임베디드 리눅스를 기반의 건물 정보시스템의 구현

김용기*, 이병권*, 전중남*
*충북대학교 전자계산학과
couragesuper@hotmail.com, {sonic747, joongnam}@cbu.ac.kr

An Implementation of Building Information System based on Embedded Linux

YongGy Kim*, ByungKwon Lee*, Joongnam jeon*
*Dept of Computer Science, Chung-Buk University

요 약

임베디드 리눅스를 상에서 수행되는 건물 정보 서비스를 구현하였다. 종래의 웹에서 제공하는 맵(map) 서비스 기능에 건물 내부의 정보를 조회하는 기능을 추가하였다. 건설교통부에서 제공하는 수치 지형도를 사용하였고, 이로부터 지리정보데이터베이스를 구축하기 위하여 파서 및 속성정보 입력기를 구현하였다. 건물 정보를 표현하기 위하여 OralceSpatial 컴포넌트를 사용하였다. 그리고 Qt-E을 사용하여 지리정보를 보기 위한 클라이언트 프로그램을 개발하였다.

1. 서론

최근에는 GIS(Geographic Information Service)가 낯선 개념이 아니다. 자동차에서 네비게이터 기능을 통해 위치를 확인 및 이동하는 목적지까지의 경로를 안내 받는 GPS가 대표적이며 휴대폰이나 PDA를 통한 위치 추적 같은 개인 서비스도 제공 중이다.

GIS의 응용 범위는 지역정보 시스템, 도시 정보 시스템, 토지 정보 시스템, 교통 정보 시스템, 지하 매설물 관리 시스템 등이 있다. 본 연구는 임베디드 리눅스 기반 시스템에서 수행되는 건물 정보 시스템(BIS, Building Information System)을 구현하는 과정을 설명한다. BIS는 기존의 GIS 기능에 건물의 상세한 정보를 서비스하는 기능이 추가된 시스템을 의미한다. 건물은 층, 내부 시설, 주소, 소유주, 시가 등의 여러 가지 속성을 가지기 때문에 특수한 용도에 따른 BIS는 한 지역에서 특징적인 검색으로 위치 기반의 검색을 더 할 수 있다는 점에서 이용도가 크다고 할 수 있다.

GIS 계층을 작성하기 위하여 건설교통부에서 발행하는 수치 지형도를 사용하였으며, 파서 및 오라클 컴포넌트를 이용하여 웹 DB를 구축하고, Qt-E를 사용하여 클라이언트 어플리케이션을 작성하였다[2][3].

OracleSpatial은 오라클 DBMS 제품군에 속하는 컴포넌트로써 지리정보를 위한 SQL스키마, 인덱싱 메커니즘, 함수 연산들의 집합이다[4]. OracleSpatial은 공간정보의 입출력에 대한 편의성을 제공하며 새로운 좌표공간의 정의 및 공간 인덱싱과 공간 산술 연산 등을 제공한다.

Qt-E 라이브러리가 범플랫폼적인 성격을 가지기 때문에 익숙한 환경에서 작업을 하고 임베디드 리눅스에 전달할 수 있으며, 오라클 컴포넌트의 사용으로 지리정보와 속성데이터의 결합이 가능하다[1][5][6].

2. 위치 정보 획득

지리 정보를 얻기 위해 건설교통부에서 발행하는 수치 지형도 데이터를 구입하였다[4][5]. 이 지형도 데이터는 AutoCad의 저장 형식인 DXF 파일 형태로 제공이 되며, 여러 개의 섹션과 섹션 개별적인

개체로 구성되어 있다[4].

2.1 DXF 파일의 구조

DXF 파일은 텍스트 파일 구조와 이진 구조 두 가지로 구분되며 두 가지 경우 모두 개체의 타입에 따라 다른 형식의 파싱 방법이 요구된다. 모든 데이터는 데이터 식별 번호인 정수와 그 값인 스트링의 쌍인 그룹 코드로 구성되는데 [표1]은 DXF의 개체 영역의 문맥을 기술하고 있다. 영역은 시작과 끝을 나타내는 경계 구분자와 임의의 개수의 개체들에 대한 속성 정보로 구성된다.

표1. AutoCad 파일의 Entity Section

```
    0
  SECTION
    2
  ENTITIES
  //  Beginning of ENTITIES section
    0
  <entity type>
    5
   <handle>
    330
   <pointer to owner>
    100
   AcDbEntity
    8
   <layer>
    100
   AcDb<classname> .
     . <data>
     .
    // One entry for each entity definition
    0
   ENDSEC
   End of ENTITIES section
```

2.2 파서

파서는 C++코드로 직접 구현한 인식기를 사용하였다. [표2]는 오브젝트 정보를 추출하여 개체를 생성하는 파서를 의사 코드(pseudo-code)로 표현한 것이다.

