XML 데이터베이스를 이용한 전자해도의 관리

이성대* · 박휴찬**

Management of Electronic Navigational Charts Using XML Database

Seong Dae Lee* · Hyu Chan Park**

요 약

전자해도는 해안선, 수심, 항로표지 등 선박의 항해에 필요한 정보를 담고 있는 디지털 해도로서 S-57 형식으로 기술되어 있다. 이러한 전자해도가 선박의 안전 운항에 성공적으로 사용되고 있지만, 특수한 데이터 형식과 표시 시스템으로 인하여 제한적으로만 사용되고 있고 응용 범위 또한 제한적일 수밖에 없다. 이러한 문제를 해결하기 위해서는 S-57 형식의 전자해도를 GML과 같은 범용의 형식으로 변환하여 데이터베이스화하고 인터넷을 통하여 접근 가능하게 할 필요가 있다. 이에 본 논문에서는 S-57 전자해도를 GML로 변환하는 방법과 XML 데이터베이스를 이용하여 관리하는 방법을 제안한다. 또한 이 XML 데이터베이스로부터 사용자의 검색 질의를 처리하는 방법을 제안한다. 이러한 방법의 타당성을 검증하기 위하여 프로토타입 시스템을 개발하고 여러가지 시험 수행을 진행하였다. 이러한 시스템을 이용하면 일반인들도 전자해도에 포함되어 있는 다양한 해양정보에 손쉽게 접근할 수 있을 뿐만 아니라 GML과 XML 데이터베이스에 의한 호환성 및 효율성의 증대도 가능하다.

ABSTRACT

Electronic Navigational Charts (ENCs) are digital charts encoded in S-57 format, which contain navigational informations such as coastlines, depth areas, and nautical marks. Although they have been successfully used for the safe navigation of ships, they have limited usages and applications because of their specialized data format and access systems. To cope with such drawbacks, S-57 ENCs need to be transformed into more generalized format such as Geography Markup Language (GML). The transformed GML ENCs can be kept in a database for efficiency, and can be accessed through Internet for usability. This paper proposes a new method for transforming and managing ENCs with XML database. S-57 ENCs are first transformed into GML format, and then stored in a XML database. On the database, users can query for their needs. To validate the feasibility of the proposed method, we developed a prototype system, and then conducted several test runs. The system can provide users with easy access to the marine information contained in ENCs. It also provides compatibility and efficiency by virtue of GML and XML database, respectively.

키워드

전자해도, XML 데이터베이스, S-57, ENC, GML

Ⅰ. 서 론

전자해도 ENC(Electronic Navigational Chart)는 해안선, 수심, 등대 및 부이 등 항로표지, 위험물, 항로 등과 같은 다양한 정보를 국제수로기구 IHO(International Hydrographic Organization)의 S-57 형식에 따라 제작한

* 한국해양대학교 산학협력단

** 한국해양대학교 컴퓨터·제어·전자통신공학부 (교신저자)

접수일자 : 2007. 7. 23

디지털 해도이다[1]. 이러한 전자해도는 해양 지리정보로서의 유용성에도 불구하고 전자해도 표시시스템, 항해용 전자참고도, 혹은 어선 조업용 장치 등과 같은 특수한 목적의 장비에서만 주로 사용되고 있다. 하지만 최근 해양에 대한 관심이 높아지면서 언제 어디서나 바다에 대한 정보를 이용하려는 사용자들의 요구가 증대되고 있다. 이러한 요구에 부응하기 위하여 전자해도가 활용될 수 있음에도 불구하고 고가의 전용 장비나 브라우저를 구비해야 하는 문제 때문에 그 활용이 쉽지 않다. 또한, 전자해도는 특수한 형태의 S-57로 기술되어 있기 때문에 다른 시스템과의 호환성이 떨어지고 다양한 활용에 많은 어려움이 있다.

