통합데이터베이스 환경에서 CORBA 기반의 디렉토리 서비스 설계

박재형†, 김기봉‡, 진성일†
(† 충남대학교, ‡대전보건대학)

Abstract

In CALS/EC environment, there are geographically dispersed heterogeneous databases and various applications. Integrated database environment provides transparent access mechanism for heterogeneous and distributed databases and interoperability between applications in various platforms. Because elements exist geographically in integrated database environment, directory service is necessary for providing unique access mechanism for elements. In this paper, we design a directory service using CORBA based access interface and object-oriented DIB model for accessing transparently to elements and interoperability with other information system.

I. 서론

통합데이터베이스(Integrated Database)는 CALS/EC 구현을 위한 가장 중요한 분야로서 개념적으로는 제품의 수명주기활동에 관여하는 각각의 사람들이 다루게 되는 자료들을 지리적 원근이나 하드웨어, 소프트웨어, 정보통신망 등 플랫폼의 상이함에 관계없이 쉽게 생산, 관리 등의 분야에 활용할 수 있는 환경을 의미한다[1].

현재 통합데이터베이스 구현을 위한 참조모형(Reference Architecture)은 미국방부가 구상하고 있는 IWSDB(Integrated Weapon System Database)와 IWSDB를 민간 개념으로 확대시킨 IDE(Integrated Data Environment) 및 제조업 중심의 가상기업의 시험구현을 위해 정부와 민간이 공동추진 중인 NIIIP(National Industrial Information Infrastructure Protocols) 등이 있다[2].

통합데이터베이스 환경은 기본적으로 여러 가지 통합데이터베이스를 기반으로 하는 어플리케이션들이 분산되어 존재하고, 이 어플리케이션들을 사용하는 사용자

들이 지리적으로 분산되어 존재하게 된다. 또한 각 어플리케이션들이 운영되는 플랫폼이 상이하고, 사용자가 사용하는 클라이언트 환경이 각기 다를 수 있다. 이러한 이질적인 플랫폼을 지원하기 위한 방법으로 CORBA 나 DCOM 등의 미들웨어를 사용하고 있으나 대부분의 통합데이터베이스 참조 모형들은 대부분 CORBA(Common Object Broker Architecture)에 기반하는 시스템 모델을 제시하고 있다.

기존에 존재하는 X.500 이나 LDAP 등의 디렉토리 시스템들은 네트워크 프로토콜 형태로서 제공되므로 통합데이터베이스 환경에서 사용될 경우 다른 시스템들과의 상호 연동을 위해서는 동일한 네트워크 프로토콜을 사용해야만 하는 문제점이 있을 수 있고, 분산되어 존재하는 클라이언트 어플리케이션들이 디렉토리 시스템에 투명하게 접근하지 못하는 문제점이 발생할 수 있다.

그러나 CORBA 기반의 디렉토리 서비스를 구성할 경우 다양한 종류의 시스템들이 가질 수 있는 플랫폼의 상이함을 극복할 수 있고, 각 시스템들 간의 상호 연동성을 극대화 시킬 수 있으며, 각각 지리적으로 분산되어 존재하는 클라이언트 어플리케이션들이 시스템에 접근하는데 있어서 투명성을 보장할 수 있다[3].

따라서 본 논문에서는 통합데이터베이스 환경에서 존재하는 각 자원에 대한 정보 및 사용자 정보, 조직체 정보 등을 저장하고, 이를 필요로 하는 각 어플리케이션들에게 서비스를 제공하기 위한 디렉토리 서비스 시스템을 구성하기 위하여 디렉토리 서비스 시스템의 구조를 설계하였고, 디렉토리 모델을 객체지향 모델링기법을 사용하여 모델링을 수행하였으며, 디렉토리 접근 인터페이스를 CORBA 의 ORB 기반으로 설계하였다.

II. 디렉토리 서비스 표준 분석

1. X.500 디렉토리 서비스

X.500 디렉토리 서비스는 정보 통신망에 필요한 정보들을 특정한 규칙에 맞게 데이터베이스로 구축할 수 있는 여러 가지 규정과 구축한 정보를 바탕으로 사용자의 요청을 처리할 수 있는 여러 가지 편리한 기능 및 서비스를 규정하여 놓은 것이다[3].

