만이 아니라 (그림 2)와 같이 이들을 조합 시킨 형태의 응용 프로그램 개발도 유연하게 이루어질 수 있다는 것이 하나의 특징이다

RSOS가 수행 및 개발 환경으로서 지원하려 하



(그림 1) 응용 프로그램의 관계와 원격 객체, 복제 객체

[1] Gigabit Information And Networking Technology



(그림 2) 원격 객체와 복제 객체의 조합

는 그룹웨어는 분산 응용 프로그램 중에서도 응용 프로그램간의 상호 작용이 많고 응용 프로그램 간의 협력(coordination)이 매우 많이 요구되는 특징을 갖는다[10]. 또한 분산 데이타 베이스와 같이 개별적인 사용자들에게 독립적으로 서비스를 제공하는 것이 아니라 그룹의 사용자들에게 서로간의 정보를 교환하게 하는 것이므로 빠른 응답 시간이 더욱 중요하다. 그러므로 RSOS는 빠른 응답 시간을 얻을 수 있고 그룹의 대등한 응용 프로그램간의 협력을 잘 지원할 수 있는 복제 공유 객체의 지원에 많은 중점을 두었다.

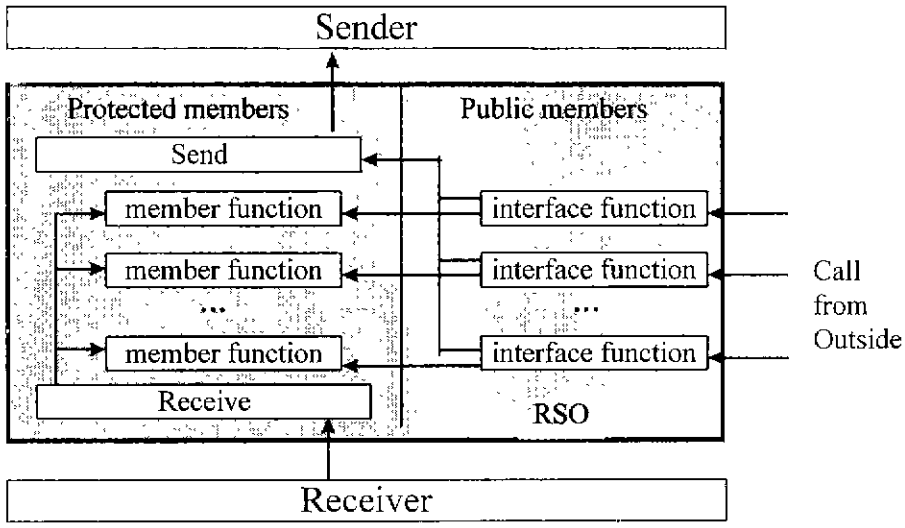
4.1 분산 객체 모델

RSOS의 분산 객체 모델은 C++ 언어의 객체 모델을 기반으로 영구성과 동시성을 가질 수 있도록 확장하여 프로그래머들이 이해하기 쉬우며, 한 프로그램 내에서 지역적인 C++의 객체와 동일한 방법으로 다룰 수 있도록 되어 있다.

C++ 객체 모델 확장을 기반으로 C++ 언어 기능의 관련 기능들을 확장하여 설계한 D++ 언어는 프로그램 상에서 RSOS 상에서 지원되는 분산 객체를 D++ 언어로 프로그램할 수 있도록 하고 있으며 이 언어의 번역기는 이 프로그램을 RSOS의 API를 포함하는 C++ 코드를 생산한다[3].

분산 객체의 한 멤버 함수가 한 응용 프로그램 내에서 호출될 때 이를 대응하는 다른 분산 객체에 멤버 함수의 호출하기 위하여 분산 객체는 <그림 3>과 같이 인터페이스 함수를 갖는다. 응용 프로그램 내에서 분산 객체의 내용을 참조하거나 수정하기 위하여 인터페이스 함수를 호출하면 인터페이스 함수는 그 내용을 다른 객체에게 전달한다. 분산 객체가 대리 객체일 경우 인터페이스 함수는 함수의 호출 결과가 반환될 때를 기다렸다가 그 값을 자신의 결과 값으로 반환한다. 분산 객체가 복제 객체일 경우 (그림 3)는 반환 값을 기다리지 않고 복제 객체의 대응하는 멤버 함수를 호출하고 그 결과를 반환한다. 즉 복제 객체의 경우 각 응용 프로그램에서 동일한 함수의 수행이 일어난다.

