윈도우 툴바 기반의 ENUM Client 구현 연구

A Study on the Implementation of ENUM Client Based on Window Tool-bar

강민구
한신대학교 정보통신학과

Min-Goo Kang(kangmg@hs.ac.kr)

요약

본 논문에서는 IETF에 의해 정의된 표준 프로토콜인 ENUM(tElephone NUmber Mapping)의 E.164 전화번호(현재 ITU 주관으로 사용하는 국제전화번호체계)를 정보식별자로 인터넷 도메인네임시스템(DNS)에 수용함으로서 인터넷 접속과 이메일 전송, 팩스 전송 및 인터넷 전화 등이 가능한 ENUM 클리언트와 ENUM API를 설계하고 ENUM서비스를 구현하였다.

■ 중심어 : | ENUM 서버 | ENUM DNS | ENUM API | E.164 | e164.arpa |

Abstract

In this paper, the plug-in scheme for ENUM client and ENUM services was proposed using an ENUM DNS applied by E.164 telephone numbering. Especially, ENUM services(ENUM HTTP, ENUM FTP) were implemented according to ENUM root domain(e164.arpa).

■ keyword : | ENUM client | ENUM DNS | ENUM API | E.164 | e164.arpa |

I. 서 론

새로운 식별체계인 ENUM의 활용으로 ENUM API 정의와 표준이 필요하다. 클라이언트에서 입력된 ENUM 주소를 ENUM DNS로 전달이 되고, ENUM DNS는 다시 필요한 서비스에 맞추어서 다시 변환하여 클라이언트로 보내져야 한다.

즉, ENUM 주소를 입력하면 웹 클라이언트에서는 사용자가 원하는 서비스를 식별하여 해당하는 ENUM 서비스를 지원한다. 본 논문에서는 Plug-in을 이용한 Internet Explorer 주소창에 전화번호를 입력하고, ENUM 클라이언트를 구축함으로서 전화망의 주소체계와 인터넷의 주소체계를 Mapping하는 ENUM 서비스 방식이 가능하도록 ENUM 서버를 설계한다[1].

II. ENUM 시스템 설계

1. 착신인터넷용 ENUM시스템 연구

ENUM은 공중전화교환망(PSTN)과 인터넷프로토콜(IP) 네트워크를 결합하는 변환프로토콜로서 E.164 전화번호만으로 홈페이지, 이메일, 팩스, 전화 등의 서비스에 접근 가능하게 하는 통합적인 성격의 새로운 인터넷 주소체계이다.

* 이 논문은 2004년도 한신대학교 학술연구비 지원에 의하여 연구되었음
접수번호 : #041020-004
접수일자 : 2004년 10월 20일
심사완료일 : 2004년 12월 01일
교신저자 : 강민구, e-mail : kangmg@hs.ac.kr

E.164 전화번호를 통해 인터넷에 접속하기 위해서는 ITU와 IETF의 협의에 의해 결정된 ENUM root domain인 e164.arpa를 활용한다. 그림 1은 인터넷의 착신기능을 부가하기 위한 ENUM 클라언트 위치와 서비스의 구성도이다[2].

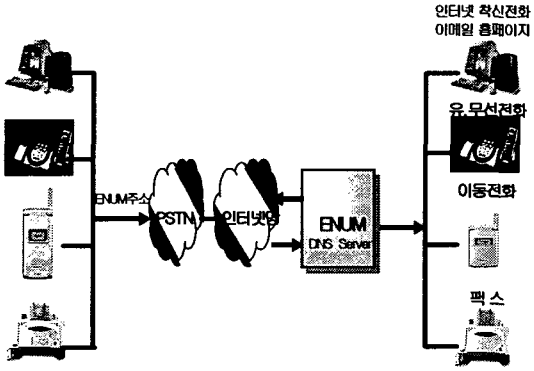

그림 1. 인터넷 착신기능용 ENUM 서비스구성도

1.1 PSTN에서 IP기반 네트워크로의 호출

그림 2는 인터넷 착신기능을 위한 PSTN 사용자가 IP망의 인터넷폰 착신위한 호 처리도이다[2].

