□ 기술해설 □

DCE를 이용한 분산 컴퓨팅

데이콤종합연구소 성재모*

목 차

1. 서 론

일반적으로 분산 컴퓨팅이란 분산 컴퓨팅 시스템(distributed computing system)에 의해 제공되는 서비스로, 여러 대의 컴퓨터를 네트웍으로 연결하여 컴퓨터 상호간에 업무적 기능들이 네트웍을 통해서 성취되는 형태를 의미한다. 기술적 관점에서는 컴퓨터 상호간에 주 기억장치를 공유하지 않는 것으로, 정보 교환은 광역 변수(global variable)를 통해 전달되지 않고 네트웍을 통한 메시지 교환으로 이루어지는 것을 의미한다[1]. 사용자 관점에서의 분산 컴퓨팅은 한 그룹의 컴퓨터를 마치 한 기계처럼 보이도록 구성하는 것을 의미하며, 사용자에게 컴퓨터들 사이의 관계가 투명(transparent)하고 무관(irrelevant)하게 보이도록 만드는 것을 말한다[2].

이러한 분산 컴퓨팅은 컴퓨터 시스템의 급진적 발전으로 인한 소형 컴퓨터의 성능 향상과 가격 하락으로 보급 수량이 증가하고, 통신 기술 발달에 힘입어 LAN/WAN 등을 통한 컴퓨터 이용이 증가됨에 따라 서로 다른 형태, 크기, 모습으로 나타나고 있다. 외국의 경우 금융계, 정부 기관, 교육 기관, 연구 기관, 생산 업체와 의료계 등 산업 기반 전체에 걸쳐 사용되고 있는 실정이다.

이러한 추세에 맞추어 본 연구소에서는 당사에서 제공하고 있는 PC 통신 서비스의 일부를 대상으로 분산 응용 시스템 프로토타입을 구현하여 향후 당사의 PC 통신 서비스 시스템의 확장과 변화에 필요한 방안을 제시하였다. 분산 응용 시스템 구현에 있어서 클라이언트-서버 모형을 적용하였고, OSF(Open Software Foundation) DCE(Distributed Computing Environment)를 사이모로 사용하였다.

본 고에서는 먼저 DCE의 구조와 기능에 대해 알아보고, 프로토타입으로 구현한 PC 통신 서비스 중 회원 정보 서비스에 대한 구현 사례와 성능 시험 결과를 내용으로 다루고자 한다.

2. DCE

분산 응용 시스템을 개발하기 위해서 개발자는 프로그램과 자료 등의 자원 분산, 자원의 위치 정보 관리, 이기종간의 통신 방식 조정, 통신상의 보안 문제 등 여러 가지 문제를 고려해야 하는 어려움이 있다. DCE는 이러한 문제를 쉽게 해결할 수 있도록 분산 컴퓨팅 환경을 제공해 주는 사이모로서 이기종으로 구성된 컴퓨팅 환경에서 분산 응용 시스템을 개발, 이용,

*비 회 원

관리, 유지하는데 필요한 여러 가지 도구와 서비스를 제공해 준다.

2.1 DCE 구조 및 기능

그림 1은 DCE 구조와 그 기술적 구성 요소들을 보여 주고 있다. 또한 응용 프로그램, 기존 시스템의 자원, 향후 기술 동과의 관계도 나타내고 있다.

DCE 스레드(thread)는 POSIX 1003.4a/D4 pthreads 규정에 따라 구현된 것으로 경량 프로세스(lightweight process)라고 불리운다. 이 기능은 한 프로세스내에서 여러 독립적인 스레드를 만들어 관리하고 동기화 제어가 가능하므로 여러 수행 흐름을 가지고 동시에 여러 일을 수행할 수 있다. 개념적으로는 DCE의 하위 계층인 운영 체제의 일부로 볼 수 있다. 이미 기존 운영 체제가 스레드를 제공한다면, DCE 스레드를 쓰지 않고 기존 스레드를 사용할 수도 있다.