X.500 디렉토리 서비스는 ITU 와 ISO/IEC/JTC/SC21 협동 프로젝트로 1985 년부터 개발에 착수하여 1988 년도에 X.500 시리즈 권고사항으로 발표된 이래 각 국가 별로도 표준화 단체 및 관련 기관에 의해 꾸준히 표준화 작업이 진행되고 있다. 1992

년까지는 많은 기관들이 X.500 디렉토리 서비스 서버를 설치하여 약 300,000 개의 엔트리를 가지며, 인터넷에 연결된 370 여 기관에서 서비스가 이루어졌다. 이제 디렉토리 표준은 인터넷에서 방대한 자원을 관리하기 위한 필수 조건이 되고 있는 것이다.

X.500 디렉토리 서비스 시스템은 디렉토리에 서비스를 요구하는 디렉토리 사용자와 디렉토리간의 인터페이스를 담당하는 DUA(Directory User Agent)로 구성되며 이는 그림 1 과 같다. 디렉토리의 기본 개념은 실생활에 존재하는 정보들을 사용자가 사용하는데 편리하도록 구성한 것이다. 따라서 우선적으로 디렉토리에는 이를 구성하는 정보들이 필요하다. 또한 디렉토리의 정보는 결국 사용자가 이용하기 위한 것이므로 사용자를 위한 응용 프로세스가 필요하다. 디렉토리에 사용자의 요구를 전달하고, 디렉토리로부터 사용자의 요구에 대한 응답을 받는 응용 프로세스에는 디렉토리와 사용자 사이의 인터페이스 역할을 수행한다.

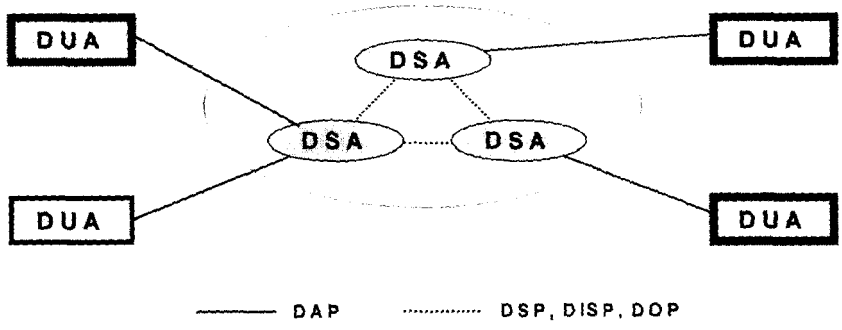

그림 1 X.500 디렉토리 서비스 시스템

이 응용 프로세스가 DUA(Directory User Agent)이다. 또한 디렉토리 내에 존재하는 정보들은 각각의 시스템에 분산되어 관리, 운영되기 때문에 이를 디렉토리 내에서 디렉토리 프로토콜을 이용하여 디렉토리 정보를 보유하고 있는 시스템들과 상호 협력해야 한다. 또한, 사용자의 요구를 수행하는 응용 프로세스가 존재하여 최종적으로 사용자는 분산된 디렉토리 전체를 하나의 지역 디렉토리로 간주하여 사용할 수 있다. 분산된 디렉토리 시스템에서 사용자의 요구를 디렉토리 프로토콜에 따라 수행하는 응용 프로세스들이 DSA(Directory Server Agent) 이다.

DSA 는 사용자가 지시한 명령이나 요구들에 대해서 디렉토리를 통해서 그것을 수행하고 그것의 결과를 사용자에게 응답하는 역할을 하며 디렉토리에는 여러 개의

DSA가 존재할 수 있다. DUA는 DSA에 있는 접근점을 통하여 디렉토리에 접근하게 되는데 하나의 DSA는 하나 이상의 접근점을 가지고 있다. 이들 구성요소 사이에 사용되는 프로토콜은 다음과 같다.

- DAP(Directory Access Protocol) : 디렉토리 서비스를 요구하는 하나의 DUA와 DSA 사이의 프로토콜
- DSP(Directory System Protocol) :디렉토리 서비스의 연결을 지원하는 두 DSA 사이의 프로토콜
- DSIP(Directory Information Shadowing Protocol) : 복제 정보의 교환을 지원하기 위하여 shadowing 등의 선정에 관한 두 DSA 사이의 프로토콜
- DOP(Directory Operational Binding Protocol) : 두 DSA 사이의 관리 정보 교환에 관한 프로토콜

이와 같이 X.500 디렉토리에서 클라이언트와 디렉토리간의 통신을 위하여 사용되는 프로토콜인 DAP은 일반적인 기업환경에 필요 없는 많은 기능을 가지고 있고, 프로토콜의 복잡성으로 인하여 구현이 어려우며, 클라이언트에 과다한 부하가 걸린다는 문제가 있다. 따라서 DAP의 복잡성을 해결하고, 일반적인 기업환경에 적합한 디렉토리와 클라이언트 사이에 새로운 프로토콜인 LDAP이 제안되었다.