표2. Entity 파서의 Pseudo Code

```
if( Field name is SECTION )
Next Token;
while( bEndSection is false )
{
  if( group code is 0 )
  {
    if( Field name is "Endsec" )
      bEndSection = true;
    else if ( Field name is "insert" )
    {
      Create Insert Node / Push Node to vecEntity
    }
    .
    .
    .
    else if( Field name is "Vertex" )
    {
      Create Vertex Node / Push Node to vecEntity
    }
    else{
      Next Token
    }
  }
}
```

2.3 속성 데이터의 추가

위치 정보를 기반으로 속성 정보를 추가하는 수 있도록 어플리케이션을 작성하였다. 어플리케이션은 선택된 DXF 파일들을 파싱하여 트리 컨트롤에 나열하고, View에서 오브젝트들이 선택되었는지를 판단하여 데이터베이스 속성 입력기에 표시한다.

건물을 바탕으로 서비스 시설, 상가 시설, 내부 부서를 선택적으로 추가하며 작업 내용을 메타(meta) 데이터로 저장하거나 오라클을 위한 SQL 구문을 생성한다. [그림1]은 어플리케이션이 수행될 때 화면의 구성을 보여준다.

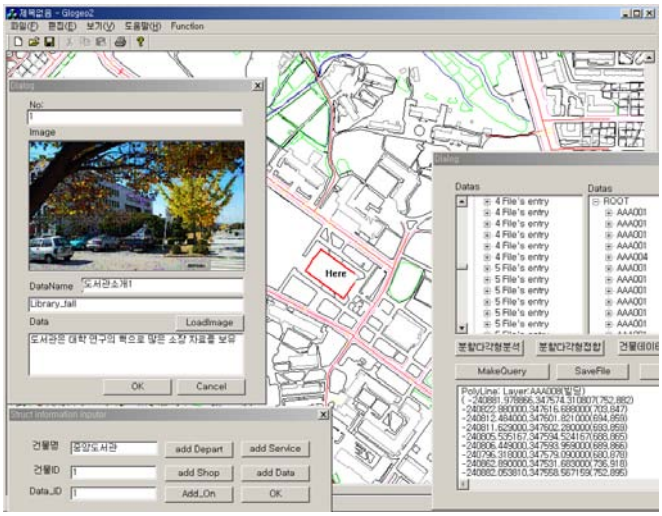

그림1. BIS 정보 등록

2.4 데이터베이스 엔티티 설계 및 DFD

데이터베이스는 건물을 중심으로 부속되어 있는 부서, 상가, 지원 시설 등을 속성으로 추가할 수 있도록 만들어져 있다. 건물 테이블에 위치 정보를 저장하는 필드가 있다는 것이 일반 GIS 시스템과 다른 특징이며, 다른 사항은 일반적인 RDB(Relation DB)의 데이터베이스의 구성과 같도록 하였다. [그림2]와 같이 건물은 건물의 ID와 지리 정보로 구성된다. 그리고 건물 ID 기반으로 부서와 서비스 시설, 상가들이 건물에 속하도록 한다. 어플리케이션이 데이터베이스화 작업 중일 때는 재작업을 방지하기 위해서 현재 작업을 저장할 수 있어야 한다. 이를 위하여 파싱된 개체를 메타 데이터로 저장하는 계층을 추가할 필요가 있다. [그림3]은 최초의 DXF 파일로부터 최종적인 데이터베이스가 생성되는 과정을 설명하고 있다.