DCE RPC(Remote Procedure Call)는 클라이언트-서버 모델을 구현하기에 필요한 요소 중 하나로, 두 개의 다른 프로세스에서 한 쪽에는 호출하는 코드가 다른 쪽에는 호출되어지는 코드가 있는 프로세스 호출 형태를 말한다. RPC 관점에서 프로시져 호출을 하는 프로그램을 클라이언트라하고 이 클라이언트 호출에 의해 실행되는 프로시져를 포함한 프로그램을 서버라 한다. 이 때 처리 과정은 호출자에게 은폐되며, 개발자는 네트웍 전반에 걸쳐 자원을 사용할 수 있다.

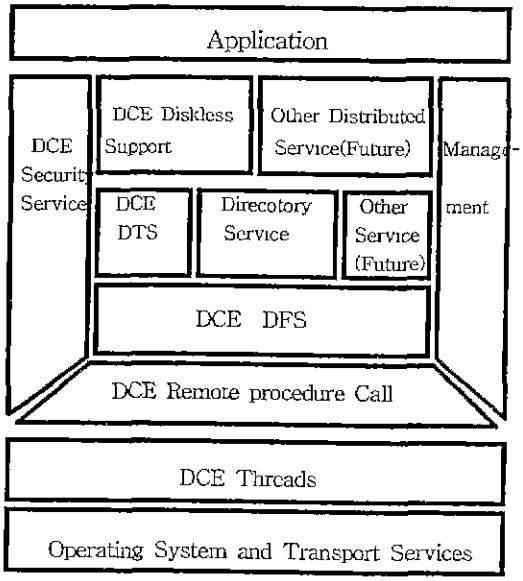

그림 1 DCE 구조

디렉토리 서비스(Directory Service)는 분산 시스템의 자원, 즉 사용자, 컴퓨터, 응용 프로그램, 자료등에 관한 정보를 보관하고 관리해 주는 서비스이다. 클라이언트가 RPC를 사용할 때 서버의 위치를 저장하였다가 알려준다.

DCE DTS(Distributed Time Service)는 Digital사의 DCEdts에 기반을 둔 것으로 분산 컴퓨팅 환경에서 컴퓨터들 간에 동기화된 시간을 보장해 준다. 이것은 국제 표준인 CUT(Coordinated Universal Time)를 이용하며, DTS 서버는 외부의 신뢰성 있는 시간 제공자로부터 입력을 얻거나 보통 3개 이상의 DTS 서버들 간에 동기화된 시간을 DTS 클라이언트들에게 제공해 준다.

DCE 보안 서비스(Security Service)는 HP-Apollo에 의해 제공된 것으로 Kerberos와 PasswdEtc 등에 기본을 두고 있다. 이것은 안전한 통신 방법과 통제된 분산 자원의 사용을 담당하기 위해 DCE 사용자, 컴퓨터, 서버 프로세스 등을 등록 관리하는 등록 서비스(Registry Service), DCE 사용자의 동일성을 결정해주는 인증 서비스(Authentication Service)와 특정 객체의 사용 권한 여부를 결정 짓는 권한 서비스(Authorization Service)를 제공해 준다.

DCE DFS(Distributed File Service)는 물리적인 위치에 관계없이 네트웍 내의 파일 서버에 저장되어 있는 모든 파일들을 지역 파일 시스템처럼 사용할 수 있게 해주며, ACL(Access Control List)에 의해 모든 파일들과 디렉토리들의 사용 권한이 통제된다.

DCE DSS(Diskless Support Service)는 디스크가 없는 시스템들을 위하여 네트웍을 통해 다른 시스템에 있는 운영체제를 사용할 수 있게 해준다.

그림 1의 관리(Management) 부분은 실제로는 단일 구성 요소는 아니고 다른 구성 요소들과 연결되어 있다. 각 DCE 서비스들은 네트

웍을 통하여 관리할 수 있도록 관리적인 부분을 포함하고 있다. 예를 들면 사용자 정보는 보안 서비스에 등록하고, 응용 서버의 네트웍 내에서의 위치는 디렉토리 서비스에 등록한다.