2. LDAP(Lightweight Directory Access Protocol)

LDAP은 인터넷 TCP/IP 사용자에게 OSI 프로토콜로 구축된 X.500 디렉토리 서비스 시스템을 접근할 수 있도록 인터넷 TCP/IP 프로토콜 기반 위에 구축된 프로토콜이다. DAP이 OSI 기반으로 설계되었기 때문에 매우 복잡한 체계를 가지고 있어 접속하는 클라이언트와 디렉토리에 많은 부하를 주게 되지만, LDAP은 TCP/IP를 기반으로 하고 있으므로 클라이언트와 디렉토리에 대한 부하를 줄였다. 이에 따라 현재 개발되고 있는 대부분의 디렉토리 서비스 시스템은 LDAP에 기반하여 개발되고 있다.

DAP에 비해 LDAP이 가지는 장점을 살펴보면 다음과 같다.

- **TCP/IP 기반의 구현이다.**
  DAP이 OSI stack에 기반하여 동작하는 것과는 달리 LDAP은 TCP/IP에 기반하여 동작한다. 이전의 DAP이 기반으로 하고 있는 OSI 7 계층과는 달리, LDAP에서 기반으로 하고 있는 TCP/IP는 4계층으로 구성되어 있어 캡슐화 된 데이터의 길이가 OSI의 경우 보다 짧고, 그 결과 전송률 또한 개선

된다. 또한 기존의 네트워크를 사용 하는 대부분의 사용자들이 TCP/IP 를 기반으로 하고 있는 인터넷 사용자이므로 이들 사용자들에 대해 LDAP 을 구축하기가 편리하다.

- **일반적인 기관에서는 필요로 하지 않는 X.500 연산자 수를 줄였다.**
  LDAP 에서 비슷한 기능을 하는 DAP 연산자들을 하나의 연산자로 표현하여 연산자 수를 줄이고, 그 결과 사용자들이 고려해야 할 연산자 수가 줄어들어 서비스 받기가 좀 수월해 졌다.

- **데이터를 전송하기 위해 DAP 이 복잡한 ASN.1 방식의 문자열을 사용하는 것에 비해 LDAP 은 간단한 문자열을 사용한다.**
  DAP 에서는 데이터의 ASN.1 방식의 표현과 BER 인코딩으로 인한 복잡성이 문제가 되었으나, LDAP 에서는 데이터 원소들과 특히 DN 에 대하여 간단한 스트링 인코딩을 허용함으로써 데이터 길이를 줄이고 단순화하였다.

- **디렉토리에 접근하기 위해 DAP 이 디렉토리 서버의 주소를 사용하는 것에 비해 LDAP 은 URL 을 사용한다.**
  DAP 에서는 디렉토리 서버의 주소로써 DSA 의 접근점을 사용하였는데 URL 주소체계를 사용하고 있는 인터넷 사용자들이 사용하기에는 어려움이 있었다. 이에 LDAP 에서는 이를 개선하여 URL 을 LDAP 서버의 주소로 사용 하여 사용자들이 쉽게 이용할 수 있다.

LDAP 은 기본적으로 사용자의 이용 편리성에 중점을 두어 개발된 디렉토리 접속 프로토콜로서, IETF 에 의해 RFC 1487 표준으로 제정되었다. 현재는 버전 2 가 RFC 1777 표준으로 정의되어 있고, 버전 3 이 RFC 2251 로 정의되어 대다수 디렉토리 서버 업체에서 버전 3 을 지원하는 디렉토리 서버를 개발하고 있다. LDAP 버전 2 까지는 DAP 에서 사용하는 많은 기능을 삭제하고 간단한 디렉토리 서버와 클라이언트 간의 통신 프로토콜을 만들려고 노력하였다. 그러나, LDAP 이 기업환경에 적용됨에 따라 점차 사용자의 요구 사항이 많아지기 시작하였으며, LDAP 버전 2 를 가지고는 사용자가 요구하는 모든 기능을 수행하는데 어려움이 있었다. 이에 따라 LDAP 버전 3 이 정의되었고, LDAP 버전 3 에서는 Replication 및 Referral 등의 많은 기능이 추가됨에 따라 프로토콜은 다소 복잡하게 되었으나, DAP 보다는 간단하며 사용자의 요구에 부응할 수 있는 프로토콜이 되었다.

