LDAP을 이용한 학생정보시스템 구현

윤성중, 김건웅

목포해양대학교 해양전자·통신공학부

Implementation of Student-Information System by LDAP

Yoon Sung-Jung, Geonung Kim

Division of Marine Electronic & Communication Engineering, Mokpo National Maritime University

e-mail: {zeroyoon, kgu}@mmu.ac.kr

요 약

본 논문에서는 디렉토리 서비스를 이용하여 학생정보시스템을 구현한 결과를 소개한다. 먼저 필요한 속성들을 추가로 선언하였으며, 그것들을 바탕으로 새로 객체 클래스 "myStudent"를 정의하였다. LDAP 서버는 OpenLDAP 2.x를, PHP는 4.3.x를 이용하였다.

ABSTRACT

In this paper, we introduce an implementation of Student-Information System using LDAP(lightweight directory access protocol) service. We defined new attributes for student-information, and then defined new objectClass "myStudent". We implemented the system on the OpenLDAP 2.x and PHP 4.3.x.

키 워 드

LDAP, PHP, Schema, objectClass

Ⅰ. 서 론

네트워크가 IT산업의 제 1의 기본 구조로 등장하면서 네트워크를 통한 정보 처리가 급속하게 증가하고 있다. 이에 따라 네트워크에서의 정보 관리의 중요성이 이슈로 대두되고 있다.

사람, 응용프로그램, 자원 등과 같은 다양한 정보는 대부분의 IT 기업을 통해 분산되며 계속 확산되고 있다. 이를 사용하고 지원하는 응용 프로그램과 플랫폼의 수가 증가함에 따라, 여러 곳에 있는 정보를 관리해야 한다. 응용 프로그램과 운영체제 기능에는 영향을 주지 않으면서 비용을 최소화하고 변화에 대응하는 능력을 키우기 위해 정보의 저장, 액세스, 관리할 공통 장소를 제공하는 별도의 서비스가 필요하게 되었다. 이러한 서비스가 디렉토리(directory) 서비스이다[1].

본 논문에서는 이러한 디렉토리 서비스와 이를 인터넷에 적용하기 위해 제안된 LDAP(Lightweight Directory Access Protocol :경량의 디렉토리 접근 프로토콜)의 이점을 알아보고, 이를 이용하여 구축한 학생정보시스템을 소개한다. 학생정보시스템을 구축하기 위하여 LDAP 스키마(schema)를 새로 작성하였으며, PHP를 이용하여 정보를 등록, 탐색, 수정, 삭제하는 기능을 구현하였다.

본 논문의 구성은 다음과 같다. 먼저 2장에서는 디렉토리 서비스와 LDAP의 장점을 정리하고, 3장에서는 학생정보시스템을 위한 스키마 작성 과정을 설명하며, 다음 4장에서는 PHP를 이용한 학생정보시스템 구현 과정과 결과를 소개하고, 마지막으로 5장에서 결론을 맺는다.

Ⅱ. 디렉토리 서비스의 이점

1980년대 말에 특정분야의 디렉토리 서비스의 이용, 개발 요구가 높아감에 따라 CCITT(International Telegraph and Telephone Consultative Committee)와 ISO(International Organization for Standardization) 두 단체가 함께 X.500이라는 디렉토리 서비스 표준을 만들기 시작했다. 결국 1990년에 CCITT가 표준을 발표했고 1993년, 1997년 몇 번의 수정작업을 거쳐 현재에 이르렀다. 이 X.500은 최초의 일반적인 목적의 디렉토리 시스

템이었고 다양한 쿼리(query)를 사용하는 강력한 검색기능을 제공하였을 뿐만 아니라, 서버와 데이터의 분산이 용이하고, 무엇보다도 특정 운영체제나 특정 네트워크, 특정 응용프로그램에 구애받지 않고 사용될 수 있는 표준이라는 점이 특징이다.

디렉토리 서비스는 각 시스템을 통합, 조직화, 통합된 관리를 할 수 있다. 각각의 시스템은 서로 연동하여 비즈니스 목적에 맞는 정보 공유, 통합 서비스 채널을 사용할 수 있으며, 사용자, 조직, 비즈니스의 필요성을 만족하는 네트워크로 통합된다. 네트워크는 각각 시스템을 바탕으로 단일 관리 서비스를 제공하며, 네트워크 자원들 간의 관계가 설정된다. 그리고 고객, 비즈니스, 인트라넷, 인터넷 사이를 유지 관리한다. 네트워크는 비즈니스에 존재하는 모든 관계를 구현할 수 있다.