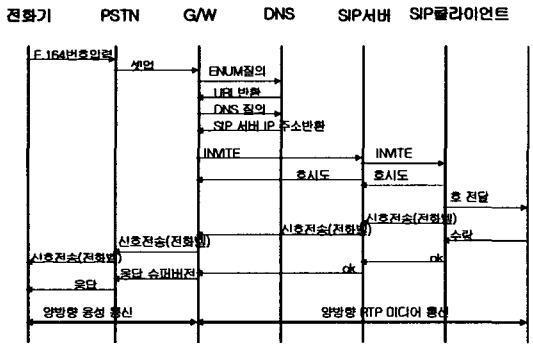

그림 2. PSTN사용자가 IP망의 인터넷폰 착신위한 호 처리 다이어그램

1.2 IP 기반 네트워크에서 PSTN으로의 호출

그림 3은 인터넷발신을 위해 IP망의 인터넷폰 사용자가 PSTN 착신위한 키폰접속용 호 처리도이다.

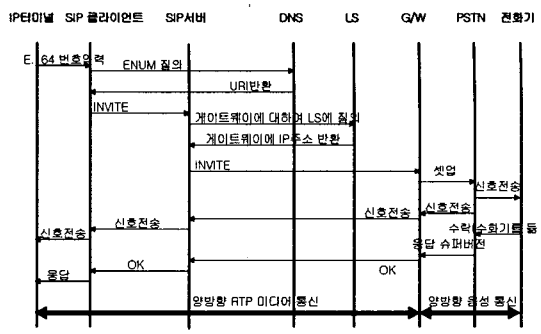

그림 3. IP망의 인터넷폰 사용자가 PSTN 착신위한 호 처리 다이어그램

2. ENUM시스템용 API 설계

본 논문에서는 최근 ENUM-Forum, IETF, ITU-T, NCA, PEC-ETRI, ENUM Reference Material, KRNIC 등의 기술동향에 대한 분석을 통하여 ENUM API의 연구를 통해 ENUM 서비스를 원활히 할 수 있도록 클라이언트에서 필요한 Web 클라이언트를 새롭게 정의한다.

2.1 클라이언트용 ENUM API설계 및 서비스모델

ENUM API를 정의하기 위한 웹 클라이언트에서의 서비스 모델의 구조는 그림 4와 같다.

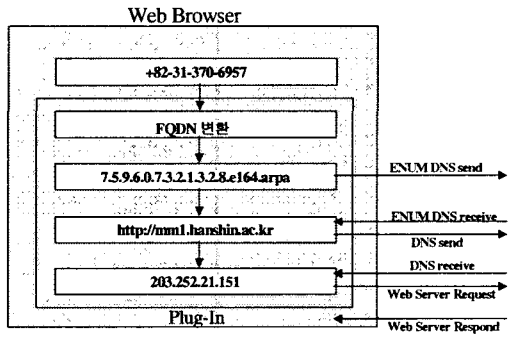

그림 4. 클라이언트접속용 Plug-in S/W

웹 클라이언트로는 웹 브라우저를 예제로 선택하였다. 웹 브라우저에는 사용자가 브라우저에 입력하는 주소를 plug-in S/W를 둔다.

Plug-in S/W에서 ENUM 주소가 인식되면 ENUM을 인터넷 주소체계로 바꾸어 DNS 클라이언트에 웹 서

비스를 위한 ENUM API를 사용하여 해당 서비스를 위한 정보를 요구한다. DNS 클라이언트는 해당 정보를 가져오기 위해 ENUM DNS에 request를 보낸다.

ENUM DNS는 request에 대한 response로 ENUM 주소에 대한 DNS에 저장되어 있는 NAPTR Record를 DNS클라이언트에 보내준다.

DNS 클라이언트는 웹 서비스를 위해 Plug-in S/W가 호출한 ENUM API에 따라 해당하는 정보를 Plug-in S/W에 보내주게 되며, 가져온 정보를 가지고 Plug-in S/W는 웹 서비스를 제공하게 된다[6].