현재 대부분의 네트워크 이용자들은 인터넷을 광범위하게 사용하고 있고 이러한 인터넷은 기본적으로 TCP/IP 를 기반으로 하고 있다. 이에 LDAP 은 기존의 DAP 과

동일한 기능을 사용자들에게 제공하면서도 TPC/IP 상에서 동작할 수 있도록 구현되었다. 또한 사용자들이 고려해야 할 연산자들을 수를 줄여 사용자들이 서비스를 받는데 고려해야 할 부분 중 많은 부분을 개선하였다. 결국 LDAP은 기존의 DAP과는 다른 형태를 가지게 되어 X.500 디렉토리 서버와는 호환성을 가질 수 없게 되었는데, 이를 해결하기 위하여 LDAP에서는 LDAP Server를 두어 이런 문제점을 해결하였다.

III. 통합데이터베이스 환경에서 디렉토리 서비스 설계

1. 디렉토리 서비스 시스템 구조

본 논문에서는 통합데이터베이스 환경에서 동작하는 디렉토리 서비스 시스템을 설계하기 위하여 다음 그림 2와 같은 디렉토리 서비스 시스템의 구조를 설계하였다.

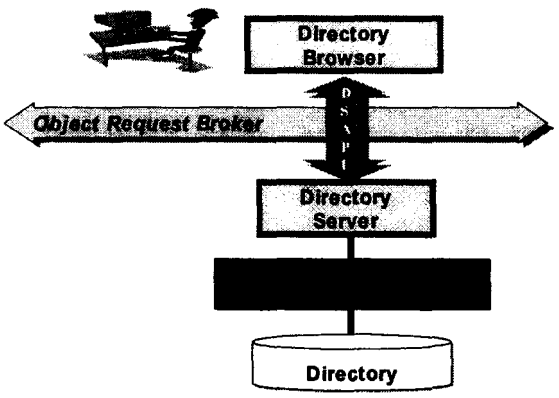

그림 2 디렉토리 서비스 시스템 구조

그림 2의 디렉토리 서비스 시스템 구조에서 각 구성 요소들은 다음과 같은 기능을 갖는다.

- **Directory Browser**
  사용자의 요구를 받아들여 디렉토리 서버에 전달하는 디렉토리 서비스를 사용하는 분산객체 환경에서 동작하는 어플리케이션들을 나타낸다.

- **Directory Server**
  X.500의 DSA 또는 LDAP의 LDAP Server에 해당하는 구성요소로서 클라이언트로부터 요청된 요구들을 받아서 데이터베이스 질의어로 변환하고,

변환된 질의를 Database Service Interface 를 통하여 수행하는 역할을 하는 디렉토리 관리 서버이다.

- **Object Request Broker**
  OMG 에서 제안한 CORBA 의 핵심 구성요소로서 클라이언트의 객체에 대한 서비스 요청을 서버부분의 객체에 전달하는 클라이언트와 서버 간의 중간자 역할을 수행한다.

- **DSAPI(Directory Server API)**
  클라이언트 어플리케이션이 Directory Server 에 서비스를 요청하기 위해 사용하는 CORBA 기반의 API 들의 집합으로서 DAP 또는 LDAP 의 각 연산자들에 해당한다.

- **Database Service Interface**
  Directory Server 에 의해 변환된 데이터베이스 질의어를 실제로 수행하는 부분으로 DBMS 가 제공하는 인터페이스이다.

- **Directory**
  X.500 에서 제공하는 디렉토리 정보 모델에 따라 실제로 디렉토리 정보들이 저장되는 데이터베이스이다.

2. 디렉토리 모델

X.500 의 디렉토리 정보 모델에는 DIB (Directory Information Base) 구현에 대한 상세한 기술은 포함하고 있지 않고, DIT (Directory Information Tree)라는 구조적인 형태를 제안하였다[4]. 그러나 DIT 는 DIB 의 구조적인 형태만을 표준화한 것이고 실질적인 DIB 구축에 관한 문제는 무슨 모델로 어떻게 구현하느냐에 따라 그 성능이 크게 좌우될 수 있다[5].