그러나 X.500의 DAP(Directory Access Protocol) 규격이 방대하고 복잡하여 구현이 어렵고 시간이 많이 소요되고, 인터넷 환경에 비교적 부적합하고, 구축하는데 비용이 많이 필요하다는 단점이 있다. 이러한 문제들을 해결하기 위해 제안된 것이 LDAP이다.

LDAP는 DAP의 주요 기능은 대부분 지원하면서 복잡했던 부분이나 잘 쓰이지 않았던 부분을 단순화하거나 없애버렸다. 그리고 대부분의 데이터 형식에 있어서 단순한 문자열을 사용하여 구현을 단순화하고 성능을 향상시켰다.

III. 스키마 생성

사용자가 LDAP에서 기본적으로 제공되는 Schema 디렉토리 이하에 있는 파일들의 객체 클래스(ObjectClass)들을 그대로 사용한다면 편리하지만, 필요로 하는 조건에 맞는 객체 클래스가 없는 경우, 용도에 맞는 스키마를 추가로 작성하여 LDAP 서버를 이용할 수 있다[2][3].

본 연구에서는 LDAPv3를 지원하는 OpenLDAP 2.x를 이용하여 구현하였는데, 여기서는 openldap/etc/openldap/schema 디렉토리에 있는 core.schema 파일을 기본으로 사용한다[4][5].

OpenLDAP 2.x에서 새로운 속성(attribute)을 작성하기 위해서 attributeType 이라는 지시자를 사용하여 새로운 속성을 정의한다.

```
AttributeTypeDescription = "(" whsp
        numericoid whsp
        [ "NAME" qdescrs ]
        [ "DESC" qdstring ]
        [ "OBSOLETE" whsp ]
        [ "SUP" woid ]
        [ "EQUALITY" woid
        ["ORDERING" woid
        [ "SUBSTR" woid ]
        [ "SYNTAX" whsp noidlen whsp ]
        [ "SINGLE  VALUE" whsp ]
        [ "COLLECTIVE" whsp ]
        [ "NO  USER  MODIFICATION" whsp ]
        [ "USAGE" whsp AttributeUsage ]
        whsp ")"
```

위의 설명에서 whsp란 공백(' ')문자를 의미하고 qdescrs란 하나 또는 하나 이상의 이름이 위치한다는 뜻이다. qdescrs의 예를 들면 'cn'처럼 하나를 쓸 수도 있고, 두 개 이상의 이름을 위치시킬 필요가 있을 때는 '$'문자로 구분을 두고 쓸 수가 있다. 그리고 woid란 이름 또는 OID(Object IDentifier)가 위치할 수 있다. 예를 들어 'cn'의 OID가 2.5.4.3이면 'cn'을 쓰거나 2.5.4.3을 쓸 수가 있다.

numericoid는 새로운 속성에 부여할 OID이다. LDAP에서는 서로 다른 LDAP 객체들이 서로 구분될 수 있도록 모든 요소에 OID를 부여한다. 그리고, 자신에게 또는 그룹에게 할당받은 OID 아래로 계속 확장하여 쓸 수 있다.

[NAME" qdescrs]는 새로 정의할 속성의 이름이고, ["DESC" qdstring]는 속성에 대한 설명이다. ["SUP" woid]는 woid에 해당하는 속성으로부터 상속을 받는다.

["SYNTAX" whsp noidlen whsp]는 속성의 데이터 형을 정의한다. ["SINGLE VALUE" whsp]는 속성 값이 하나만 가지게 된다. 이 지시자를 생략하면 기본적으로 Multi-Value가 된다. 즉 하나이상의 값을 속성이 가질 수 있다.

["NO USER MODIFICATION" whsp]는 유저가 이 속성 값을 수정할 수 없도록 설정한다. 이 지시자를 생략하면 속성 값이 수정 가능하다.

앞에서 설명한 규칙에 따라 학생들의 이름, 학번, 학년, 전화번호, E-Mail, 주소, 학생 사진 속성들을 아래와 같이 새로이 정의 한다.

```
attributetype ( 1.1.2.1.1 NAME 'SchoolNum'
        EQUALITY integerMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.27
        SINGLE-VALUE)

attributetype ( 1.1.2.1.2 NAME 'SchoolYear'
        EQUALITY integerMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 )

attributetype ( 1.1.2.1.4 NAME 'myPhoto'
        DESC 'RFC2798 : a JPEG image;
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.28 )

attributetype ( 1.1.2.1.5 NAME 'myEmail'
        EQUALITY octetStringMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.40 )

attributetype ( 1.1.2.1.6 NAME 'HomeAddress'
        EQUALITY octetStringMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.40 )
```