다른 응용 프로그램에서 분산 객체에 대한 멤버 함수의 호출이 있어서 그 내용이 전달되었을 경우는 그 메시지가 해당 분산 객체에 배달된다. 배달된 메시지의 내용에 관련하여 해당 멤버 함수가 호출, 수행된 후 메시지가 대리 객체로부터



(그림 3) 복제 객체간의 객체 수정을 위한 복제 객체의 구조

왔을 경우는 수행의 결과를 다시 반환하고 다른 복제 객체로부터 왔을 경우는 반환하지 않는다.

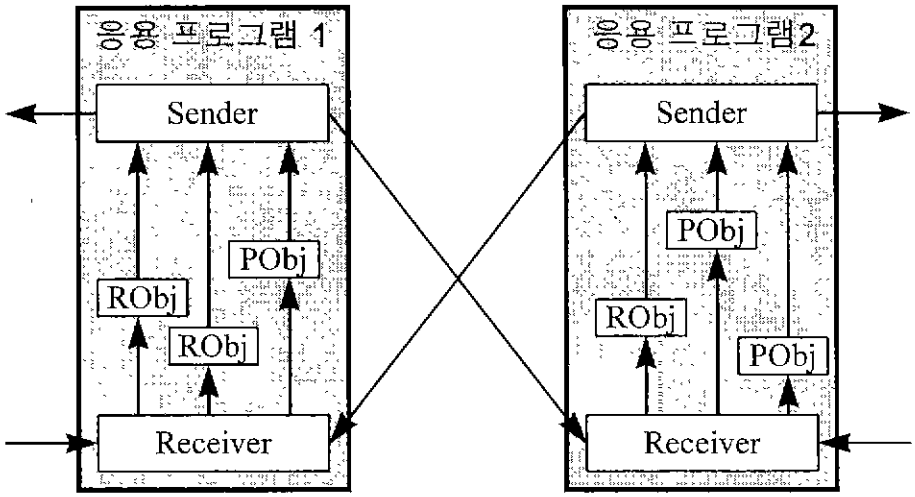
4.2 분산 객체의 관리

분산 객체는 관련된 개별 응용 프로그램들의 존재와 무관하게 존재하므로 응용 프로그램의 수행 기간 동안에만 존재하고 참조되는 응용 프로그램의 지역 객체와는 다르게 그 이름이 관리되어야 한다. RSOS에서 분산 객체는 다른 응용 프로그램에서도 알 수 있는 객체의 영구적인 이름을 가지며 객체간의 메시지 전달시 이 이름을 통하여 대응하는 객체들이 참조 된다.

객체들간의 메시지 교환을 위한 응용 프로그램의 구조는 (그림 4)와 같이 메시지를 비동기적으로 수신하고 송신하는 역할을 하는 2개의 객체, Sender 와 Receiver가 존재한다. Sender는 응용 프로그램 내의 객체들이 사용될 때 그에 관련된 메시지를 다른 응용 프로그램들에게 분배하는 역할을 하며 Receiver는 다른 응용 프로그램으로부터 수신한 메시지를 분산 객체 관리자를 통하여 해당 객체에게 배달하는 역할을 한다. Sender와 Receiver 객체는 하위의 그룹 통신 부분의 기능을 인터페이스 한다. 그러므로 그룹 통신 부분이 바뀌는 경우 이 두 객체만을 수정하여 전체 시스템을 수정할 수 있다.

