

## 사례 발표

# CORBA기반 원격지 서버 관리 시스템

이 정 배\*

### ● 목 차 ●

1. 서 론
2. 사례 연구 및 배경
3. 목표 시스템의 설계
4. 목표 시스템의 구현
5. 결론 및 연구과제

## 1. 서 론

웹 기반의 기업 관리 시스템 (WBEM : Web-based Enterprise Management System) 은 NMS (Network Management System) 와 SMS (System Management Software) 그리고, EMS (Enterprise Management System) 을 통합한 형태로 발전하고 있다. 즉, 별개로 존재하던 전산환경이 통합되면서 일원화된 형태로 발전해 가고 있는 것이다. 이러한 기업 관리 시스템의 통합적인 개발에 있어 근간이 되는 기반 기술로 부각되고 있는 기술이 CORBA (Common Object Request Broker Architecture) 이다. 본 연구에서는 다기종 분산 서버 시스템의 통합적인 관리를 위한 WBEM 시스템을 CORBA 기반으로 설계 및 구현한다.

기존의 웹 기반의 기업관리 시스템에 관한 연구는 단일 서버 시스템 관리에 치중해 있다[23]. 이러한 관리 시스템들은 UNIX 운영 체제에서의 서버 시스템 관리 혹은 윈도우즈 NT 운영 체제에서의 서버 시스템 관리 등과 같이 각각의 서버 시스템 관리 모델 및 구조에 종속적으로 설계 및 구현되어

있다. 물론 각각의 서버 시스템 관리 모델은 최적화되어 있고 인터페이스 또한 이해하기 쉽도록 잘 구현되어 있다. 그렇지만 통합적인 형태의 분산 서버 시스템 관리에 대한 연구는 아직도 부족한 편이다. 현재의 기업관리 시스템은 단일 서버 시스템만 존재하지는 않기 때문에 통합적인 형태의 서버 시스템 관리에 대한 연구는 반드시 필요하다 할 것이다. 또한 같은 종류의 서버 시스템이라 하더라도 다수의 서버 시스템을 동시에 관리할 수 있는 서버 관리 시스템의 개발도 함께 고려되어야 할 것이다. 이러한 통합적인 형태의 웹 기반의 기업 관리 시스템을 위해 본 논문에서는 CORBA 기반의 원격 다기종 서버 관리 모델의 설계 및 구현에 관한 내용을 제시한다.

본 논문의 구성은 다음과 같다. 2장에서는 사례 연구 및 배경에 관한 내용을 기술한다. 3장에서는 목표 시스템의 설계에 관한 내용을 기술한다. 4장에서는 목표 시스템의 구현 및 인터페이스에 관한 내용을 기술한다. 마지막으로 5장에서는 결론 및 연구과제를 기술한다.

\* 부산외국어대학교 컴퓨터공학과 교수

## 2. 사례 연구 및 배경

### 2.1 OMA

OMA(Object Management Architecture)는 이종의 분산 환경에서 필요한 모든 기능을 정의하기 때문에 상당히 광범위하다. OMA가 제공하는 모든 서비스는 객체 개념을 기반으로 있다. OMA는 크게 이종의 분산 환경에서 통신을 담당하는 CORBA와 객체를 조작하는 데 필요한 각종 기본 기능들을 정의하고 있는 COSS(Common Object Service Specification), 그리고 추가적으로 제공되는 지원 서비스가 존재하고 있다[24]. 그리고 마지막으로 이들 기능들을 이용하여 사용자가 작성한 각종 응용 객체들이 존재한다. OMA구조에서 가장 중요한 것이 CORBA이며 현재 수많은 구현 제품이 존재하고 있다. 그러나 실제 CORBA만으로는 복잡한 응용 프로그램들을 작성하는 데 있어 충분한 기능을 제공하지는 못한다. 따라서 이들 필요한 기능을 모아서 만든 것이 CORBA 서비스이다. CORBA 서비스에서 제공하는 서비스는 아주 다양하고 광범위하다.