디렉토리 정보들을 저장하기 위한 방법으로 파일 시스템을 사용하거나, 관계형 데이터 모델, 또는 객체지향 모델에 입각한 데이터베이스들을 사용하는 방법들이 사용된다. 각 방법들은 각각 장단점을 동시에 가지고 있기 때문에 디렉토리의 성능을 위하여는 구현 도메인의 특성에 따라서 저장 방법이 선택되어져야 할 것이다[6].

본 논문에서는 디렉토리 데이터베이스 구축의 방안으로 X.500 의 DIB 모델에 따라 객체 지향 모델링을 통한 구축 방안을 선택하였다. 다음 그림 3 은 본 논문에서

제시하는 DIB 의 객체지향 모델이다.

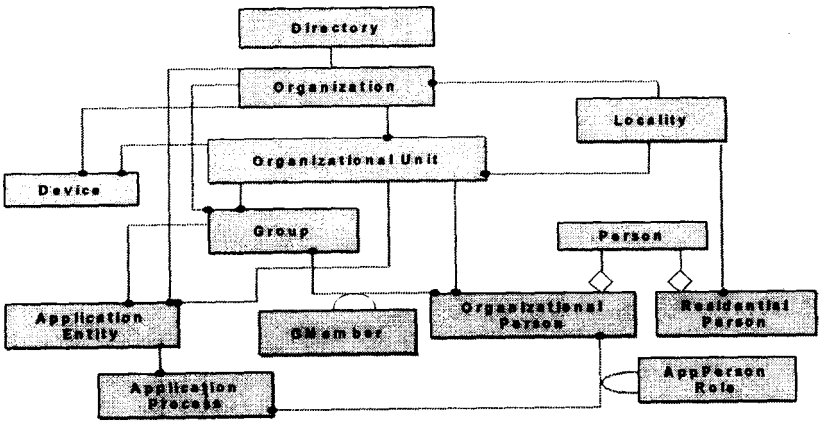

그림 3 DIB 객체지향 모델

그림 3 의 DIB 의 객체지향 모델을 이루고 있는 각 Object Class 들은 X.500 과 LDAP 에서 정의한 Object Class 들 중에서 간략화한 것이고, 객체지향 모델로 변환하면서 GMemeber 와 AppPersonRole 객체클래스가 새로이 포함되었다.

3. 디렉토리 접근 인터페이스(DSAPI)

디렉토리 접근 인터페이스들을 정보 모델 측면에서 특정 디렉토리 엔트리 정보를 얻고자 하는 읽기(Read) 동작과 여러 개의 디렉토리 엔트리 정보를 얻고자 하는 검색 (Search) 동작, 그리고 디렉토리 수정(Modify) 동작과 디렉토리 처리오류(Error) 동작 등으로 분류된다[7].

X.500 및 LDAP 에서는 디렉토리 추상 서비스를 정의하여 디렉토리 서비스를 기술하고 있는데, 디렉토리 추상 서비스에 필요한 공통인수 처리, 처리 절차, 결합 및 해제 동작, 읽기 동작, 탐색 동작, 수정 동작, 오류 등을 정의하였다[7].

본 논문에서는 이와 같은 디렉토리 추상서비스에 기초하여 통합데이터베이스 환경에서 여러 어플리케이션들이 디렉토리에 응답을 요구하는 요청을 내릴 수 있도록 CORBA 를 기반으로 하는 디렉토리 서비스 인터페이스(DSAPI)를 정의한다. X.500 과 LDAP 에서는 인터페이스 정의를 위하여 ASN.1 을 사용하지만[8], CORBA 의 IDL 을 사용하여 분산 객체 인터페이스를 정의한다.

그림 4는 클라이언트 어플리케이션과 디렉토리 서버 간의 인터페이스를 나타낸다.

