

클라이언트/서버기법을 이용한 분산시스템 구축(I)

Building a Distributed System using Client/Server model

전성택

한국전자통신연구소 소프트웨어 공학 연구실

Jun, Sung-taeg./ETRI, Software Engineering Section

▶ 연재순서

1. 시스템 구조 선정

2. 클라이언트/서버 도구의 선정①

3. 클라이언트/서버도구의 선정②

1. 시스템 구조 선정

클라이언트/서버, 다운사이징, 중간모
(Middle-ware) 등의 용어에는 친숙하면
서도 분산처리 환경 또는 분산시스템하면 웬
지 모르게 낯설은 느낌을 갖는 분들을 전산관
련자들 중에도 종종 볼수 있다.

실제는 클라이언트/서버 기법이 분산시스템을
구현하는 한가지 방안으로서 궁극적으로는 분
산시스템을 구축하는데에 사용이 되는 것이
다.

본고에서는 한국전자통신연구소 소프트웨어
연구부에서 수행하는 분산시스템 소프트웨어
기술 개발 사업(일명: 한우리, 사업책임자: 김
명준 책임연구원)을 중심으로 효율적인 분산
시스템 구축을 위하여 알아야 할 사항들을 기
술한다.

한우리는 국산 주전산기인 타이컴을 분산처
리환경에 적합한 시스템으로 사용하기 위하여
분산시스템 소프트웨어를 분산트랜잭션처리
(한우리/T), 클라이언트/서버 도구(한우리/C)
및 분산관리(한우리/M)의 3개 분야로 나누어
수행한다.

이 중 데이터베이스와 직접적인 관련이 있
는 한우리/T와 한우리/C의 기능을 중심으로
하여 설명하여 나간다.

클라이언트/서버 시스템을 구축하는데에 있
어서 가장 먼저 고려해야할 사항은 2계층(2-
tier) 시스템을 구축할 것인가 3계층(3-tier) 시

시스템으로 할 것인가 하는 점을 결정하여야 한다. 3계층 시스템은 ① 화면 표시와 전반처리 기능을 담당하는 클라이언트 ② 비즈니스 로직을 처리하는 응용서버 및 ③ 관계형 데이터베이스, 화일 시스템, 큐등의 자원관리기의 3계층으로 이루어져있다. 2계층 시스템은 이중 비즈니스 로직을 수행하는 응용 서버의 기능을 클라이언트 또는 서버에서 포함하는 시스템을 말한다.

2계층 구조의 대표적인 예로는 마이크로소프트사에서 규정한 ODBC(Open Data-Base Connectivity)를 들수 있다. ODBC는 56개의 API로 규정된 표준으로 이 표준에 따라 각 데이터베이스 회사가 ODBC 드라이버를 제공하여 클라이언트와 다양한 데이터베이스와의 연결을 지원하여 준다.

3계층 시스템중 중간에 자리잡은 응용서버를 자칭하여 중간모라 하는데 이를 공식적으로 정의하면 통신망 프로토콜과 응용프로그램의 중간에 존재하는 다양한 네트워크 프로토콜과 다양한 컴퓨터 구조 및 다양한 프로그래밍언어를 지원하는 소프트웨어 계층이라고 정의할 수 있다. 중간모는 응용프로그래밍 인터페이스(API)형태로 제공되어 사용이 용이하도록 되어 있다. 대표적인 중간모로는 분산 트랜잭션 처리기(DTP), 원격프로시저호출(RPC), OMG (Object Management Group)의 C-OBRA(Common Object Re-quest Broker Architecture), OSF(Open Software Foundation)의 DCE(Distributed Computing Environment)등이 있다.

클라이언트 또는 서버에 포함이 가능한 중간모를 따로 분리하여 독립적인 계층으로 구

성하는 이유는 효율성, 확장성, 안정성을 들수 있다. 이를 중간모 중에서 데이터베이스와 가장 관련이 적은 분산트랜잭션 처리기를 통하여 알아보자. 분산 트랜잭션 처리기를 사용하면 기존의 데이터베이스관리기에서 제공하는 OLTP(On-Line Transaction Processing) 기능에 비하여 어떤 장점이 있는지를 알아보자. 기존의 OLTP는 특정 데이터베이스에 연결되어 특정한 서버에 있는 데이터를 접근하는데 사용되는데에 비하여 분산트랜잭션 처리기는 여러 개의 서버를 효율적으로 접근하여 부하를 분산하는 기능을 가지고 있다. 새로운 서버를 추가하는 것이 용이하도록 되어 있어 확장이 간편하며 데이터를 여러개의 서버에 분산하는 것이 가능하여 지역적인 고장을 감내할 수 있는 안정성이 있다.