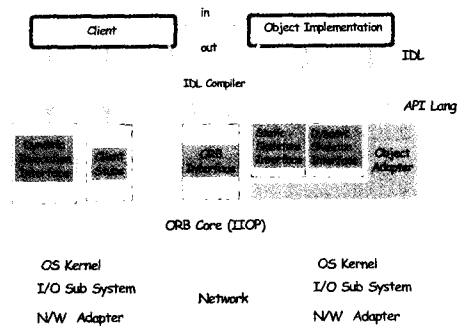
### 2.2 CORBA

OMA의 객체 지향 기술을 기반으로 하여 이종의 분산된 환경하에서 응용 프로그램들을 서로 통합할 수 있는 표준기술이 CORBA이다. 이것은 응용 프로그램간의 결합 뿐만 아니라 객체의 생성, 소멸에서부터 저장, 트랜잭션 기능에 이르기까지 분산 객체 환경에서 필요한 모든 서비스를 총칭하는 것이다. 이들 기능 중 CORBA는 컴퓨터 내부의 버스처럼 서로 다른 프로그램들 사이의 버스역할을 하는 모듈로서 OMA의 가장 중요한 요소이다. 결국 CORBA는 OMA의 한 부분이고 ORB(Object Request Broker)는 CORBA의 핵심기술을 말하는 것이다[14].

CORBA의 장점은 주요 미들웨어 업체들의 지원, 개방형 표준이라는 점이고 제한 사항은 모든

CORBA 서비스가 구현된 것은 아니라는 점과 사용자의 새로운 기술(IDL, Language Mapping(IDL to C++/JAVA) 등)들의 습득이 필요하다는 점이다[15].

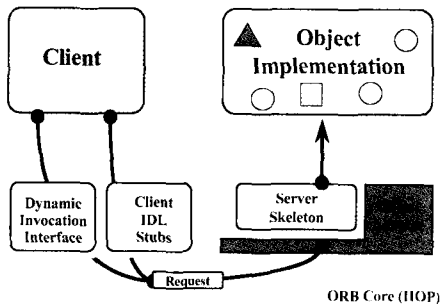
분산 객체 시스템을 위한 OMA에서 CORBA는 가장 중추적인 역할을 한다. CORBA는 분산 환경에서 객체간의 통신을 담당하기 때문이다. CORBA는 객체 지향 개념을 바탕으로 하여 객체의 메시지를 호출함으로써 서비스가 이루어진다.



(그림 1) ORB 구조

그림 1에서 보는 바와 같이 ORB는 클라이언트와 구현 객체(Object Implementation) 사이의 중계자 역할을 수행하는 여러 가지 컴포넌트와 인터페이스로 구성되어 있다. ORB를 단정적으로 말한다면 분산된 객체들을 클라이언트/서버 관계로 만들어 주는 핵심적인 객체 미들웨어라고 할 수 있다[14]. ORB는 클라이언트 객체가 구현 객체의 오퍼레이션을 호출할 때 ORB는 클라이언트의 호출을 받아 자신이 가지고 있는 구현 객체들의 정보를 참조하여 클라이언트가 원하는 구현 객체를 찾고 클라이언트에서 보내 준 서비스 요청 정보를 해당 구현 객체에게 전달하여 해당 객체로 하여금 그 작업을 수행하게 한다. 그리고 작업이 완료되면 ORB는 역으로 그 결과 값을 받아 클라이언트에게 전달하는 역할을 수행한다. ORB는 이렇게 클라이언트와 구현 객체들을 연결 시켜주는 역할을 해줌으로써 클라이언트는 구현객체가 어디에 있든 상관하지 않

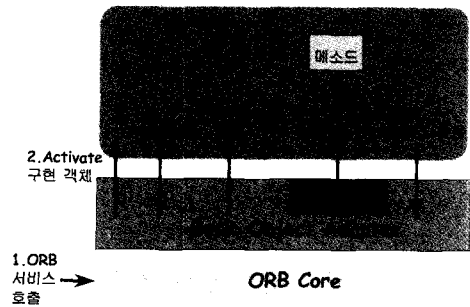
고 단지 원하는 객체의 오퍼레이션(operation)을 간단히 호출할 수 있게 된다[15].