2.2 ENUM 클라이언트 Program

그림 5는 ENUM ToolBar 생성을 위해 COM 인터페이스 개체 연결부와 IE에 Plug-in을 위한 밴드의 정보를 노출하는 인터페이스 선언부이다.

```
BEGIN_COM_MAP(CStockBar)
    COM_INTERFACE_ENTRY(IStockBar)
    COM_INTERFACE_ENTRY(IInputObject)
    COM_INTERFACE_ENTRY(IOleWindow)
    COM_INTERFACE_ENTRY_IID(IID_IDockingWindow, IDockingWindow)
    COM_INTERFACE_ENTRY(IObjectWithSite)
    COM_INTERFACE_ENTRY_IID(IID_IDeskBand, IDeskBand)
    COM_INTERFACE_ENTRY(IDispatch)
END_COM_MAP()

// IDeskBand
public:
    STDMETHOD(GetBandInfo)(DWORD dwBandID, DWORD dwViewMode, DESKBANDINFO* pdbi);

// IObjectWithSite
public:
    STDMETHOD(SetSite)(IUnknown* pUnkSite);
    STDMETHOD(GetSite)(REFIID riid, void **ppvSite);

// IOleWindow
public:
    STDMETHOD(GetWindow)(HWND* phwnd);
    STDMETHOD(ContextSensitiveHelp)(BOOL fEnterMode);

// IDockingWindow
public:
    STDMETHOD(CloseDW)(unsigned long dwReserved);
    STDMETHOD(ResizeBorderDW)(const RECT* prcBorder, IUnknown* punkToolbarSite, BOOL fReserved);
    STDMETHOD(ShowDW)(BOOL fShow);

// IInputObject
public:
    STDMETHOD(HasFocusIO)(void);
    STDMETHOD(TranslateAcceleratorIO)(LPMSG lpMsg);
    STDMETHOD(UIActivateIO)(BOOL fActivate, LPMSG lpMsg);

// IStockBar
public:
    void FocusChange(BOOL bHaveFocus);
    STDMETHOD(Invoke)(DISPID dispidMember,REFIID riid, LCID lcid, WORD wFlags,
                DISPPARAMS * pDispParams, VARIANT * pvarResult,
                EXCEPINFO * pexcepinfo, UINT * puArgErr);
```

```
protected:
    BOOL RegisterAndCreateWindow();
protected:
    DWORD m_dwBandID;
    DWORD m_dwViewMode;
    BOOL m_bShow;
    BOOL m_bEnterHelpMode;
    HWND m_hWndParent;
    CReflectionWnd m_ReflectWnd;
    IInputObjectSite* m_pSite;

    CComQIPtr<IWebBrowser2, &IID_IWebBrowser2> m_spWebBrowser2;

    DWORD m_dwCookie;  // Connection Token - used for Advise and Unadvise

    enum ConnectType { Advise, Unadvise };  // What to do when managing the connection

    BOOL ManageConnection(enum ConnectType eConnectType);
```

그림 5. dll registry 등록

IobjectWithSite의 인터페이스 메서드는 IE가 Embedding할 경우 제일 먼저 Query되는 인터페이스이다. IoleWindow는 IE가 밴드에게 윈도우 핸들을 요구할 때 사용해야 하는 인터페이스이다. 여기서는 웹브라우저 컨트롤을 생성시켰으므로 웹브라우저 컨트롤에 대한 핸들을 돌려주면 된다.

IdockingWindow는 세 가지 메소드를 포함하고 있다. 각각의 메소드는 IE가 밴드 오브젝트를 보여줄 때나 종료 시킬 때, 혹은 밴드 오브젝트의 사이즈를 조정할 때 호출하게 된다. IdeskBand는 GetBandInfo 메소드 하나만을 가지고 있지만 이 부분에서 밴드가 보여지는 속성의 대부분을 지정해 줄 수 있으므로 밴드 UI 표현에 있어서 매우 중요한 부분이다[3].

2.3 Windows registry 등록

Windows Registry는 윈도우 운영 체제의 핵심적인 요소로서, 윈도우 운영 체제와 응용 프로그램들이 필요한 여러 가지 정보를 담고 있다.

레지스트리는 윈도우 디렉토리에서 SYSTEM.DAT, USER.DAT, 이 두개의 파일에 저장된다.