특히, 특정 데이터베이스 제품에 종속되지 않고 여러가지 이기종의 데이터베이스를 동시에 관리할 수 있으며 새로운 제품 또는 브랜드의 데이터베이스를 추가할 수 있는 장점을 가지고 있다.

이러한 장점은 종래의 컴퓨팅 시스템을 확장할 때에 기존의 시스템에 있는 응용 프로그램들 때문에 더욱 뛰어난 시스템을 알면서도 특정한 시스템으로만 국한해야하던 종래의 컴퓨팅 방식을 크게 개선한 것이다.

그러면, 이러한 DTP만으로는 다양한 기종의 컴퓨터에 분산되어 있는 데이터, 화일 등의 자원을 최대한으로 활용하는 것이 불가능하다. 다양한 회사의 다른 기종들의 컴퓨터를 효율적으로 접근 조작하기 위해서는 개방형분산 시스템 기반을 필요로 한다.

이 개방형 분산시스템 기반에 관한 잘못된

인식을 갖고 있는 사람들을 종종 볼 수 있는데 작년에 어느 전산 관계자와의 대화를 소개하면 그 분은 분야의 특수성때문에 보안이 특히 강조되는 분야에서 근무하는데 개방형 시스템을 사용하면은 보안의 기능을 상실하게되는 것이 아니냐는 질문을 받은 적이 있었다. 이는 개방형 분산시스템을 잘못 이해한 것으로 개방형분산시스템의 의도는 같은 회사 제품의 컴퓨터끼리는 컴퓨팅 자원의 효율적인 분배가 쉬운 반면 다양한 회사의 컴퓨터를 하나의 네트워크로 구성하여 컴퓨팅 자원을 공유하는 것은 어렵다.

이렇게 여러 회사 제품의 컴퓨터를 하나의 네트워크에 연결하여 시스템 구성을 하기 쉽도록하기 위해서 나타난 것이 개방형분산시스템의 목적이며 컴퓨터내의 정보를 개방한다는 의미는 아니다.

이러한 컴퓨터의 보안문제는 개방형분산시스템이거나 기존의 기업 독자적인 구조를 가진 시스템이나의 문제가 아니라 네트워크에 연결되어 다른 사용자가 컴퓨터에 원격로그인을 할 수 있느냐 하는데에 발생하는 문제점으로 개방형 분산시스템 구조가 다른 시스템에 비하여 보안이 허술한 것이라는 생각은 잘못된 발상이다.

실제로 OSF의 DCE같은 시스템은 기존의 네트워크로 연결된 컴퓨터에 비하여 강화된 보안서비스를 제공하고 있는데 이에 사용되는 보안기법은 미국 MIT대학에서 개발한 Kerberos 시스템을 기반으로 하고 있다.

이 Kerberos 시스템에서는 사용자(Authentication)내기 위하여 시간도장(time stamp)를 요구하는데 이를 위하여 OSF의

DCE는 분산시간서비스를 포함하고 있다.

개방형분산시스템 기반 중에서 가장 기능이 다양하며 실제 사용되고 있는 시스템은 OSF의 DCE를 꼽는다. OSF의 DCE는 앞에서 언급한 ① 보안 서비스와 ② 분산시간 서비스 외에도 ③ 기존의 소켓(Socket)을 이용한 프로그래밍 보다 간편한 원격 프로시저 호출(RPC)서비스, ④ 효율적인 서버의 공유를 가능케 하는 스레드(Threads), ⑤ 시스템 이름만으로 네트워크에서 찾을 수 있도록 하여 주는 디렉토리 서비스(CDS) 및 ⑥ 카네기멜론 대학에서 개발한 앤드류 화일 시스템(AFS)를 기본으로하는 분산화일 서비스의 여섯 가지 서비스를 제공하여 분산시스템에서의 컴퓨팅 지원을 효율적으로 공유할 수 있도록 한다. 이러한 개방형 분산시스템 기반의 기능은 실 사용자가 직접적으로 사용하기는 곤란하며 상위에 존재하는 프로그램들에 API형태로 제공된다.