DCE의 장점 중 하나는 서비스들 간의 결합성에 있다. 구성 요소들 자체가 잘 정의된 인터페이스를 가진 모듈이면서 잘 통합되어 있다. DCE의 여러 구성 요소들은 서로 다른 구성 요소들이 제공하는 서비스를 사용하고 있다. 예를 들면, RPC는 디렉토리 서비스를 사용하여 서버와 그 특성을 찾는다. 또한 RPC는 DCE 스레드를 사용하여 다수의 RPC를 동시에 실행할 수 있다. DFS는 스레드, RPC, 디렉토리 서비스, 분산 시간 서비스, 보안 서비스 등을 이용하고 있다.

2.2 DCE 셀의 구성

DCE 서비스를 이용하여 분산 컴퓨팅 환경을 구성하는 컴퓨터, 사용자, 그리고 다른 자원과 일단의 서비스의 모임을 셀(Cell)이라고 한다. 모든 DCE 사용자와 기계는 하나의 지역 셀(local cell or home cell)에 포함되어 있어야 하며, 자신이 포함되어 있지 않는 다른 셀을 외부 셀(foreign cell)이라고 부른다.

셀의 예를 들면 대학이나 거대한 회사의 과

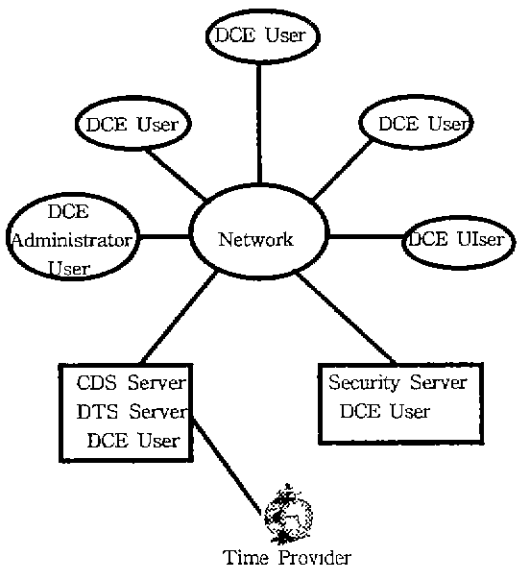

그림 2 DCE 셀

단위나 프로젝트 그룹이 하나의 셀이 될 수 있고, 작거나 중간 크기의 회사 전체가 한 셀로 구성될 수도 있다.

셀의 특성으로는 한 대의 컴퓨터에서 수천 대의 컴퓨터의 모임이 하나의 셀이 될 수도 있고, 단일 기종 뿐만 아니라 이기종 컴퓨터들도 포함할 수 있다. 또, 하나의 LAN으로 구성되거나 여러 LAN들의 모임 또는 WAN으로도 하나의 셀을 구성할 수 있다.

그림 2는 간단한 DCE 셀의 구성 예인데, 모든 셀은 적어도 보안 관리 서버(Security Server), 셀 디렉토리 서버(Cell Directory Server), 분산 시간 관리 서버(Distributed Time Server)를 하나씩은 가지고 있어야 한다.

3. 분산 응용 시스템 프로토타입

3.1 응용 시스템 개요

본 시스템은 현재 당사에서 제공하고 있는 PC 통신 서비스의 일부를 DCE를 플랫폼으로 하여 분산 컴퓨팅 시스템으로 구현한 것으로 PC 통신 서비스 시스템 모형 (PC Communication Service System; PCSS)이라 칭한다.

PC 통신 서비스란 PC에서 통신망을 통해 PC 통신 서비스 시스템에 접속하여 공개 자료실, 게시판, 전자 메일, 상용 DB검색 등의 서비스를 온라인으로 받는 것을 말한다. PC 통신 서비스 시스템은 접속한 가입자의 로그인 ID와 비밀 번호를 입력받아 사용자를 인증한 후, UNIX의 쉘과 유사한 역할을 하는 프로그램을 동작시켜 사용자의 명령에 따라 여러 가지 서비스를 제공한다.