SYSTEM.DAT는 윈도우 운영체제의 핵심적인 정보와 하드웨어에 관한 정보를 담고 있다. USER.DAT는 사용자들에 대한 정보를 담고 있다.

그림 6. 레지스트리 등록

레지스트리 등록은 보통 두 가지 방법이 있다. 하나는 윈도우 레지스트리를 보거나 편집하려면 레지스트리 편집기를 실행하여야 한다.

실행 창에다가 REGEDIT.EXE를 입력하면 레지스트리 편집기가 뜬다. 두 번째 등록방법은 명령프롬프트 창에서 자동으로 레지스트리를 등록 시키는 방법으로 DLL 파일을 REGSVR32으로 입력한다. 그림 6은 두 번째 방법으로 직접 등록시키는 모습이다[6].

2.4 ENUM ToolBar 생성1) ENUM 주소 입력

클라이언트에서 자동으로 registry를 등록시킨 후 Internet Explorer를 실행하면 그림 7과 같은 모습으로 ENUM 번호만을 넣을 수 있는 ToolBar가 자동으로 생성될 것이다. Plug-in의 주된 역할은 IE의 이벤트를 가로 채기 위함이다.

ENUM ToolBar 주소창에 입력되는 주소값을 서버에 request 하기 위해 ENUM FQDN(fully qualified domain name) 변환 절차를 거쳐 ENUM 주소로 변환해준다.

3. ENUM 프로토콜 변환

그림 8은 Plug-in 프로그램에서 ENUM 주소가 입력되었는지 확인하는 함수이다.

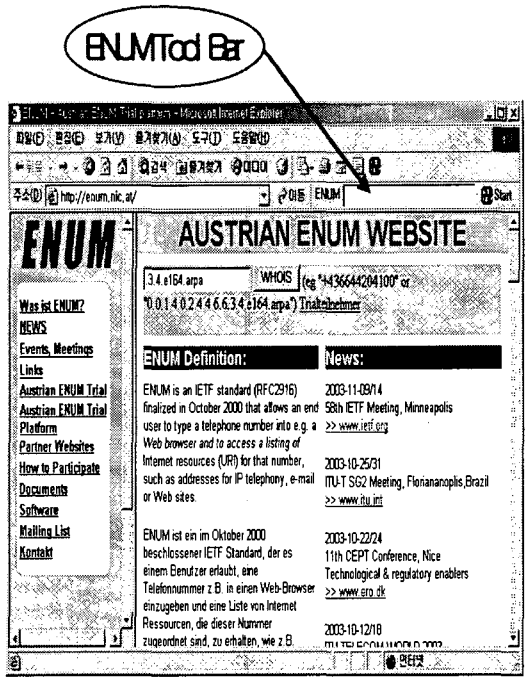

그림 7. ENUM Tool Bar

```
if (m_pBrowser)
{
    VARIANT vEmpty;
    VariantInit(&vEmpty);
    m_pBrowser->Stop();
    _bstr_t bsSite1;
    length = m_EditWnd.GetWindowTextLength();

    if (length)
    {
        OLECHAR* EnumProtocol = NULL;
        //ENUM 주소인지 확인하는 부분
        m_EditWnd.GetWindowText(&EnumProtocol);
        if(EnumProtocol[0] >= '0' && EnumProtocol[0] <='9')
        {
            ENUMConvert(length, EnumProtocol);
        }
        else
        {
            m_EnumDlg.DestroyWindow();
            bsSite = EnumProtocol;
        }
    }
```