속성들을 작성 후 이를 바탕으로 아래와 같은 새로운 객체 클래스를 만들었다.

```
objectclass ( 1.1.2.2.1 NAME 'myStudent'
        SUP person
        MUST (SchoolNum $ SchoolYear )
        MAY ( myPhoto $myEmail $ HomeAddress)  )
```

새로 만든 객체클래스의 OID는 1.1.2.2.1이고, 이름은 "myStudent"이다. SUP 항은 person 객체 클래스를 상속했음을 나타내고 있고, MUST 는 반드시 있어야 하는 속성들을 뜻하는데, 여기서는 SchoolNum과 SchoolYear 속성을 지정했다. MAY는 선택 입력 속성항목이다. 또한 '$' 문자로 속성들을 구분한다.

Ⅳ. 시스템 구현

openldap 2.X 는 LDAPv3를 기본으로 지원한다. 그러나 PHP 4.3에서는 LDAPv2를 기본으로 지원하고 있다. 따라서 PHP를 이용하는 경우 반드시 LDAP 버전을 설정해 주어야 한다. 버전이 다른 경우 LDAP 서버와 연결 후 바인딩이 되지 않는 오류가 발생한다. 그래서 ldap_set_option() 함수를 다음과 같이 사용하여 LDAP 버전을 v3로 설정했다[6].

```
ldap_set_option($ds,LDAP_OPT_PROTOCOL_VERSION,3);
```

LDAP 서버에 새로운 엔트리(entry)를 추가하는 함수는 ldap_add( )인데, 'dn' 인자로 지정된 DN(distinguished name)에 엔트리 인자로 지정된 항목을 추가한다. 엔트리 인자에는 추가할 엔트리 속성들의 값을 담은 배열을 넣어야 한다. 하나의 속성이 여러 개의 값을 가지는 경우에는 해당 요소에 값들을 넣은 배열을 넣어야 한다. 엔트리 추가에 실패하면 'false(0)'값을 돌려준다.

다음은 엔트리를 추가하는 예이다.

```
$dn = "sn=soung,o=Mokpo,dc=mmu,dc=com";
$info["objectClass"]="mystudent";
$info["sn"]="soungjung";
$info["cn"]="Yoon";
$info["SchoolYear"]="2";
.....
$result = ldap_add($ds,$dn,$info);
```

아래와 같이 LDAP 서버에서 엔트리를 삭제하는 함수는 'dn' 인자로 지정된 엔트리를 삭제한다. 엔트리 삭제에 실패하면 'false(0)'을 되돌려준다.

```
$dn = "sn=soung,o=Mokpo,dc=mmu,dc=com";
$result = ldap_delete($ds,$dn,$info);
```

아래와 같이 LDAP 서버에 엔트리 검색은 'dn' 인자로 지정된 디렉토리와 모든 하위 디렉토리들에 대해 검색을 수행한다. 'searchFilter' 인자는 생략할 수 있으며, 'searchFilter' 인자를 지정하면 여기에 지정된 특성들을 갖는 엔트리들만 결과에 포함된다.

```
$searchFilter= "sn=*";
$ds=ldap_connect("localhost");
$r=ldap_bind($ds,"cn=Manager,dc=mmu,dc=eci,dc=net",
                                        "secret");
$result=ldap_search($ds,"dc=mmu,dc=com",$searchFilter);
```

검색 결과에 속한 엔트리들에 대한 정보는 ldap_get_entries( ) 함수의 'result' 인자로 지정된 결과 세트에 담긴 모든 엔트리들을 담은 3차원 배열을 돌려준다. 색인이 'count'인 요소에는 배열이 담긴 엔트리 개수가 들어 있다. 각 항목은 0번부터 시작된다. 각 항목 요소(2차원 배열)마다 'count' 요소와 'dn' 요소가 하나씩 들어 있다. 항목의 각 속성들은 숫자 색인으로 참조 할 수도 있고 속성 이름으로 참조할 수도 있다. 각 속성 요소 (3차원 배열)에도 'count' 요소가 하나씩 들어 있으며 특성 값들은 숫자 색인으로 참조한다.

```
$info = ldap_get_entries($ds,$result);
echo "Data for " . $info["count"] . " items returned:<p>";
for ($i=0; $i<$info["count"]; $i++) {
        echo "dn is: " .$info[$i]["dn"]."<br/><hr/>";
        echo "cn is: " .$info[$i]["cn"][0]."<br/><hr/>";
            .....
        echo " E - MAIL is: "
                    .$info[$i]["myemail"][0]."<br/><hr/>";
}// 검색 결과를 화면에 나타낸다.
```