90년대 이후 PC통신 인구는 급격히 증가하고 있으며, 최근에는 멀티미디어형 서비스와 고속 모뎀의 출현으로 가입자 증가 추세는 더욱 가속화 되고 있는 실정이다. 이로 인해 가입자 관리 시스템의 부하는 갈수록 증가하고, 멀티미디어 자료 전송 등으로 PC 통신 시스템 전반에 걸쳐 문제가 발생하고 있어, 분산 컴퓨팅의 필요를 느끼고 있는 실정이다.

분산 컴퓨팅으로 시스템을 구현할 경우 사용자 과다로 인한 PCSS의 성능 저하를 회복시키고 사용자가 원하는 자료를 위치에 관계없이

자유자재로 접근할 수 있는 장점이 있다.

호스트쪽 사용자 인터페이스는 PCSS에 접속한 가입자를 인증하고 메뉴를 제공해 주며 사용자가 원하는 서비스 시스템을 구동시킨다. 이를 지원하기 위해 로그인 서비스, 메뉴 서비스 등 부 서비스들이 있다. 현재 로그인 서비스는 한 대의 호스트에 의해 제공되어지고 있는데 특정 시간대에 가입자의 접속이 과다하게 발생하여 시스템 성능이 현저하게 저하되고, 이 시스템에 장애가 발생하면 잠시 동안 새로운 가입자 접속은 중단되는 등 문제점들이 산재해 있다.

본 고에서는 본 연구소에서 구현한 PCSS 중에서 로그인 시스템을 중점적으로 다루고자 한다.

3.2 로그인 시스템 구현

로그인 시스템은 클라이언트－서버 구조를 기본으로 하고, 서비스 분산을 위하여 DCE를 사이모로 사용하였다. 그리고 중단없는 안정된 서비스를 제공하기 위해 n(=2) 개로 분산했고, 각각의 사본을 만들어 원본과 서로 다른 호스트에 두도록 구성하여 한 호스트가 장애시 다른 호스에서 사본을 이용해 바로 서비스 되도록 구성하였다.

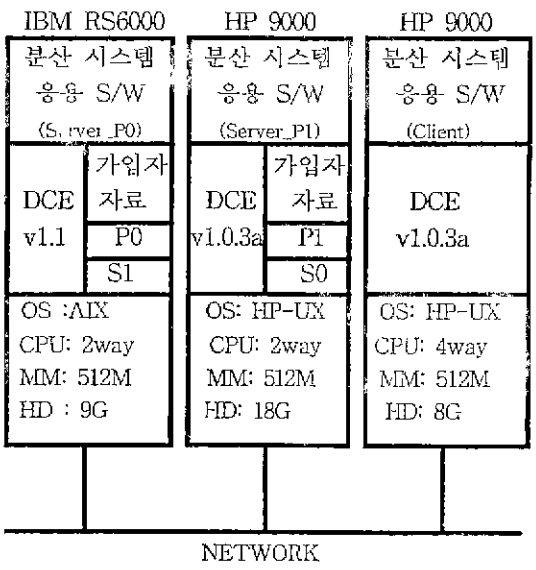

그림 3 로그인 시스템 환경

향후 100만 이상의 가입자 수용을 위해 가입자 데이터는 실제 정보량과 유사하도록 C-ISAM을 이용, 가공하여 두 대의 시스템에 50만씩 분산시켰으며 네트웍은 FDDI와 Ethernet을 사용하였다.

그림 3은 로그인 시스템의 하드웨어 및 소프트웨어 환경을 나타내고 있다.

기존 응용 프로그램은 프로그램을 호출하는 부분과 수행하는 부분이 같은 공간에 있는 것이 보통이지만, 분산 응용 프로그램은 서로 다른 기계의 주소 공간에 있을 수 있다. 즉 그림 3에서 분산 응용 S/W는 클라이언트－서버 모형을 취하여 클라이언트 프로그램과 서버 프로그램이 서로 다른 기계에 존재하는 것을 볼 수 있다.

클라이언트－서버 프로그램을 구현하는 가장 간단한 방법은 RPC(Remote Procedure Call)를 사용하는 것이다. 네트웍을 사용하여 소프트웨어를 구현할 때 따르는 많은 어려움은 RPC를 사용함으로써 해결될 수 있다. 즉 개발자는 소프트웨어를 구현할 때 네트웍 통신에 관한 부분은 깊이 알 필요가 없어 개발자의 부담을 줄일 수 있다.