그림 8. ENUM 주소 확인

그림 9는 Plug-in 프로그램에서 ENUM URI를 ENUM FQDN 변환 절차에 맞추어서 ENUM 프로토콜변환 하는 함수이다.

```
void CMFToolbar::ENUMConvert(int length, OLECHAR EnumProtocol[])
{
    for(int i=0;i<256;i++)
    {
        EnumPT[i]='\0';
    }

    m_EnumDlg.DestroyWindow();

    for(int i=0; i<256; i++)
        EnumProtocol1[i] = '\0';

    int k=0;

    for(int j= (length-1); j>=0; j--)
    {
        EnumProtocol1[k] = EnumProtocol[j];
        EnumProtocol1[k+1] = '.';
        k=k+2;
    }

    EnumProtocol1[k] = 'e';
    EnumProtocol1[k+1] = '.';
    EnumProtocol1[k+2] = '1';
    EnumProtocol1[k+3] = '6';
    EnumProtocol1[k+4] = '4';
    EnumProtocol1[k+5] = '.';
    EnumProtocol1[k+6] = 'a';
    EnumProtocol1[k+7] = 'r';
    EnumProtocol1[k+8] = 'p';
    EnumProtocol1[k+9] = 'a';
    EnumProtocol1[k+10] = '.';
    EnumProtocol1[k+11] = '\0';

    MessageBox (EnumProtocol1,_T("ENUM 주소"),MB_OK);

    NAPTRlength = (length+2)+12;
```

그림 9. ENUM Protocol 변환

3.1 ENUM 주소 입력

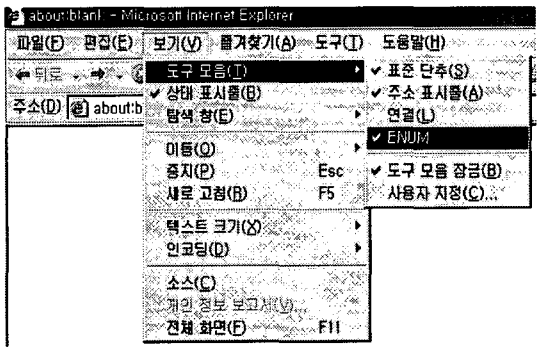

그림 10. ENUM Tool Bar 실행

3.2 ENUM Protocol로 변환된 주소

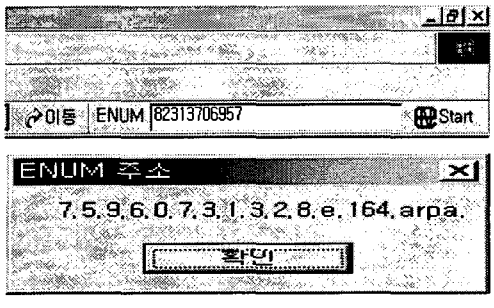

그림 11. ENUM Protocol FQDN

Plug-in이란 원래의 프로그램에서 제공하지 못하는 기능을 원본 프로그램을 수정하지 않고 그 프로그램에 필요한 기능을 추가하는 것을 말한다. 현재 사용하는 Web Browser에서는 ENUM 서비스에 맞는 프로토콜로 ENUM 서버에 전달할 수가 없다.

이것을 해결하기 위해서 Web Browser 안에 Plug-in된 ENUM 주소창에 EUNM URI가 입력되면 그것을 확인하여 Plug-in 프로그램에서 ENUM URI를 그림 11과 같이 ENUM FQDN 변환 절차에 맞추어서 변환한다. 다음은 ENUM FQDN 변환 절차이다.

1. 국가 코드 IDDD를 포함한 완전한 형태의 E.164번호를 찾는다. 예: 82313706957
2. 각 숫자들 사이에 점(".")을 찍는다.
   예: 8.2.3.1.3.7.0.6.9.5.7
3. 숫자의 순서를 뒤집는다.
   예: 7.5.9.6.0.7.3.1.3.2.8
4. 끝에 "e.164.arpa."를 붙인다.
   예: 7.5.9.6.0.7.3.1.3.2.8.e164.arpa

위의 순서와 같이 ENUM 프로토콜에 맞추어 변환한 후 ENUM RR Packet Format에 맞추어 ENUM 서버에 보낸다[6].