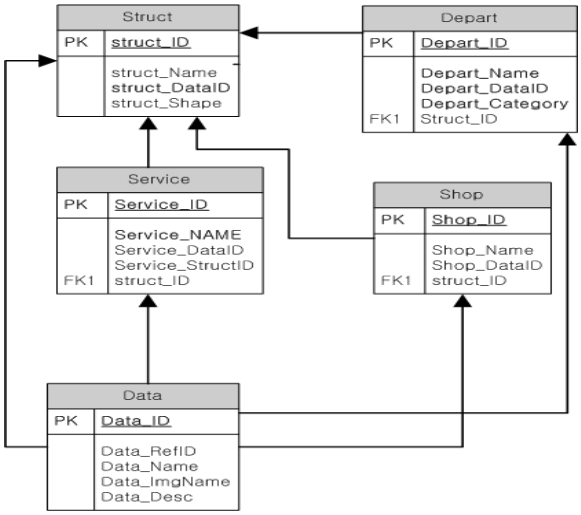

그림2. 데이터 Entity 설계도

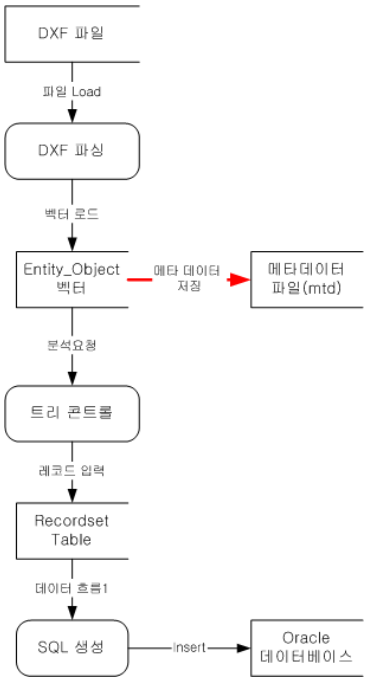

그림3. 데이터베이스 생성

2.5 오라클 데이터베이스

오라클 제품군은 OracleSpatial이라는 제품군을 제공하여 공간 요소들에 대한 데이터베이스화에 편의성을 제공한다. 오라클에 지리정보 데이터를 생성하는 과정을 테이블의 생성, 공간 설정, 공간 인덱스 생성, 레코드 입력의 과정을 거친다.

[표3]은 오라클 공간 컴포넌트를 저장할 테이블을 생성하는 질의문장이다. 지리정보는 제공되는 스키마를 사용하기 때문에 데이터 형식을 MDSYS.SDO_GEOMETRY로 설정하였다.

테이블을 생성한 후에는 공간 데이터베이스의 범위를 결정한다. SDO_GEOM_METADATA에 차원

표3. 건물테이블의 생성

```
create table Struct(struct_Id int primary key ,
                    struct_Name varchar(30) ,
                struct_Shape MDSYS.SDO_GEOMETRY,
                      --Spatial Schema
                    strcut_DataId int );
```

표4. 좌표 공간의 설정과 인덱스 생성

```
-- creating coordinate metadata
insert into user_sdo_geom_metadata values(
  'struct' , 'struct_Shape ' ,
   MDSYS.SDO_DIM_ARRAY(
   MDSYS.SDO_DIM_ELEMENT('X',
     239981.540000 , 241776.649000 , 0.005 ),
   MDSYS.SDO_DIM_ELEMENT('Y',
     346397.150000 , 348629.380000 , 0.005 ) ), NULL );
-- making index
CREATE INDEX Struct_Spatial_Idx
          ON Struct(struct_Shape)
          INDEXTYPE IS MDSYS.SPATIAL_INDEX;
```

단위의 배열로 저장되는데, 2차원의 경도와 위도 단위의 공간을 [표4]와 같이 생성한다. 또한, 건물 테이블의 지리정보에 대해 인덱스를 설정하고 실제로 건물 정보를 입력한다. 지리정보를 데이터베이스에 하나의 객체로 저장하기위한 방법으로 MDSYS.SDO_GEOMETRY 사용한다. MDSYS.SDO_GEOMETRY는 지리 정보의 데이터를 구성하는 여러 개의 인자들로 구성이 되며 ORDINATE_ ARRAY는 지리 데이터의 위치정보를 저장하는 좌표배열이고, ELEM_INFO_ARRAY에서 이 좌표 배열이 어떠한 형태의 지형, 지물 데이터인지 해석하는 정보를 담고 있다. [표5]는 2차원의 다각형 형태의 데이터에 DXF 파일에서 추출한 위치정보를 저장한 인스턴스의 입력 과정을 보여주고 있다.

표5. 건물 데이터 입력

```
-- insert object
I insert into Struct values( 1 , '중앙도서관' ,
  MDSYS.SDO_GEOMETRY(2003,NULL,NULL,
    MDSYS.SDO_ELEM_INFO_ARRAY(1,1003,3),
    MDSYS.SDO_ORDINATE_ARRAY(
      240881.979,347574.311,
      240822.880,347616.688,
                 ....
      240806.449,347593.959,
      240796.318,347579.090,
      240862.890,347531.683,
      240882.054,347558.567)) , 1 );
```

3. QT/E를 이용한 어플리케이션 계층을 개발

어플리케이션은 윈도우즈에서 개발하였고, 이것을 리눅스 플랫폼으로 포팅하기 위하여 리눅스용 GUI 개발 도구인 QT/E를 어플리케이션 라이브러리로 사용하였다[8][9]. 임베디드 리눅스 시스템으로 포팅하는 과정은 다음과 같다.