분산 응용 프로그램 개발은 그림 4와 같이 세 단계로 나눌 수 있다. 클라이언트와 서버의 RPC에 관한 내용을 기술한 것을 인터페이스(Interface)라고 한다. 인터페이스만 정의되면 클라이언트와 서버는 서로 다른 기계에서 개발할 수도 있다.

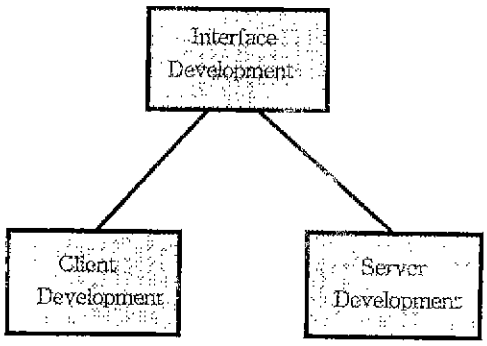

그림 4 응용 소프트웨어 개발

3.2.1 인터페이스 정의

DCE로 클라이언트－서버 프로그램을 개발

하려면 먼저 클라이언트와 서버 간의 인터페이스를 정의해야 한다. 인터페이스는 DCE에서 제공하는 인터페이스 정의 언어(Interface Definition Language)로 작성하며, 모든 원격 프로시저에 대해 호출부와 구현부에서 함께 준수할 규칙을 정의한다. 인터페이스 정의는 머리말과 본문으로 나뉘는데, 머리말에는 인터페이스 머리말 속성과 인터페이스 이름이 포함되고 본문에는 포함문, 상수 정의, 자료형 정의, 원격 프로시저 선언이 포함된다.

로그인 시스템은 클라이언트인 clogin-client 와 서버인 clogin-server 사이에 인터페이스 정의 파일 clogin.idl을 갖는다. 아래 [예 1]은 clogin.idl의 일부이다.

[예 1] clogin.idl

```
[
uuid(0088e46a - eae8 - 1deb - bba9 - 08005a0d8842),
version(1.2)
] interface clogin
{
  error_status_t clogin_GetUserInf(
    [in] handle_t clogin_bh,
    [in] long length,
    [in,string] char filename[41],
    [in,out] char user_rec[265]
  );
}
```

이것을 IDL 컴파일하여 클라이언트쪽 스텁 코드(clogin_cstub.c)와 서버쪽 스텁 코드(clogin _sstub.c)를 얻는다. 스텁 코드는 클라이언트-서버가 주고 받는 자료를 IDL 자료 형식으로 변환시켜 이기종간 네트웍 통신시에도 일관된 자료의 형식을 가지게 해주고, RPC 수행 라이브러리와의 통신을 담당한다. 또한 원격 프로시저의 리턴 값과 파라메타가 가질 수 있는 속성 중에는 통신 상의 장애 발생과 서버쪽 스텁 또는 원격 프로시저에서 발생한 예외 상황을 알려주는 것이 있다. [예 1]에서도 볼 수 있는데 error_status_t의 값으로 리턴되어 오는 것이 그것이다. 정상적인 수행시 값은 rpc_s_ok 이다. 따라서 개발자는 통신과 관련된 복잡한 과정은 RPC에 맡길 수 있다.

3.2.2 서버 구현

모든 DCE 응용 서버 프로그램은 서버 초기화 부분과 원격 프로시저 구현 부분으로 구분된다. 서버 초기화 과정은 아래와 같으며 [예 2]는 clogin_server의 초기화 부분의 일부이다.

① 인터페이스를 RPC 수행 라이브러리에 등록한다.
② 서버 바인딩 정보를 생성한다.
③ 서버의 위치를 클라이언트가 찾을 수 있도록 CDS(Cell Directory Server)의 DB에 등록한다.
④ 서버가 있는 시스템의 endpoint map에 endpoint를 등록한다.
⑤ 원격 프로시저 호출을 기다린다.