PHP에서 파일을 업로드를 하기 위해서는 PHP 설정 파일인 php.ini에서 file_uploads, upload_max_filesize, upload_tmp_dir, post_max_size 값을 설정해 주어야 한다. 학생 사진 파일을 업로드하기 위해 PHP에서 설정한 디렉토리에 임시로 파일을 저장한 후, 저장된 파일을 fread() 함수를 사용하여 binary 값으로 읽어 들여서 LDAP 서버에 저장하였다.

```
$uploadfile=$_FILES['myphoto']['tmp_name'];
$fp=fopen($uploadfile,"r");
$st=fread($fp,$_FILES['myphoto']['size']);
fclose($fp);
```

사진을 가져와서 보여주는 함수에서는 LDAP 서버에 저장된 사진파일이 binary 값으로 구성되어 있으므로 ldap_get_values_len( )를 사용하였다. 불러온 사진 파일은 ./nobody 폴더에 임시적으로 이름을 부여하여 저장한 후 화면에 출력하였다.

```
if(ldap_count_entries($ds,$result)>0){

$ldapResults=ldap_get_entries($ds,$result);

for($en=ldap_first_entry($ds,$result);$en!=NULL;
                        $en=ldap_next_entry($ds,$en)){
        $ldapBinary=ldap_get_values_len($ds,$en,"myphoto");
        $uploadfiledir="./nobody/".$j.".JPG";
        $fp=fopen($uploadfiledir,"w");

        for ($i = 0; $ldapBinary[$i] != NULL; $i++) {
```

```
                fwrite($fp,$ldapBinary[$i]);
            }

            fclose($fp);
            $j++;
        }
    }
```

다음 그림은 LDAP 학생 정보를 입력하여 추가하는 화면이다.

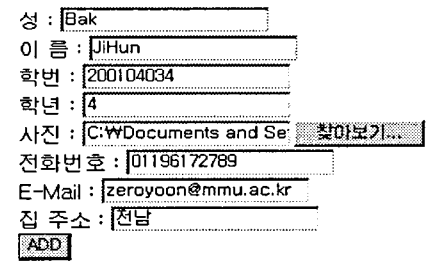

그림 1. 학생 정보 추가

다음 그림은 학생정보를 검색하는 화면이다.

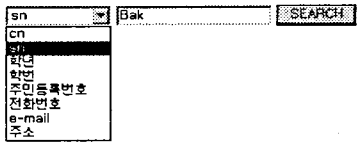

그림 2 학생 정보 검색

다음은 학생 정보를 검색한 결과를 표로 나타낸 화면이다.

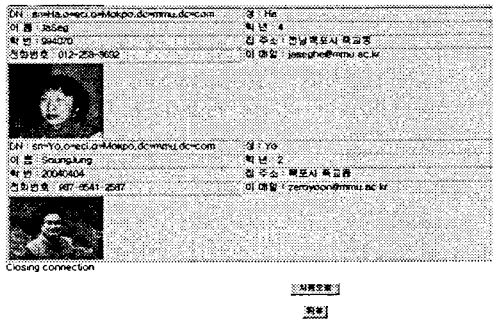

그림 3 학생 정보 검색 결과

V. 결론

본 논문에서는 LDAP를 이용하여 구축한 학생 정보시스템을 소개한다. 학생정보시스템을 구축하기 위하여 필요한 속성을 새로 정의하고, 새로 객체 클래스를 정의하는 과정을 보였으며, PHP를 이용하여 학생 정보를 등록, 탐색, 수정, 삭제하는 기능을 구현한 결과를 보였다. 특히 학생들의 사진과 같은 바이너리 정보를 등록하고, 탐색하는 기능을 구현하였다.

본 연구를 통해 얻어진 OpenLDAP과 PHP4.3에 대한 지식과 경험을 바탕으로, 학교전산망 자원 관리시스템, 보안키 관리시스템 등의 응용 시스템을 계속 구축할 예정이다.

참고문헌

[1]류광택, "디렉토리서비스 기술동향 연구", 한국전산원

[2]"Lightweight Directory Access Protocol (v3)", RFC 2251, December, 1997

[3]"Lightweight Directory Access Protocol (v3): Attribute Syntax Definitions", RFC 2252, December, 1997

[4]http://www.openldap.org/

[5] "Definition of the inetOrgPerson LDAP Object Class", RFC 2798, April 2000

[6]http://www.php.net/manual/kr/ref.ldap.php