(그림 2) CORBA의 호출 원리

CORBA의 호출 원리는 그림 2에서 보는 바와 같다. 일단 CORBA의 클라이언트 객체에는 클라이언트 스텀브와 프로그램 수행시 원하는 메시지를 호출하는 동적 호출(Dynamic Invocation) 기능을 통해 구현 객체를 호출할 수 있다[24]. 일단 클라이언트 객체의 호출이 발생하면 통신을 담당하는 ORB 기능을 통해 구현 객체에게 클라이언트 객체의 호출이 전달된다. 이때 구현 객체 쪽의 객체 어댑터(Object Adapter)는 기본적으로 해당 객체를 호출 가능하게 생성하고 원하는 호출에 해당하는 메시지를 실행시킨다. 일단 실행된 메시지에 의해서 처리된 결과는 다시 클라이언트 객체에게 전달된다. 클라이언트 객체에 의해서 요청된 서비스는 CORBA를 통해 원격지의 구현 객체에게 전달되고 구현 객체에 의해서 처리된 결과는 다시 클라이언트 객체에게 반환된다. 호출시 전달되는 정보로는 호출시 대상이 되는 객체와 메시지, 전달 인자 등이 있다.

CORBA의 구현객체 구동방법은 그림 3에서 보는 바와 같다. 먼저 ORB는 클라이언트 객체의 서비스 호출 요청을 받고 구현 저장소를 참조한 후 구현 객체의 활성화 여부를 검사한다[24]. 구현 객체가 활성화 중이지 않을 경우 활성화(Activate)시킨다. 이때, 구현 객체에게는 BOA와 통신하는 데 필요한 정보를 전달받는다. 구현 객체는 impl\_is\_ready 메



(그림 3) 구현 객체 구동방법

서드를 호출하여 BOA에게 자신의 존재를 등록한다. BOA는 등록된 구현 객체를 활성화시킨다. 그리고 BOA는 스켈로톤을 통해 클라이언트가 요청한 메시지를 호출한다. 구현 객체에 대한 모든 이용이 끝나면 BOA에서 제공하는 deactivate\_impl 메시지를 호출하여 구현객체를 비활성화시킨다.

CORBA는 다음과 같은 분야에 적용이 가능하다.

- 기존 시스템을 변경하지 않고 웹 환경으로 변환할 때
- 이기종 시스템간의 데이터 교환을 해야 할 때
- 이기종 데이터베이스간 통합이나 데이터 교환을 해야 할 때
- 고립되어 있는 메인프레임이나 중형 서버, PC를 서로 연동하여 운영하고 싶을 때
- 서버에 어떤 작업을 요청한 후 기다리지 않고 다른 일을 한 후, 서버가 응답을 주면 처리하고 싶을 때
- 기존 시스템을 객체지향모델로 재구성하고 싶을 때
- 기존 어플리케이션 기능이 부족하여 확장하고 싶을 때

### 2.3 분산 객체 기술

고성능 PC와 충분한 대역폭을 제공하는 네트워크에 의해 분산컴퓨팅이 가능하게 됨에 따라 기업 정보시스템은 다양한 이기종 컴퓨터 환경에서 데이터와 응용 프로그램을 분산 운영하면서 프로그

램의 상호 운영성과 사용자에게 분산의 투명성을 제공할 수 있게 되었다.