[예 2] clogin_server.c

```
① rpc_server_register_if(
        clogin_v1_2_s_ifspec, . . .
  );
② rpc_server_inq_bindings(
        &bind_vector, . . .
  );
③ rpc_ns_binding_export(
        rpc_c_ns_syntax_default,
        (unsigned_char_t *)entryname,
        clogin_v1_2_s_ifspec,
        bind_vector, . . .
  );
rpc_ns_group_mbr_add(
     (unsigned_char_t *)groupname, . . .
     (unsigned_char_t *)entryname, . . .
  );
④ rpc_ep_register(
        clogin_v1_2_s_ifspec,
        bind_vector, . . .
  );
⑤ rpc_server_listen(
        max_threads, /* process one re-
        mote
        procedure call at a time */ &sta-
        tus
  );
```

초기화 과정중 서버의 바인딩 정보를 생성하고 등록하는 과정을 볼 수 있는데, 서버, 자료 등 모든 자원이 여러 기계에 분산되어 있는 분산 시스템에 클라이언트 프로세스는 원하는 자원에 접근하기 위해서 그 자원의 위치를 알아야 한다. 이 과정을 바인딩이라 하고 그 자원의 위치 정보를 바인딩 정보라 한다. 바인딩 정보는 통신 프로토콜, 네트웍 주소, endpoint를 포함하고, DCE에서 객체 구분자로 이용하는 객체 UUID(Universal Unique Identifier)를 포함하기도 한다.

DCE CDS는 이러한 바인딩을 쉽게 할 수 있도록 방법을 제공해 준다. 서버를 비롯한 모든 자원은 바인딩 정보를 자신의 이름(entry name)과 함께 CDS의 DB에 한 엔트리로 저장해 놓고, 클라이언트 프로세스는 자원의 CDS 엔트리 이름을 키로 서버의 바인딩 정보를 검색한다. 개발자의 입장에서 볼 때, 접근하고자 하는 자원의 CDS 엔트리 이름만으로 그 자원을 담당하고 있는 서버와의 바인딩이 가능하기 때문에 서버의 위치와 무관하게 투명한 시스템을 개발할 수 있다.

3.2.3 클라이언트 구현

본 장에서는 로그인 클라이언트의 업무적인 기능 보다는 통신하고자 하는 서버 프로세스를 찾는 과정을 주로 설명하고자 한다. 클라이언트는 서버의 바인딩 정보가 저장된 바인딩 핸들(binding handle)을 얻어 원격 프로시져 호출을 한다. [예 3]은 clogin-client.c의 일부분으로 바인딩 핸들을 얻는 과정이 나타나 있다.

```
[예 3]  clogin—client.c
①st = group—binding(groupname,
        clogin — v1 — 2 — c — ifspec,NULL,
        clogin—bh); . . .
②st  =  clogin—GetUserInf(clogin—bh,
        length,
filename, user—rec); . . .
```

[예 3]의 ①에서 서버의 그룹 이름과 인터페이스를 가지고 바인딩 핸들 clogin-bh를 얻어 ②에서와 같이 원격 프로시져 clogin-GetUser Inf의 첫 매개변수로 서버 프로세서를 찾아 업무를 수행한다. clogin-GetUserInf는 [예 1]의 clogin.idl에서 정의된 원격 프로시져와 동일함을 알 수 있다.

CDS를 이용한 그룹 바인딩은 특정 서비스를 제공하는 서버들의 CDS 엔트리 이름들을 동일한 CDS 그룹 엔트리에 등록해 놓음으로써 가능하다. 그림 5는 로그인 서버가 등록한 CDS 이름 공간을 나타낸다. 로그인 서버의 모든 바인딩 정보는 ③과 같이 서버 엔트리에 등록되고 같은 호스트에서 수행되는 서버 엔트리들은 ① 또는 ②와 같이 한 그룹 엔트리의 멤버로 등록된다. 로그인 클라이언트는 사용자의 로그인 ID에 따라 해당 그룹 엔트리를 검색하여, 그룹 엔트리에 속한 멤버 중에서 임의의 서버 엔트리의 바인딩 정보를 얻는다.