4.3 그룹 통신

RSOS의 프로토타입 시스템은 TCP를 사용하여 메시지의 분배를 하고 서버를 사용하여 메시지 전체 순서를 유지하는 방법으로 구현 되었다. 현재는 RSOS의 분산 객체 관리 부분과 독립된 그룹 통신 부분을 ITU-T T.120 데이타 회의 표준화에서 규정하는 MCS(Multipoint Communication Service)[1]를 그룹 통신 부분으로 대치하여 구현 중에 있다. MCS의 통신구조는 그룹의 응용 프로그램들이 나무 구조로 연결되어 통신하는 것이며 이들간의 메시지 순서화는 최상위 MCS에서 주관하므로 결국 메시지 순서화를 위하여 서버를 사용하는 것과 같은 특징을 갖는다.



(그림 4) 분산 객체의 수정을 위한 응용 프로그램의 구조

5. 향후 발전 방향

현재 진행되고 있는 분산 객체 시스템의 한 방향은 분산 객체 시스템의 표준화로서 분산 환경에서의 표준화는 서로 다른 컴퓨터 시스템 간의 연동성의 측면에서 매우 중요하다. 현재 가장 중요하게 고려되고 있는 분산 시스템의 표준은 CORBA로서 원격 객체의 지원을 기본으로 한다.[8]

표준화에 관련된 활동 속에서 발생하는 일반적인 문제의 하나는 여러 사용자 혹은 개발자들의 공통의 요구 사항을 만족 시키면서, 개방형 시스템으로서 현재 개발된 앞선 기술, 혹은 앞으로 예상되는 진보된 기술을 얼마나 적절히 수용할 수 있는가 하는 점이다. 이러한 점에서 현재의 CORBA는 분산 객체의 수용을 위한 기본적인 구조만을 가지고 있다고 할 수 있다. 그러나 CORBA가 제공하는 확장 가능성은 앞으로의 기술을 어느 정도 수용할 수 있는 가능성은 있다.

또 다른 시도로는 기존의 네트워크 환경을 사용하는 사용자들에게 빠르고 쉽게 접근하고 이들을 끌어들이기 위하여 기존의 World Wide Web 환경을 통신 하부 구조로 사용하여 분산 객체 시스템을 제공하려는 것이다. 사용자들의 친숙성을 기반으로 쉽게 접근할 수 있는 장점이 있으나 WWW이 그러한 목적으로 설계된 것이 아니므로 효율성의 저하 등이 문제가 될 수 있다[2].

분산 객체 시스템의 기술적으로 남은 여러 가지 문제 중에서도 현재 관심을 모으는 시도의 하나는 비디오와 오디오 같은 실시간 연속 데이타의 처리를 어떻게 분산 객체의 개념 하에서 수용하는 가 하는 것이다. 현재까지 실시간 연속 데이타는 데이타의 생성 지점으로 부터 도달 지점까지 연결된 데이타 전송관을 기본 개념으로 다루어져 왔는데 이러한 것을 분산 객체의 개념 하에 어떻게 종합할 것인가가 중요한 문제로 보인다[1].

분산 환경은 개인용 휴대폰의 발전 등과 더불어, 이제 이동 계산 환경(mobile computing)으로 그 영역을 넓혀가고 있다. 이와 같은 추세에 따라 이동 계산 환경하에서 이를 적절히 지원할 수 있는 분산 객체 시스템이 향후의 또 다른 중요한 방향이 될 것이다.

분산 객체 시스템 상에서 분산 응용 프로그램을 개발할 수 있는 개발 환경으로서 시각적 개발 환경에 대한 연구 또한 현재 진행되고 있는 중요한 방향의 하나이다. 시각적 개발 환경은 보다 손쉬운 소프트웨어의 개발과 관리를 위한 방법으로 여러 분야에 응용 가능한 방법이나 시각화가 보다 용이할 것으로 보이는 분산 객체를 이용하여 분산 응용 프로그램 개발의 복잡성을 해결하려는 시도는 적절한 응용 분야로 보이며 현재 많은 관심이 모아지고 있는 부분이다[11]

6. 결 론

본고를 통하여 분산 객체 시스템에 대하여 다루어지고 있는 기본적인 개념들을 정리하고 현재 개발 중에 있는 복제 공유 객체 시스템에 대하여 소개 하였다.