은행 정보 시스템은 은행의 각 지점별 BP(Branch Processor)가 지점 업무를 처리하고, 각 지점의 BP는 중앙시스템과 필요한 업무에 대하여 처리를 하는 전형적인 분산시스템의 예라고 할 수 있다. 이러한 분산시스템은 원격시스템간의 자원 공유, 개방성, 시스템 사이의 병렬성, 확장성, 고장감 내성, 위치 투명성 등을 제공하여야 한다.

또한 하드웨어 발달과 소프트웨어 발달의 불균형이 심화되면서 소프트웨어 개발의 새로운 전환이 요구됨에 따라 1980년대 객체지향 기술이 대두되었다.

객체지향기술이 제공하는 추상화, 캡슐화, 상속성, 다형성 등의 성질은 컴포넌트 프로그래밍을 가능하게 하였으며, 소프트웨어 생산성을 보장하는 소프트웨어 개발 방법론으로 성장하였다. 또한 객체들은 데이터와 그 데이터를 조작할 수 있는 메서드로 구성되어 있으며, 프로그램은 메시지에 의한 객체간 상호 작용을 기술함으로써 여러 문제를 해결한다.

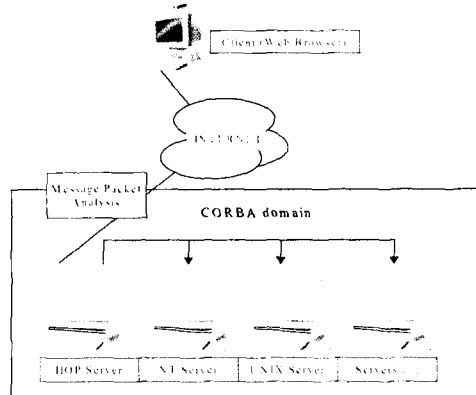
분산 객체 기술은 이러한 두 가지 기술의 장점을 효과적으로 통합하여 주는 기술이며, 개발자에게 어플리케이션 개발의 생산성 향상을 제공하고 사용자에게 분산 환경에 투명하게 통합된 정보를 제공하여 준다.

### 3. 목표 시스템의 설계

#### 3.1 시스템 구조

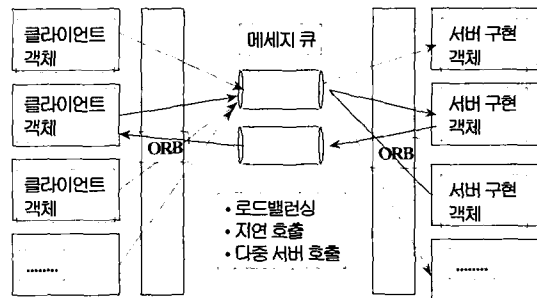
CORBA 기반 원격 대기종 서버 관리 시스템의 전체적인 구조는 그림 4에서 보는 바와 같다. 클라이언트에서는 웹브라우저를 통해 다수의 서버들이 위치하고 있는 CORBA 도메인 중 IIOP (Internet Inter-ORB Protocol) 서버로 접속하도록 한다. IIOP

서버는 CORBA 도메인 내의 다수 서버들의 현황을 동적으로 등록 및 유지할 수 있다. 즉, IIOP 서버 프로그램 수행시 다른 서버들이 자신의 존재를 등록할 수 있도록 한다.



(그림 4) CORBA 기반 원격 대기종 서버 관리 모델

클라이언트는 웹을 통해 IIOP 서버로 접속한다. 웹을 통한 접속시 보안 인증 절차를 거친 후 특정 서버에 대한 원격 관리 및 조회 명령을 그림 5에서 보는 바와 같이 메시지 패킷(Message Packet) 형태로 IIOP 서버에게 전송한다. 이런 형태의 메시지 패킷 전송을 통해서 로드 밸런싱의 효과를 가질 수 있다.