그룹 엔트리에 속한 서버의 임의의 선택은 부하의 균등 분배가 고려되지 않는다. 시험해 본 바에 의하면 90% 이상의 선택이 첫 번째 서버에 몰린다. 따라서 로그인 서버의 부하를 균등히 분배하기 위해 (프로세스 ID % 맴버 서버 수)가 서버 엔트리 이름의 끝자리와 일치하는 것을 선택하도록 구현하였다.

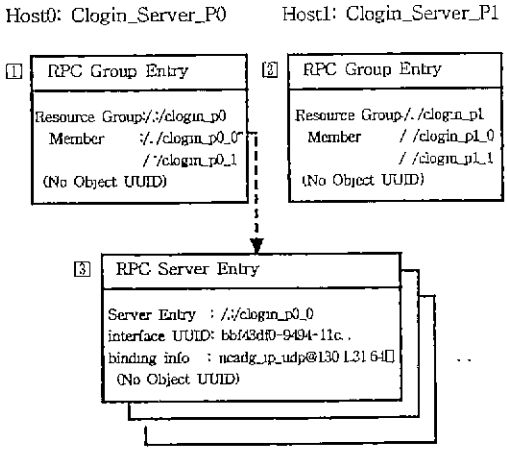

그림 5 서버가 등록한 CDS 엔트리

3.2.4 DCE 스레드 활용

DCE를 이용하여 구축한 응용 서버는 기본적으로 다중 스레드(multi-threads) 프로세스이다. 개발자가 스레드를 생성하지 않아도 서

버의 수행 라이브러리가 몇 개의 원격 프로시져 호출을 받아들이는 스레드를 생성하는데 이를 호출 스레드라 부르고, 응용 프로그램에서 개발자가 생성하는 스레드를 응용 스레드라 부른다. [예 2]의 ⑤에서 max-threads의 값이 호출 스레드의 갯수를 결정한다. 로그인 시스템에서도 max-threads(=10)으로 정하여 서버의 성능을 높였다.

4. 성능 시험

시험 환경은 그림 6과 같이 HP W/S에 DCE 셀 디렉토리 서버(cdsd), 보안 서버(secd) 와 분산 시간 서버(dtsd)를 두고 각 기계에는 DCE 클라이언트 모듈과 원격 프로시져 호출 서버(rpcd)를 두는 시험 셀을 구성하였다.

로그인 시스템의 사양은 그림 3에 나타나 있다. 로그인 시스템은 사용자가 PC 통신 서비스 접속시 정당한 사용자인가를 확인하기 위해 로그인 ID와 비밀번호를 입력 받아 사용자 인증을 하는 서비스 시스템이다. 성능 측정을 위하여 로그인 ID를 RPC 로 전송한 뒤, 돌아온 정보 값으로 비밀번호 확인 과정까지의 응답 시간을 측정했다.

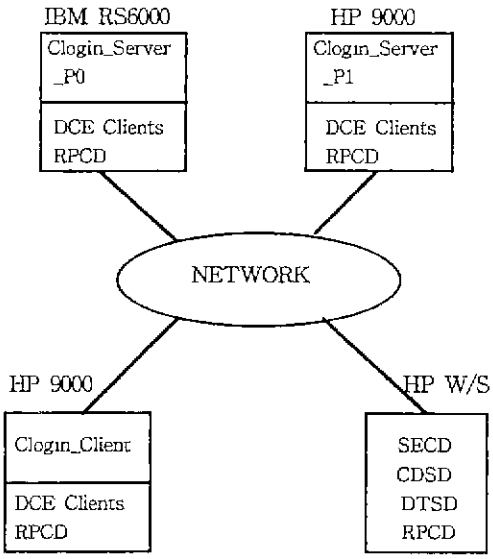

그림 6 시험 셀

서버는 두 대의 기계(IBM RS6000과 HP 9000)에 각 50만의 가입자 파일을 가지고 서비스를 하고, 클라이언트 기계(HP 9000)에서는 클라이언트(clogin-client)가 100, 200, ..., 600개 단위로 수분간 동시에 접속 서비스를 수행하도록 하였다. 시험 결과는 표 1의 응답 시간에 나타나 있다.