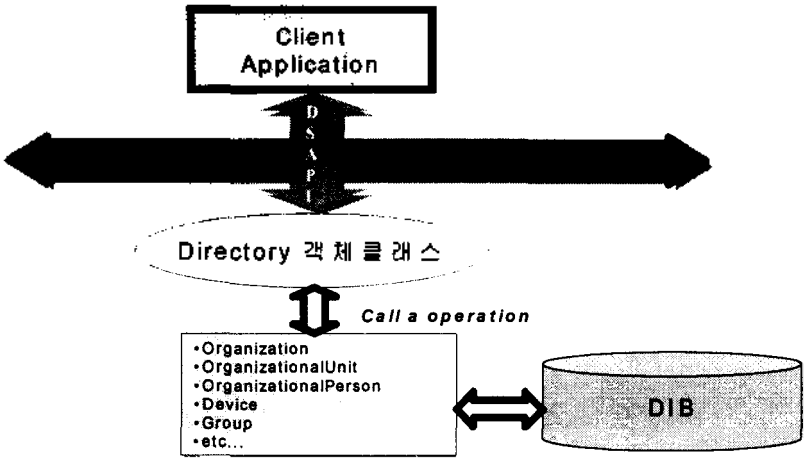

그림 4 서비스 인터페이스

그림 4와 같이 클라이언트 어플리케이션들에게 제공되어지는 디렉토리 서비스 인터페이스(DSAPI)는 Directory 객체클래스의 오퍼레이션들이다.

Directory 객체클래스는 자신의 오퍼레이션들을 통하여 전달된 클라이언트의 질의를 처리할 적당한 객체클래스를 선택한 후, 선택된 객체클래스의 오퍼레이션을 호출한다. 호출된 객체클래스는 전달된 질의를 데이터베이스 질의어로 변환하여 실제 정보가 저장되어 있는 DIB에 접근하게 된다.

디렉토리 서비스 인터페이스, 즉 Directory 객체의 오퍼레이션들은 다음과 같이 정의된다.

```
Interface Directory {
LDAPResult addEntry(in AddRequest addRequest);
LDAPResult modEntry(inout ModRequest modRequest);
LDAPResult delEntry(inout DelRequest delRequest);
LDAPResult read(inout ReadRequest readRequest);
LDAPResult search(inout ReadRequest readRequest);
// 이하 생략
}
```

각 객체클래스들도 다음과 같은 오퍼레이션들을 갖게 되고, Directory 객체클래스에 의하여 호출된다.

- 디렉토리에 대한 질의
  - ➡ read, compare, list, search, abandon, modify

- 디렉토리에 대한 수정
  - ➡ addEntry, removeEntry, modifyEntry, modifyDN

지금까지 정의한 디렉토리 서비스 인터페이스의 동작 과정 중에서 "박재형"이라는 사람에 대한 read 연산의 수행 과정은 그림 5 와 같다.

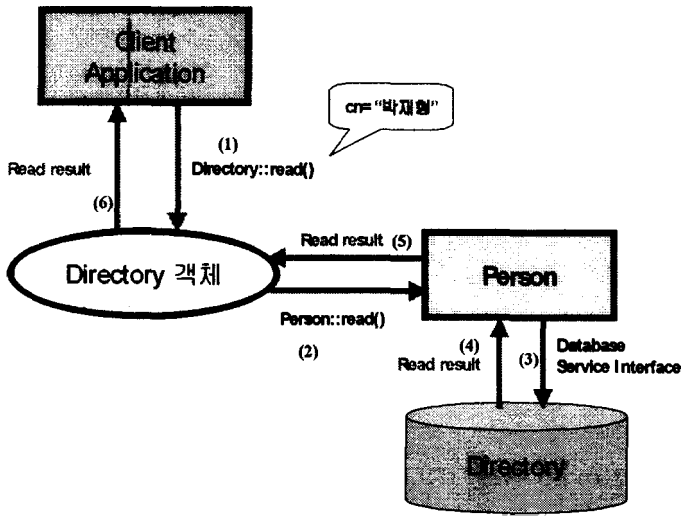

그림 5 read 연산의 수행 과정

1. 클라이언트에서는 "cn=박재형"이라는 DN(Distinguished Name)을 Directory::read()의 매개변수 형태로 Directory 객체에 전달한다.

2. 디렉토리 객체는 이 요구가 Person 객체에 대한 오퍼레이션임을 해석하여 Person::read() 오퍼레이션을 호출한다.