(그림 5) 메시지 패킷의 전송 원리

IIOP 서버는 클라이언트에서 전송된 메시지 패킷을 분석하여 어느 서버의 어떠한 관리 메시지만가를 확인한 후 해당 서버를 호출한 후 메시지를 수행하도록 한다. 해당 서버에서 클라이언트의 메

시지가 수행되고 나면 그 결과는 다시 IIOP 서버로 보내지고 IIOP 서버에서는 결과를 클라이언트로 전송한다. 즉 IIOP 서버가 클라이언트와 서버간의 중재자 역할을 하는 것이다. 이런 형태로 CORBA 도메인내의 모든 서버들을 동일한 웹 인터페이스를 통해 원격 관리가 가능하도록 처리한다. 주목할 부분은 CORBA 도메인 내에서는 모든 서버에 대해 동적인 등록이 이루어진다는 것과 클라이언트에서는 서버의 종류에 관계없이 동일한 인터페이스하에서 작업이 이루어진다는 점이다.

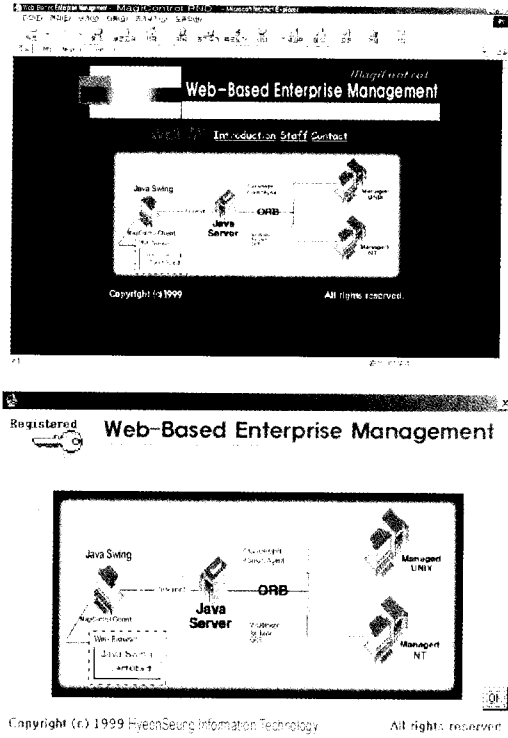
#### 4. 목표 시스템의 구현

본 시스템은 자바 언어로 구현되어 있기 때문에 플랫폼에 관계없이 자바가상기계(Java Virtual Machine)가 동작하는 환경이면 실행이 된다. 본 시스템의 구현 환경은 다음과 같다.

- 자바언어 : JDK1.2
- CORBA 프로그래밍 : Visibroker 3.0
- 서버 운영체제 : Windows NT, UNIX
- 서버 시스템 호출 개발 언어
  - Windows NT(ADSI), UNIX(셸 스크립트)
- 웹 서버 소프트웨어 : 아파치 웹서버
- 클라이언트 개발 도구 : JBuilder 3.0

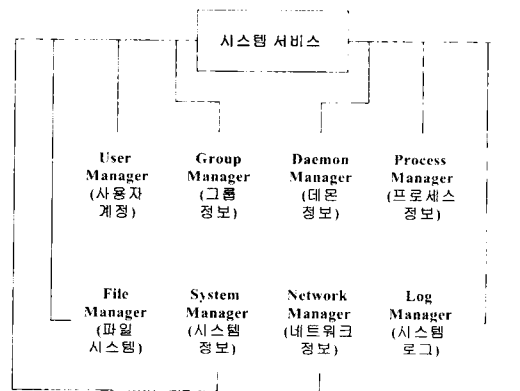
CORBA 기반 원격 다기종 서버 관리 시스템의 사용자 인터페이스는 그림 6에서 보는 바와 같다. 클라이언트의 요청에 동작할 수 있도록 CORBA 도메인 내의 특정 서버를 IIOP 서버로 선택하고 웹서비스를 지원하도록 한다. 상단 그림은 접속시의 홈페이지가 되고 하단 그림은 홈페이지를 클릭 했을 때 동작하는 자바 애플릿으로 JBuilder의 컴포넌트를 이용하여 구현하였다.