표 1 응답 시간

클라이언트 수 (개)	100	200	300	400	500	600
평균시간(sec	0.015	0.019	0.026	0.038	0.049	0.072
최소 (sec)	0.001	0.001	0.001	0.004	0.004	0.007
최대 (sec)	0.043	0.308	0.202	0.404	0.420	0.571

5. 결 론

분산 컴퓨팅 플랫폼을 기반으로 PC 통신 서비스를 구현한 결과 개발 생산성과 확장성 및 유지보수 면에서 좋은 것으로 나왔다. 또한, 현 운영 기종과 다른 여러 기종에서 시험한 결과 이식성과 상호 운용성에 있어서도 우수한 결과를 얻었다.

분산 컴퓨팅 플랫폼을 제공하는 DCE를 사용함으로써 분산 컴퓨팅에 따르는 프로그램의 분산과 자료의 분산, 분산된 자원의 위치 정보 관리, 이기종간의 통신 방식 조정 등의 문제점들을 쉽게 해결할 수 있었고, 시스템의 자원을 보다 효율적으로 사용할 수 있었다. 로그인 시스템의 경우 가입자 자료를 쉽게 n(=2)개로 분산할 수 있었고, 로그인 클라이언트들은 DCE에서 제공해주는 RPC와 CDS 기능을 이용해 로그인 서버의 위치 변동에 관계없이 서비스를 받을 수 있었다.

참고문헌

[1] Amjad Umar, 'Distributed Computing", Prentice-Hall, 1993

[2] Transarc, 'DCE Secure Core-System Administration", Transarc Co., 1993

[3] OSF, 'OSF DCE Application Development

Guide", OSF, Prentice-Hall Inc., 1993
[4] OSF, 'OSF DCE Application Development Reference", OSF, Prentice-Hall Inc., 1993
[5] OSF, 'OSF DCE User's Guide and Reference", OSF, Prentice-Hall Inc., 1993
[6] OSF, 'OSF DCE Adminstration Guide Introduction", OSF, Prentice-Hall Inc., 1993
[7] OSF, 'OSF DCE Adminstration Guide Reference", OSF, Prentice-Hall Inc., 1993
[8] OSF, 'OSF DCE Adminstration Guide-Extended Service", OSF, Prentice-Hall Inc., 1993
[9] OSF, 'OSF DCE Adminstration Guide-Core Components", OSF, Prentice-Hall Inc., 1993
[10] Ward Rosenberry, David Kenny, Gerry Fisher. 'Understanding DCE", O'Reilly & Associates Inc., 1992
[11] John shirley, 'Guide to Writing DCE Applications", O'Reilly & Associates Inc., 1992
[12] USENIX, 'The Distributed Computing Environment Remote Procedure Call System", Usenix Summer Technical Conference, 1993
[13] USENIX, 'OSF'S Distributed Computing Environment", Usenix Summer Technical Conference, 1993
[14] USENIX, 'Introduction To Threads, Posix Pthreads, And OSF/DCE Threads", Usenix Summer Technical Conference, 1993
[15] IBM. 'AIX DCE Overview", IBM, 1993
[16] Geoorge F. Coulouris & Jean Dollimore, 'Distributed Systems", Addison Wesley, 1988
[17] Harold W. Lockhart, Jr., 'OSF DCE", McGraw-Hill, Inc., 1994
[18] Alex Berson. 'Client/Server Architecture", McGraw-Hill, Inc., 1994
[19] Daniel Cerutti, Donna pierson, 'Distributed Computing Environments", McGraw-Hill, Inc., 1993
[20] DCI., 'OSF DCE User & Developer Conference", DCI., 1995
[21] OSF. 'AES/Distribured Computing RPC Volume", OSF, 1995
[22] Raman Khanna. 'Distributed Computing-Implemenation and management strategies", Prentice-Hall, Inc., 1994

성 재 모

1990 동국대학교 공과대학 전자계산학과(학사)
1993 Stevens Institute of Technology(석사)
1993. 9~현재 현재 데이콤종합연구소 분산시스템개발팀
관심분야 : 분산 컴퓨팅 시스템, 분산 OLTP, 망관리 시스템 등