이러한 어려움을 극복하기 위하여 일련의 연구가 있어왔다. 우선, S-57 전자해도를 XML 형식으로 변환하여 데이터 교환의 용이성과 호환성을 높이려는 연구가 있었다[2]. 하지만 기본적인 XML에만 기반하고 있기 때문에 복잡한 형태의 전자해도를 효율적으로 표현하기에는 다소 어려움이 있다. 이러한 어려움을 극복하기 위한 시도로 GML(Geography Markup Language)에 기반한 연구를 들 수 있다. GML은 지리정보를 체계적으로 손쉽게 표현할 수 있는 표준으로 OGC(Open Geospatial Consortium)에서 개발되었다[3]. 이 GML은 다른 시스템과의 호환성이 우수하고 사용성 및 확장성이 뛰어나다는 장점으로 많은 분야에 적용할 수 있다. 이러한 GML로 S-57 전자해도를 변환하여 데이터베이스에 저장한 후, 사용자의 검색조건에 부합하는 전자해도를 웹상에 브라우징하는 연구가 있었다[4]. 하지만 변환된 GML 전자해도를 관계형 데이터베이스에 저장하기 때문에 속도가 느리고 체계적으로 저장하는데 다소 어려움이 있다.

그 외 Galdos사에서는 전자해도용 GML 응용스키마를 제안하여 GML 기반의 전자해도 연구에 적용할 수 있게 하였고[5], ESRI사에서는 자사제품인 ArcGIS에 S-57을 로딩할 수 있는 기능을 추가하였다[6]. 그리고 C-Navi사에서는 PDA에서 전용의 브라우저를 사용하여 전자해도를 표시할 수 있는 시스템을 개발하였다[7].

본 논문에서는 우선 S-57로 표현된 전자해도를 GML로 변환한다. 그리고 변환된 GML 전자해도를 효율적으로 관리하기 위하여 기존의 관계형 데이터베이스가 아닌 XML 데이터베이스에 저장한다. 저장된 XML 데이터베이스에 대하여 사용자가 원하는 정보를 손쉽게 검색할 수 있는 기능을 제공한다. 그 결과, GML에 의한 데이터 교환과 웹 접근이 용이해질 뿐만 아니라 데이터베이스에 의한 관리가 용이해지고 사용자의 사용 목적에 따라 다양한 활용이 가능하게 된다.

논문의 2장은 관련연구로서 S-57 전자해도, 전자해도용 GML 응용스키마, XML 데이터베이스, XML 질의어에 대하여 살펴본다. 3장에서는 전자해도의 변환, 변환된 전자해도의 XML 데이터베이스 저장, 저장된 전자해도의 검색을 위한 질의처리 방법 등을 제안한다. 4장에서는 시스템의 구현 및 시험 수행을 기술하고, 5장에서 결론 및 향후 연구과제를 논한다

II. 개요 및 관련 연구

먼저, 본 연구의 대상인 S-57 전자해도에 대하여 간략히 소개한다. 다음으로, 본 연구의 기반 기술인 전자해도용 GML 응용 스키마, XML 데이터베이스에 대하여 살펴본다.

2.1 S-57 전자해도

S-57은 전자해도 등 해양 데이터의 표현 및 교환을 위한 표준으로 국제수로기구 IHO가 2000년에 버전 3.1을 공표하였다[1]. S-57로 표현된 전자해도 데이터를 이용하기 위해서는 전자해도 표시시스템 ECDIS(Electronic Chart Display and Information System)와 같은 전용의 장비를 필요로 한다. 그림 1은 이러한 S-57 전자해도를 전용의 표시시스템에 브라우징한 예를 보이고 있다.

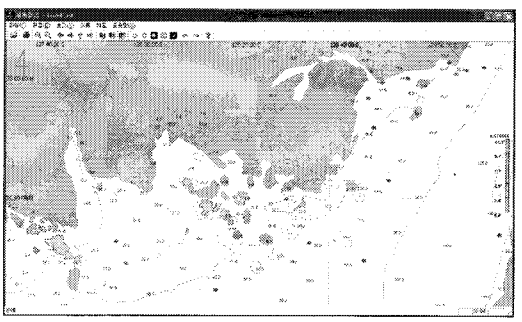

그림 1. 전자해도의 예
Fig. 1 Example of Electronic Navigational Chart

전자해도를 위한 데이터 형식인 S-57은 현실세계의 실체들을 객체(Object)라는 단위로 모델링한다. 이러한 객체는 피쳐 객체(Feature Object)와 공간 객체(Spatial Object)로 분류된다. 그림 2에서는 등대, 항구, 부이 등과 같은 피쳐 객체와 위치를 표현하는 공간 객체간의 관계를 보여주고 있다. 각 객체는 고유한 식별자(Identifier)와 어트리뷰트(Attribute)들로 구성된다.