3. Person::read()에서는 데이터베이스 질의어로 변환하고, 데이터베이스 서비스 인터페이스를 이용하여 읽기 연산을 수행한다.

4. Person 객체는 읽기 결과를 Directory 객체에 반환한다.

5. Directory 객체는 읽기 결과를 클라이언트에 반환한다.

IV. 결론 및 향후 연구

통합데이터베이스 환경에서 존재하는 다양한 시스템들은 분산되어 있고, 플랫폼들이 서로 상이하므로, 각 시스템들 간의 상호 연동성이 필요하다. 이를 위해 IWSDB 와 IDE, NIIIP 등의 통합데이터베이스 참조 모형에서는 CORBA 기반의 시스템 모델을 제시하고 있다.

X.500 이나 LDAP 등과 같은 디렉토리 표준들은 각각 OSI 나 TCP/IP 와 같은 네트워크 프로토콜에 기반하고 있으므로 네트워크 환경이 상이한 경우 서비스를 원하는 클라이언트 어플리케이션들이 서비스를 이용할 수 없게 된다. 따라서 각각 클라이언트 어플리케이션들의 상이한 플랫폼들과 프로토콜들에 대한 투명성을 보장하기 위한 디렉토리 서비스 시스템이 필요하다.

따라서 본 논문에서는 CORBA 의 Common Facility 로 동작하는 디렉토리 서비스 시스템의 구조와 디렉토리 모델, 그리고 접근 인터페이스를 설계하여 각 클라이언트 플랫폼에 독립적이고, 투명하게 디렉토리 서비스 시스템에 접근할 수 있으며, 통합데이터베이스 환경에서 다른 시스템들과의 상호 연동되어 동작할 수 있도록 하였다.

CORBA 를 이용하는 디렉토리 서비스 인터페이스는 ORB 에 기반하여 서버에 존재하는 구현객체의 오퍼레이션으로 동작하므로 기존의 디렉토리 표준에서 정의한 프로토콜에 비해 정의와 구현에 있어서 매우 쉽고 간단하게 구현할 수 있는 장점이 있고, 객체지향적인 프레임워크에 따라 재사용성과 시스템 변화에 대처할 수 있는 능력이 뛰어나다고 할 수 있다.

본 논문에서 설계한 디렉토리 모델은 X.500 과 LDAP 의 디렉토리 정보모델에 기초하고 있으나 객체지향 모델에 입각한 모델링 방법을 사용하고 있으므로 전체적인 디렉토리 시스템이 객체지향적인 개발 방법론에 입각하여 구현될 수 있을 것이다.

본 연구를 기반으로 한 향후 연구 과제로는 시스템 설계를 바탕으로 구현이 이루어져야 하고, 디렉토리 서버 간의 상호작용과 Referral 구현 및 다른 디렉토리 시스템과의 상호 연동 등의 기능을 추가하여 분산 디렉토리 시스템으로의 발전이 필요하다.

VI. 참고 문헌

[1]진성일, 이규철, 김명호 외, "통합자료관리체계 시험구현연구", 1997.12, 국방과학연구소

[2]진성일, 이규철, 김명호 외, "분산이종 DB 연동기술연구", 1998.12 국방과학연구소

[3]Randy Otte, Paul Patrick , Mark Roy, Understanding CORBA.

[4]이재호, "통신망 환경하에서 객체-능동-지식 기반 디렉토리 데이터베이스 모델 설계", 홍익대학교, 1996

[5]김경미, 이재호, 김경창, 임해철, "객체지향 X.500 DIB 구축에 관한 연구", '93 동계 데이터베이스 학술대회

[6]신석철, 이재호, 임해철, "데이타베이스 관리 시스템을 이용한 X.500 DIB 구축에 관한 연구", '94 한국정보과학회 추계학술발표회

[7]T. Howes , S. Kille , "Lightweight Directory Access Protocol v3", RFC 2251, December 1997

[8]A. Coulbeck, T. Howes , S. Kille, "Lightweight Directory Access Protocol (v3):Attribute Syntax Definitions", RFC 2252, December 1997

저자 약력 양식

논문/발표 제목 : 통합데이터베이스 환경에서 CORBA 기반의 디렉토리 서비스 설계

발표도구 : Beam Projector (0) OHP ( ) 기타

성명 : 박재형 (공동저자 : 김기봉, 진성일)
직장명 및 부서 : 충남대학교 컴퓨터과학과 직위 : 대학원생
직장주소 : 대전광역시 유성구 궁동 220 번지 충남대학교 자연과학대학
컴퓨터과학과 데이터베이스연구실 (우편번호 : 305-764)
전화 : (042) 821-7453 FAX : (042) 823-9959
Email : jhpark@cs.chungnam.ac.kr