3.3 ENUM SEND 함수

그림 12는 Plug-in S/W에서 웹 브라우저에서 입력된 주소가 ENUM FQDN을 ENUM DNS 서버에 전송하는 함수이다.

```
void CMFToolbar::SEND()
{
    m_EnumDlg.DestroyWindow();
    int sock;
    int echoStringLen;
    unsigned short echoServPort=4000;
    char *servIP="203.252.21.221";
    char echoBuffer[ECHOMAX+1];
    struct sockaddr_in echoServAddr;
    sock = socket(PF_INET, SOCK_STREAM, IPPROTO_TCP);
    memset(&echoServAddr, 0, sizeof(echoServAddr));
    echoServAddr.sin_family = AF_INET;
    echoServAddr.sin_addr.s_addr = inet_addr(servIP);
    echoServAddr.sin_port = htons(echoServPort);
    connect(sock, (struct sockaddr*)&echoServAddr, sizeof(echoServAddr));
```

```
    if(EnumPT[0]=='U')
        echoStringLen = (length*2)+17;
    else if(EnumPT[0]=='F')
        echoStringLen = (length*2)+16;
    else
        echoStringLen = (length*2)+12;//strlen(EnumPT);

    send(sock, EnumPT, echoStringLen, 0);
    for(int n=0; n<256; n++)
    {
        echoBuffer[n]='\0';
    }
    recv(sock, echoBuffer, ECHOMAX+1, 0);
    if(echoBuffer[0]=='\0')
    {
        MessageBox(_T("등록되지 않은 번호입니다.\n다시 입력해 주십시요!"),_T("Error"),MB_OK);
    }
    Connect(echoBuffer);
    closesocket(sock);
}
```

그림 12. ENUM FQDN 전송함수

3.4 ENUM Dialog Service

그림 13은 클라이언트 쪽에서는 Dialog Menu가 실행이 되면 사용하고자 하는 서비스(HTTP, FTP, EMAIL, SIP) 중에 하나를 선택하여 check 해주게 되면 해당 서비스를 실행하게 된다.

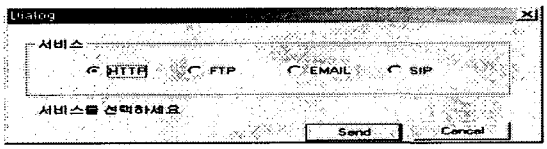

그림 13. Dialog Menu

만약 HTTP를 선택하면 처음 부분에 HTTP:를 붙여줘서 어떤 서비스를 선택 했는지를 구분 할 수 있다. HTTP:7.5.9.6.0.7.3.1.3.2.8.e164.arpa와 같은 방법으로 FQDN을 변환한다.

3.5 ENUM HTTP 서비스 구축

그림 14는 클라이언트 측에서 HTTP 서비스를 선택하여 그 결과를 보여주는 화면이다.

3.6 ENUM FTP 서비스 구축

그림 15는 클라이언트 측에서 FTP 서비스를 선택하여 그 결과를 보여주는 화면이다.

그림 14. HTTP 서비스 구축화면

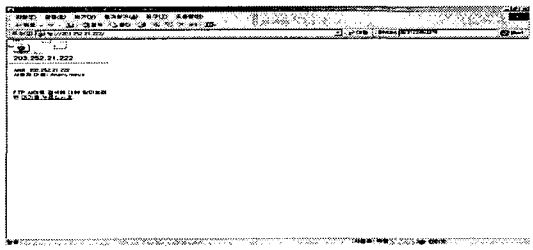

그림 15. FTP서비스 구축화면

III. ENUM 서버 구현 및 고찰

그림 16은 ENUM DNS 서버가 실행된 후 작업 관리자를 통하여 프로세스가 활성화 화면이다.

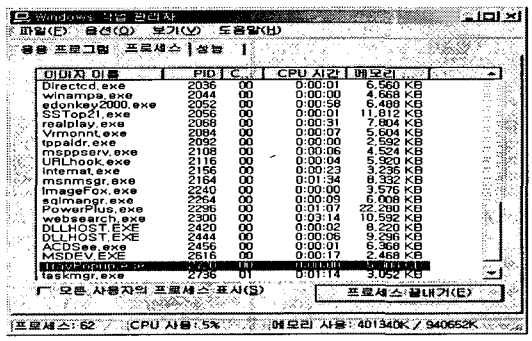

그림 16. ENUM 서버 실행

1. ENUM DNS DB

그림 17은 국가 코드인 2.8.에 대한 DB생성과 지역 코드인 1.3.에 대한 table을 생성한 체계적인 ENUM DNS를 구성과 MS-SQL2000을 사용함으로써 Homepage와

의 연동을 유연하게 활용할 수 있다.