서버 시스템마다 특징이 다른 면들이 있지만 서버 관리시 비중이 큰 서비스들을 구분하여 그림 7에서 제시하는 서버 시스템의 관리 서비스 그룹을



(그림 6) 목표 시스템의 사용자 인터페이스

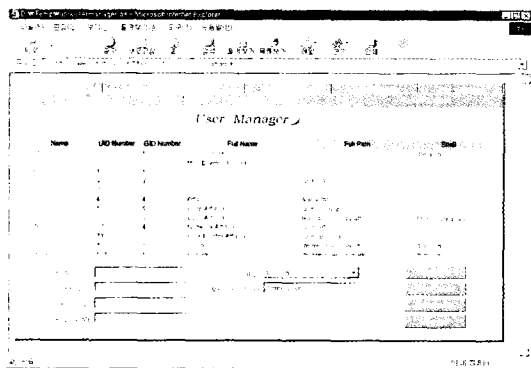
만들었다. 즉, 서버 시스템에서 빈번하게 일어나는 시스템 서비스들만 제공하도록 하였다. 그림 8에서 보는 바와 같이 사용자의 계정에 관한 서비스, 그룹 정보에 관한 서비스, 데몬 정보에 관한 서비스, 프로세스 정보에 관한 서비스, 파일 시스템을 조작



(그림 7) 서버 시스템 관리 서비스

할 수 있는 서비스, 시스템 정보를 확인할 수 있는 서비스, 네트워크 정보를 확인할 수 있는 서비스, 시스템에서 생성되는 로그 정보를 확인할 수 있는 서비스들로 구분하였다.

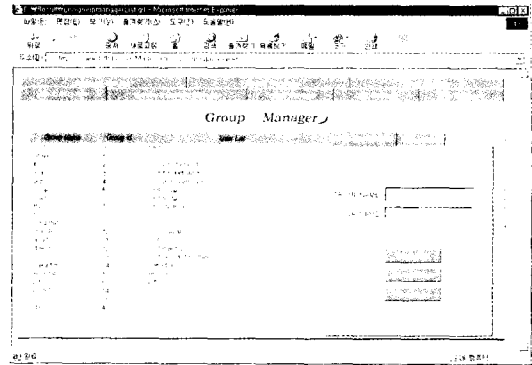
그림 8에서 보는 바와 같이 User Manager는 사용자의 계정 등록, 삭제, 갱신, 조회 등의 서비스를 담당한다. 그림의 결과는 서버의 사용자를 조회한 결과로서 그림에서 보여지는 것처럼 시스템에 등록되어 있는 다양한 사용자들의 로그인 이름, UID(User ID), GID(Group ID), 주식, 홈 디렉토리, 사용하고 있는 셸 정보 등을 조회할 수 있다. 이러한 조회 명령 외에도 사용자의 UID를 이용한 계정 삭제, 갱신 및 새로운 사용자의 등록이 가능하다.



(그림 8) User Manager 인터페이스

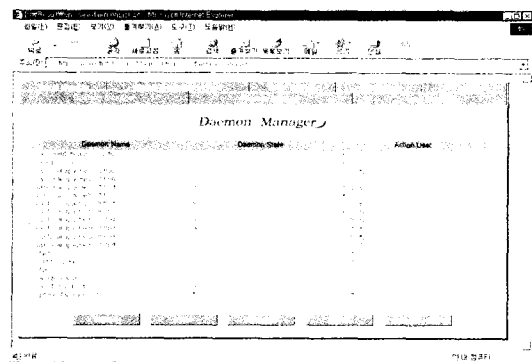
그림 9에서 보는 바와 같이 Group Manager는 사용자들이 속해 있는 그룹의 등록, 삭제, 갱신, 조회 등의 서비스를 담당한다. 그림의 결과는 서버의 그룹 사용자의 조회 명령의 결과로서 그룹 사용자의 로그인 이름, GID, 그룹에 속한 사용자들의 리스트 등을 확인할 수 있다. 이러한 조회 명령외에도 GID를 이용한 그룹 사용자의 삭제, 갱신 및 새로운 그룹 사용자의 등록이 가능하다.