앞서 언급한 바와 같이 분산 객체 시스템은 그 개념적인 간결성, 관리의 용이성, 확장성과 응용 프로그램 개발의 용이성 등의 장점으로 이미 앞으로의 컴퓨팅 환경의 초석으로 자리를 굳혀가고 있다. 분산 객체 시스템의 발전 과정 속에서 새로운 개념의 출현이나 기존 개념들의 얼마간의 수정들이 있겠으나 전체적인 방향은 분산 객체 시스템의 보편화로 진행할 것이다

분산 객체 시스템의 보편화에 대한 확신 속에서도 아직 그 상용화가 미진하고 최종적인 목표 시스템이 뚜렷이 나타나고 있지 않은 지금은 분산 객체 시스템에 대한 연구에 많은 노력을 기울일 때로 보여 진다

참고문헌

[1] *Control and Management of A/V Streams RFT*, OMG, Aug. 1996.

[2] A. Beitz, et al., Integrating WWW and Middleware, Middlewarespectra, Vol. 10, Rep. 1, pp 34-38, 1996

[3] Data Protocols for Multimedia Conferencing, Draft Recommendation T.120, ITU-T, 1996.

[4] 이동현, 박지은, 윤석환, 김평중, 신범주, "Supporting both Client-Server and Peer-to-Peer Models in a Framework of a Distributed Object Management System," Asian'96 발표, 1997 Lecture notes in Computer Science 게재 예정.

[5] 이동현, 외, "Design and Implementation of a Distributed Object Management System for Distributed Collaborative Applications," Proc. Of Intl. Conf. On Computer and Industrial Engineering, 경주, 한국, pp. 869-872, Oct. 1996.

[6] 이동현, 박치항, "RSOS. A Replicated Shared Object System for Groupware Applications," Proc. Of Parallel and Distributed Computing Systems, Dijon, France, pp. 477-480, Aug. 1996.

[7] 이동현, 박치항, "Management of Replicated Shared Object for Distributed Collaborative Applications," Proc. of Parallel and Distributed Processing Techniques and Applications, Sunnyvale, USA, pp. 199-202, Aug. 1996.

[8] Lewis, "Where is Client/Server Software Headed?," IEEE Computer, pp. 49-55, April. 1995

[9] Nelson, "Considerations in Choosing a Concurrent/Distributed Object-Oriented Programming Language," ACM SIGPLAN Notice, Vol. 29, No. 12, pp. 66-70, Dec. 1994

[10] Mowbray, R. Zahavi, *The Essential CORBA: System Integration Using Distributed Objects,* John Wiley & Sons, 1994

[11] T. Tou, Y. Eterovic, E. Wu, Prototying Synchronous Group Applications, Computer, pp. 48-56, May. 1994,

[12] Chin, S Chanson, "Distributed Object-Based Programming Systems, " ACM Computing Su-

rveys, Vol. 23, No. 1, pp. 91-124, Mar. 1991

[13] Ellis, S. Gibbs, G. Rein, "Groupware: Some Issues and Experiences," CACM, Vol. 34, No. 1, pp. 38-58, Jan. 1991

[14] Ellis, S. Gibbs, "Concurrency Control in Groupware Systems," pp. 399-407, SiGMOD 1989

이 동 현

1988년 서울대학교 자원공학과(학사)
1990년 과학원 전산학과(석사)
1990년-현재 한국전자통신연구소 시각언어 연구실 선임연구원
관심분야 : Distributed Object-Oriented System, Object-Oriented Visual Programming, Event-based Programming, Machine Learning

황 승 구

1979년 서울대 공과대학 전기공학(학사)
1981년 서울대 공과대학 전기공학(석사)
1986년 미국 University of Florida 전기공학(박사)
현재 한국전자통신연구소 멀티미디어연구부 부장/책임연구원
관심분야 Networking Computing, Intelligent Computing, Visual Computing, Mobile Computing