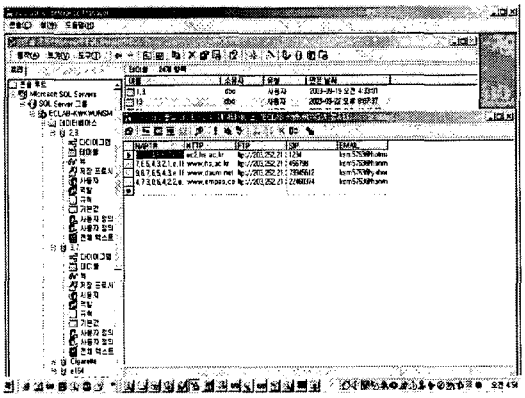

그림 17. ENUM 서버 DB

2. ENUM DNS DB 연결 함수

그림 18은 클라이언트에서 넘어온 FQDN에서 국가 코드에 맞게 DB에 연결하는 DBConn ect() 함수를 구현한 화면이다.

```
BOOL DBConnect()
{
    // 연결 설정을 위한 변수들
    SQLRETURN Ret;
    char szSQL[256];

    for(int r=0; r<256; r++)
        szSQL[r] = '00';

    // 환경 핸들을 할당하고 버전 속성을 설정한다.
    if (SQLAllocHandle(SQL_HANDLE_ENV,SQL_NULL_HANDLE,&hEnv) != SQL_SUCCESS)
        return FALSE;
    if (SQLSetEnvAttr(hEnv,SQL_ATTR_ODBC_VERSION,(SQLPOINTER)SQL_OV_ODBC3,
        SQL_IS_INTEGER) != SQL_SUCCESS)
        return FALSE;

    // 연결 핸들을 할당하고 연결한다.
    if (SQLAllocHandle(SQL_HANDLE_DBC,hEnv,&hDbc) != SQL_SUCCESS)
        return FALSE;

    if(cnt==0)
    {
        cnt++;
        szSQL[0]='2';   szSQL[1]='.';   szSQL[2]='0';   szSQL[3]='.';
    }
    else
        wsprintf(szSQL,"'%s'",Country);

    Ret = SQLConnect(hDbc, (SQLCHAR *)szSQL, SQL_NTS,(SQLCHAR *)"sa", SQL_NTS, (SQLCHAR *)"199756884", SQL_NTS);

    if ((Ret != SQL_SUCCESS) && (Ret != SQL_SUCCESS_WITH_INFO))
        return FALSE;

    // 중장 핸들을 할당한다.
    if (SQLAllocHandle(SQL_HANDLE_STMT,hDbc,&hStmt) != SQL_SUCCESS)
        return FALSE;

    return TRUE;
}
```