그림 10에서 보는 바와 같이 Daemon Manager는 시스템에서 서비스하고 있는 각종 데몬들의 서비스 시작, 종료, 재시작, 조회 등의 서비스를 담당한다.



(그림 9) Group Manager 인터페이스

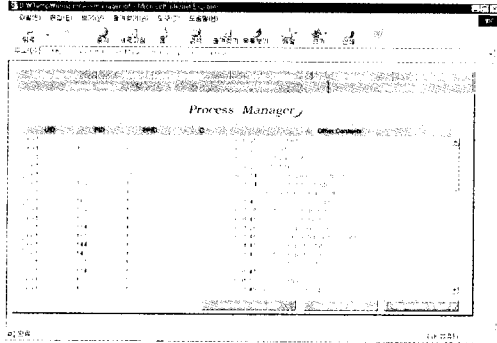
다. 그림의 결과는 서버의 데몬 리스트 조회 명령의 결과로서 데몬 이름, 데몬의 상태 정보, 데몬의 소유주 등을 확인할 수 있다. 조회 명령을 통해 나타난 데몬 정보를 이용해서 시작, 종료, 재시작 등의 서비스를 할 수 있다.



(그림 10) Daemon Manager 인터페이스

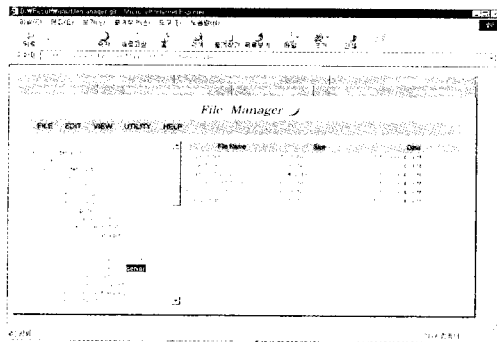
그림 11에서 보는 바와 같이 Process Manager는 시스템에서 서비스되고 있는 프로세스들의 서비스 종료, 재시작, 조회 등의 서비스를 담당한다. 그림의 결과는 프로세스 리스트 조회 명령의 결과로서 프로세스 이름, 프로세스의 PID(Parent ID), PPID(Parent Parent ID), 실행 환경 등을 확인할 수 있다. 조회 명령을 통해 나타난 프로세스 정보를 이용해서 재시작, 종료 등의 서비스를 할 수 있다.

그림 12에서 보는 바와 같이 File Manager는 시



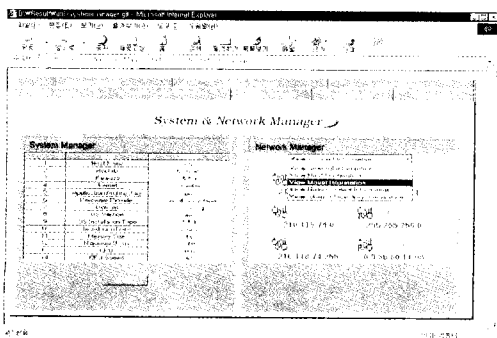
(그림 11) Process Manager 인터페이스

시스템에 있는 파일 시스템의 관리를 담당한다. 시스템에 존재하는 파일 시스템을 디렉토리별로 구분해서 보여 주고 파일 삭제, 보기, 간단한 편집 등의 기능을 제공한다.



(그림 12) File Manager 인터페이스

그림 13에서 보는 바와 같이 System Manager는 해당 서버 시스템이 갖는 로그내용의 조회를 담당



(그림 13) System &amp; Network Manager 인터페이스

하고 Network Manager는 해당 시스템 네트워크의 정보를 확인해 준다. 마지막으로 Log Manager는 IIOF 서버에서 동작하는 서비스로서 목표 시스템이 자체적으로 갖는 로그 정보를 확인해 준다.