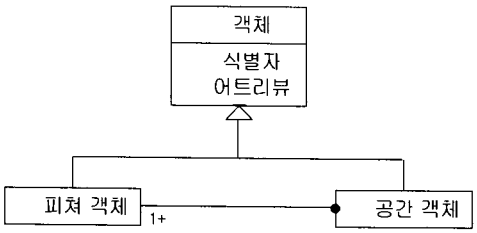

그림 2. S-57 객체 클래스
Fig. 2 S-57 Object Class

피쳐 객체는 표 1과 같은 네 가지 범주로 분류된다. Geo 객체는 등대, 항구, 부이 등과 같은 실세계의 실체에 대한 정보를 표현하며 대부분의 객체를 포함한다. Meta 객체는 데이터의 정확도, 축척 등의 정보를 표현한다. Collection 객체는 데이터의 집합이나 연관관계 등의 정보를 표현한다. Cartographic 객체는 지도 기호의 표현을 위한 영역, 문자열 등의 정보를 표현한다.

표 1. S-57 피쳐 객체의 범주
Table. 1 Classification of S-57 Feature Object

범주	설명	예
Geo Object	실세계 엔티티들의 상세한 특성들을 포함	등대: LIGHT
Meta Object	다른 객체들에 대한 정보를 포함	데이터 정확도: M_ACCY
Collection Object	다른 객체간의 관계에 대한 정보를 포함	연관: C_ASSO
Cartographic Object	지도 기호의 표현에 대한 정보를 포함	영역: $AREAS

공간 객체는 위도 및 경도 좌표의 쌍, 또는 위도, 경도 좌표 및 깊이의 쌍으로 표현되는 위치 정보이다. 공간 객체를 표현하기 위한 방법으로 S-57에서는 벡터(Vector), 래스터(Raster), 메트릭스(Matrix)의 세 가지 방법을 제공하고 있다. 그 중에서 가장 많이 사용되는 벡터 모델을 UML(Unified Modeling Language) 다이어그램으로 표현하면 그림 3과 같다. 그림에서 알 수 있듯이 벡터로 표현되는 정보의 형태는 NODE, EDGE, FACE의 세 가지이며, 지도에서의 Point, Line, Area과 각각 대응된다. Point는 위도 및 경도 좌표를 가지는 한 지점의 위치정보이며, Line은 Point와 Point의 연결, Area는 Point들의 연결로 구성된 다각형 공간으로 표현된다.

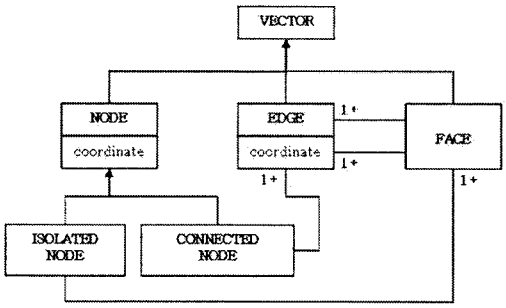

그림 3. 공간 객체의 종류
Fig. 3 Classification of Spatial Object

또한 S-57은 각 객체를 위한 식별자와 다양한 어트리뷰트들을 제공한다. 식별자는 해당 객체를 고유하게 식별하기 위한 정보를 가지고 있으며, 어트리뷰트들은 해당 객체가 어떤 세부 속성들을 가지고 있는지를 기술한다. 이러한 어트리뷰트는 세 가지로 분류할 수 있다. 피쳐 객체 어트리뷰트는 피쳐 객체와 관련된 속성을 정의하며 대부분의 어트리뷰트를 포함하는데, 색깔(COLOUR)과 같은 것을 예로 들 수 있다. 국가언어 어트리뷰트는 국가언어정보(NINFORM)와 같은 속성을 포함한다. 공간 및 메타 객체 어트리뷰트는 공간과 메타 객체에 관련된 속성을 나타내는데, 예를 들면 위치정확도(POSACC)와 같은 것이 있다. 이러한 어트리뷰트는 다음과 같은 세 가지의 집합으로 분류할 수도 있다.