그림 18. ENUM 서버 DB 연결 함수

3. ENUM DNS SQL 실행 함수

그림 19는 클라이언트에서 넘어온 FQDN에서 지역 코드에 맞게 테이블에서 NAPTR과 일치되는 쿼리를 서버에 가지고 오는 부분이다. 이렇게 클라이언트에서는 필요한 서비스를 DNS에서 쿼리를 받아 클라이언트에서 실행한다.

```
SQL DBExecuteSQL()
{
    //발용 지역 초기화
    for(int i=0; i<50; i++)
    {
        Http[i]='00';
        Ftp[i]='00';
        Sip[i]='00';
        Email[i]='00';
        Naptr[i]='00';
    }

    SQLINTEGER lNaptr,lHttp,lFtp,lEmail,lSip;
    // 최근 출력을 위한 변수들
    int y=1;
    HDC hdc;
    int arlab[2]={200,400};
    char str[256];
    char szSQL[256];
    // 결과를 출력하기 위해 바인드한다.
    SQLBindCol(hStmt,1,SQL_C_CHAR,Naptr,sizeof(Naptr),&lNaptr);
    SQLBindCol(hStmt,2,SQL_C_CHAR,Http,sizeof(Http),&lHttp);
    SQLBindCol(hStmt,3,SQL_C_CHAR,Ftp,sizeof(Ftp),&lFtp);
    SQLBindCol(hStmt,4,SQL_C_CHAR,Sip,sizeof(Sip),&lSip);
    SQLBindCol(hStmt,5,SQL_C_CHAR,Email,sizeof(Email),&lEmail);
    // SQL문을 szSQL에 저장하는 부분
    wsprintf(szSQL,"SELECT NAPTR,HTTP,FTP,EMAIL,SIP FROM %s WHERE NAPTR='%s'",Local,DBsend);
    // SQL문을 실행한다.
    if (SQLExecDirect(hStmt,(SQLCHAR *)szSQL, SQL_NTS) != SQL_SUCCESS)
    {
        return FALSE;
    }
    // 일어온 타이틀 출력
    hdc=GetDC(hWndMain);
    while(SQLFetch(hStmt)!=SQL_NO_DATA)
    {
        wsprintf(str,"NAPTR:%sHTTP:%sFTP:%s",Naptr,Http,Ftp,Email,Sip);
        TabbedTextOut(hdc,10,y*20,str,strlen(str),2,arlab,0);
        y++;
    };
    ReleaseDC(hWndMain,hdc);

    if (hStmt) SQLCloseCursor(hStmt);
    return TRUE;
}
```

그림 19. ENUM 서버 SQL 실행 함수

IV. 결 론

본 논문에서는 단 하나의 연결점을 통해 다수의 정보에 접근할 수 있는 ENUM 전화번호(E.164)를 이용한 ENUM 서버의 설계하였다.

이를 위해 ENUM API의 대표적인 응용이라 할 수 있는 웹 클라이언트의 구현을 통해 대표적인 ENUM서비스인 ENUM HTTP 서비스와 ENUM FTP 서비스를 ENUM root domain(e164.arpa) 를 활용하여 구현하였다.

특히, 본 본문에서 셀계한 ENUM 서비스를 본격적으로 활용을 한다면 ENUM 시스템 구축으로 인한 신규 장비의 수요증대와 기존 전화와 인터넷 사용자에게 하나의 전화번호로 단일화된 통신환경을 제공 및 ENUM 통합서비스로 통신비용이 감소할 수 있다.

향후 연구과제로는 본 논문의 연구결과를 바탕으로 하는 ENUM 서비스의 SIP분야, FAX분야 등 PSTN

망을 바탕으로 하는 서비스 분야의 대한 표준 API의 개발이 필요할 것이다.

참 고 문 헌

[1] P. Faltstrom, "E.164 Numbers and DNS," RFC2916

[2] M. Mealling, "The Naming Authority Pointer(NAPTR) DNS Resource Record," RFC2915

[3] 김원, 유지열, 신성우, 송관호, "ENUM 최신기술 및 동향에 대한 연구", 한국인터넷정보학회 학회지, 제3권, 제3호, 2002.

[4] 김노식, 권성민, 손승환, 강민구, "Plug-in을 적용한 ENUM서버와 서비스구현", 2003년 한국인터넷정보학회 추계학술발표대회 논문지, 제4권, 제2호, 2003.

[5] 한국인터넷정보센터(KRNIC) "ENUM API에 관한 표준화 연구", 2002.

[6] 한국인터넷정보센터(KRNIC) "NAPTR DNS에 대한 SIP 연동 프로토콜 표준화 연구", 2002.

[7] http://www.enumf.org

[8] http://uriforum.or.kr

[9] http://ietf.org

[10] http://www.enum.or.kr

저 자 소 개

강 민 구(Min-Goo Kang) 정회원

- 1986년 2월 : 연세대학교 전자공학과(공학사)
- 1989년 2월 : 연세대학교 전자공학과(공학석사)
- 1994년 2월 : 연세대학교 전자공학과(공학박사)
- 1985년~1987년 : 삼성전자 통신연구소 연구원
- 1997년~1998년 : 오사카대학 통신공학과 Post Doc.
- 1994년~2000년 : 호남대학교 정보통신공학과 교수
- 2000년~현재 : 한신대학교 정보통신학과 교수

<관심분야> : 모바일정보통신, 디지털방송통신시스템