## 5. 결론 및 연구 과제

본 논문에서는 시스템 관리자의 시스템 관리 모델에 대한 새로운 방법을 제시하였다. 기존의 서버 관리 시스템의 한계는 운영체제에 종속적이라는 데에 있다. 다양한 서버가 존재하는 기업환경에서는 매우 불편하고 힘든 환경이라 할 수 있겠다. 물론 운영체제에 매우 종속적인 부분에 대한 시스템 서비스는 운영체제에 종속적인 서버 관리 시스템을 사용하여야 할 것이다. 그렇지만 다수의 서버 시스템들에 대해 매번 똑같은 작업을 한다거나 이 기종의 서버 시스템에 대해 매번 재교육을 필요로 한다든지 하는 점들은 매우 비능률적인 요소로 작용할 것이다.

이런 관점에서 본 논문에서 제시하는 다수의 이 기종 서버 시스템들을 CORBA 도메인에서 일관된 사용자 인터페이스로 시스템을 관리하도록 하는 모델은 중요한 의미를 지닐 수 있다고 본다.

향후 연구 과제로 고려되는 것은 다음과 같다. 먼저 서버 관리 시스템의 실시간성을 보장할 수 있는 모듈의 개발이 필요하다. 그리고, 서버 시스템의 상황을 지속적으로 모니터링할 수 있는 모듈의 개발이 필요하다.

## 참고문헌

- [1] Bryan Morgan, CORBA meets Java, JavaWorld, October 1997.
- [2] CORBA specification, OMG, 1994.
- [3] CORBA Service:COMMON Object Services Specification, OMG, 1995.

[4] Curtis David, Java, Rmi and CORBA, OMG, 1997.

[5] Distributed Object Technology in action, Iona, 1996.

[6] Doug Pedrick, Jonathan Weedon, Jon Goldberg, Erik Bleifield, Programming with VisiBroker, John Wiley & Sons, Inc., 1998.

[7] Fowler M., "A Comparison of Object-Oriented Analysis and Design Methods", Quoin Technical Report TR-MI. Boston, MA, 1995.

[8] Garg Rohit, CORBA: The Future of Web Computing, 1997.

[9] Guy Eddon, Henry Eddon, "Inside Distributed COM", Microsoft PRESS, 1998.

[10] Islam Nayeem, "Distributed Objects Methodologies for customizing Systems Software", IEEE Computer society Press, 1996.

[11] Markus, Tresch, "Middleware for Distributed Object Management", Tutorial CILM96, 1996.

[12] VISIGENIC Technologies Ltd, VisiBroker for Java, 1997, Available from <http://www.visigenic.com>, 1997.

[13] 박남섭, 이정배, "웹을 기반으로 한 원격 UNIX 관리 시스템 구현에 관한 연구", '98 춘계 학술발표 논문집, 한국정보처리학회, 제9권, 제1호, 1998. 4.

[14] 박재현, WWW와 분산객체시스템, WWW-KR Workshop, Aug 17 1996.

[15] 박재현, ObjectWeb 기반의 시스템 통합기술, URL:<http://www.shinbiro.com/~jhpark>, 1997.

[16] 전병선, "Inside ATL/COM Programming with Visual C++", Microsoft PRESS, JUN 1998.

## 저자약력



### 이 정 배

1981년 경북대학교 전산공학과(공학사)  
 1983년 경북대학교 전산공학과(공학석사)  
 1995년 한양대학교 전산공학과(공학박사)  
 1982년-1991년 한국전자통신연구원 선임연구원  
 1996년-1997년 U.C.Irvine 객원교수 Dept.of Electrical & Computer Eng.  
 1991년-현재 부산외국어대학교 컴퓨터공학과 부교수  
 관심분야: 분산 UNIX 시스템, 원격 영상감시 및 제어, 멀티미디어 서버