일 예로, 분산 트랜잭션처리기 중에서 가장 진보된 기능을 가지고 있는 미국의 Transarc社 제품인 Encina같은 제품은 OSF의 DCE를 하부 구조로 요구하고 있다.

OSF의 DCE에 비하여 실용화가 약간 뒤쳐져 있는 OMG의 COBRA는 차세대 컴퓨팅 기법인 객체 지향형 기능을 직접적으로 지원하므로 규격면에서는 OSF의 DCE를 능가하나 실제적으로 사용하기 위한 각종 기능의 구현에는 향후 수년간의 노력이 필요하다. COBRA는 시스템내의 객체들간의 통신을 객체가 구현된 기법 또는 객체가 동작하고 있는 특정 하드웨어에 관계없이 ORB(Object Request Broker)가 보장하는 시스템으로 객

체의 생존주기(Life cycle)관리를 주요 기능으로 하여 객체의 생성(Object Factory), 객체의 제어, 옮겨진 객체의 추적 및 유사 객체의 관리(Class management)를 담당하는 객체 서비스(Object Services)요소와 특정 기능(Common Facilities)요소라 불리는 일반 컴퓨팅 기능을 특정 시스템에 맞도록 재단한 기능을 제공한다. 이러한 객체서비스의 예로 프린트 기능, 전자우편 기능, 데이터베이스 기능 등을 들 수 있다.

클라이언트로 가장 많이 사용되는 IBM 호환 PC에서는 마이크로소프트사에서 제공하는 OLE(Object Linking and Embedding)을 들 수 있는데 이는 PC에서만 동작하여 아직까지 네트워크에서의 사용이 보장되지 않아서 중간모의 기능을 할수 없으나 COBRA와의 호환성을 추구할 것으로 예상되어 머지않은 장래에 중간모로서의 기능을 발휘할 수 있으리라 본다.

이러한 상황에서 어떠한 형태를 기반으로 시스템을 구축하느냐 하는 것은 구축하고자 하는 시스템의 용량과 확장성들을 고려하여야 할 것이다.

2계층 구조로 하느냐 3계층 구조로 하느냐는 문제는 분산트랜잭션 처리기가 필요한 것이냐 아니냐의 문제로 귀착된다.

그러면 이러한 분산트랜잭션 처리기는 모든 클라이언트/서버 시스템에 필요한 것인가 하는 의문을 갖게한다.

이러한 질문은 역시 구축하는 시스템의 크기에 따라 결정될 사항이다.

대형의 시스템을 구축하는데에 있어서는 꼭 필요하지만 소규모 시스템을 구축하는데에는

오히려 비용만을 증가 시키는 역할을 할 수도 있다. 컴퓨터 컨설팅 전문업체인 가트너 그룹에 의하면 금년 중으로 약 절반 가량의 데이터베이스는 OLTP 또는 DTP기능을 갖출 것이라고 전망하고 있다.

그러면, 어떤 시스템부터 대용량의 시스템이라고 할 수 있는가에 대해서 여러가지 의견들이 있을 수 있는데 약 50개 이상의 클라이언트와 2개 이상의 서버를 필요로 하는 시스템을 구축할때 부터는 분산트랜잭션 처리기를 고려하여야 한다고 한다.

또한 분산트랜잭션 처리기를 사용하기로 결정을 한 뒤에는 개방형분산시스템 구조를 따를 것인가 하는 질문에 봉착하게 되는데 필자의 의견으로는 추후 확장성을 고려한 개방형 분산시스템 구조를 채택하는 것이 바람직하다고 본다.

이러한 기본적인 구조를 선정한 뒤에 실제 응용프로그램을 생성하는데에 사용되는 클라이언트/서버 도구의 선정이 필요하다. 필자가 어느 책에서 요즘 가장 오용되고 있는 말이 클라이언트/서버 도구라는 글을 보았는데 이 글이 실감이 갈 정도로 다양한 클라이언트/서버 도구가 시중에 나와 있으며 새로이 나타나고 있는 상황이다.

그러면, 이 중 어떤 제품을 사용하여야 하는지에 관하여 다음 달에는 한우리/C의 기능을 중심으로 하여 클라이언트/서버 도구의 선정시에 필수적으로 고려할 사항들을 논의하겠다. **DC**

(다음호에는 클라이언트/서버 도구의 선정①이 게재된다.)