① Linux 기반에서 프로젝트 포팅기본 프로그램을 VFB(virtual frame buffer)상에서 실행시키기 위하여 크로스 컴파일 한다.

② x86용 Qt-E에 맞는 QTDIR 과 LD_LIBRARY_PATH를 지정한다.

③ 명령(1)로 프로젝트 파일을 모의 테스트 한다.

```
bash>  # qvfb -width 640 -height 480 -depth16 (명령1)
```

④ Arm용 크로스 컴파일을 위한 Qt-E의 환경설정으로 QTDIR, LD_LIBRARY_PATH를 설정한다(명령2).

```
#> tmake -o Makefile shooting.pro        (명령2)
```

⑤ 컴파일된 클라이언트 어플리케이션을 타켓 보드로 전송한다.

⑥ 타겟 보드에 포팅 라이브러리 및 프로그램 다운로드 한다.

⑦ Arm용으로 컴파일된 Qt-E GUI 라이브러리 libqte.so.2.3.2를 다운로드 한다.

⑧ 명령(3)으로 라이브러리를 심볼릭링크 한다.

```
#>ln -s libqte.so.2.3.2 libqte.so.2.3        (명령3)
```

⑨ 임베디드 리눅스상의 환경을 다시 설정하기 위해 [표6]과 같이 bash_profile에 대하여 환경을 설정한다.

표6. bash_profile 환경 설정

QTDIR=/jffs/project/qt LD_LIBRARY_PATH=/jffs/project/qt/lib/:$LD_LIBRARY_PATH

[그림4]의 클라이언트 프로그램은 대상 임베디드 기기에서 실행되며 호스트 컴퓨터의 데이터베이스에 접근하기 위해서 소켓을 통해서 패킷을 제공 받아 지정된 건물의 속성정보를 표시하여 준다.

4. 결론

본 연구에서는 임베디드 리눅스 기반의 건물 정보 시스템의 구축하는 과정을 설명하였다. 건물 속성정보와 지리정보와의 연결을 위해 OracleSpatial의 공간컴포넌트가 유용하였고, 윈도우즈 환경에서 개발

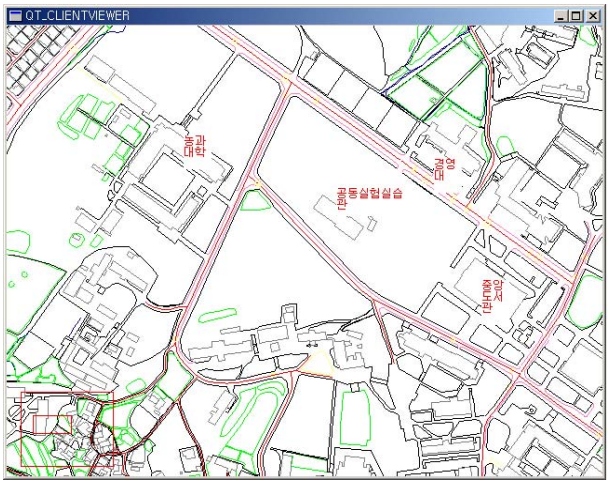

그림4. 클라이언트 어플리케이션

된 어플리케이션을 리눅스 환경으로 포팅하는 과정에 Qt-E 라이브러리를 사용함으로써 개발의 편의성을 제공해 주었다.

향후 연구에서는 임베디드 기기를 위한 BIS를 인터페이스로 사용하는 각종 지역 광고 시스템이나 내부 시설 제어 시스템에 관한 방안을 생각해 볼 수 있다.

참고문헌

[1] Mattbas Kalle Dalbeimer , "Progarmming with Qt" - 2nd Edition , O'REILLY

[2] 국립지리원, "내부 포맷 및 포맷 변환에 관한 연구",

[3] Oracle® Spatial User's Guide and Reference Release 9.2 , http://download-west.oracle.com/docs/html/A96630_01/toc.htm

[4] AutoCAD, http://www.autodesk.com/techpubs/autocad/acad200dxf/index.htm

[5] TROLLTECH, "http://www.trolltech.com/"

[6] Patrick Ward,"Qt programming for Linux and Windows 2000,"PH PTR, 2001.

[7] fuber,"ARM system on chip architecture",secend edition, ADDSION WESLEEY, 2000.

[8] Karim Yaghmour, "Building Embedded LINUX SYSTEMS" O'RELLY, 2003.

[9] 유상철, 박철, "Embedded Linux System 구조 및 설계 응용," OHM사, 2003.