- Set Attribute_A : 객체의 개별적 속성
- Set Attribute_B : 데이터 사용에 관련된 정보
- Set Attribute_C : 객체와 데이터에 대한 관리정보

2.2 전자해도용 GML 스키마

GML은 지리정보의 체계적인 표현과 교환을 위한 표준으로 OGC에서 제안되었으며 XML에 기반하고 있다.

2007년에 버전 3.2.1이 공표되기에 이르렀고 ISO의 표준으로도 채택되었다[3]. GML은 피쳐 스키마(Feature Schema)와 지오메트리 스키마(Geometry Schema) 등 26개의 스키마로 구성되어 있고, 이를 코어 스키마(Core Schema)라 한다. 이들 스키마 간의 관계를 표현하면 그림 4와 같다. 피쳐 스키마는 지오메트리 스키마를 포함하고, 다시 지오메트리 스키마는 XLinks 스키마를 포함한다.

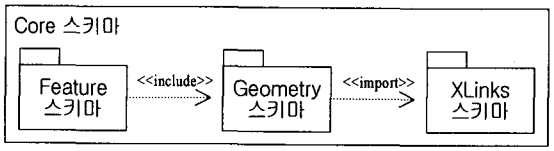

그림 4. GML 코어 스키마
Fig. 4 GML Core Schemas

GML의 피쳐 스키마는 실세계의 피쳐 정보를 표현할 수 있는 기본 틀을 제공한다. 피쳐는 도로, 강, 사람, 운송수단, 행정 경계와 같은 의미를 가지는 실세계의 객체이다. GML에서 피쳐는 XML의 엘리먼트(Element)로 기술되며, 세부적인 특성을 표현하기 위한 프로퍼티(Property)와 어트리뷰트(Attribute)의 집합으로 구성된다. 각 피쳐는 피쳐의 위치를 정의하는 gml:location 프로퍼티와 피쳐를 둘러싸는 경계를 정의하는 gml:boundedBy 프로퍼티를 선택적으로 가진다. 또한, 모든 피쳐는 고유한 식별자인 gml:id 어트리뷰트를 가진다. GML의 지오메트리 스키마는 실세계의 지오메트리 정보를 표현할 수 있는 기본 틀을 제공한다. 즉, Point, LineString, Box, LinearRing, Polygon과 여기서 확장된 MultiPoint, MultiLineString, MultiPolygon등의 지오메트리 타입을 제공한다.

이상과 같은 코어 스키마는 지리 정보를 표현하기 위한 기본적인 틀만을 제공한다. 따라서, 특정 분야에서의 구체적인 데이터를 표현하기 위해서는 응용 스키마(Application Schema)를 정의해야 한다. 즉, 응용 스키마는 코어 스키마를 확장해 실제 사용할 타입이나 프로퍼티를 정의해 놓은 스키마이다. 본 논문에서는 Galdos사에서 개발한 S-57 전자해도용 응용 스키마를 채택하였다[5]. 그림 5에서는 이 스키마의 구조를 보여주고 있으며, 각 스키마의 구성은 다음과 같다.

- Objects.xsd: S-57 전자해도의 객체들에 대한 정의
- Attributes.xsd: S-57 전자해도의 어트리뷰트들에 대한 정의
- AbstractAndSuperTypes.xsd: 추상 타입 및 메타 데이터에 대한 정의
- SupportTypes.xsd: 기본적인 타입에 대한 정의

위의 네 가지 응용 스키마 및 코어 스키마 중에서 필요한 것들만 모아서 gml4s57.xsd라는 스키마를 만들어 전자해도의 표현에 사용하게 된다.

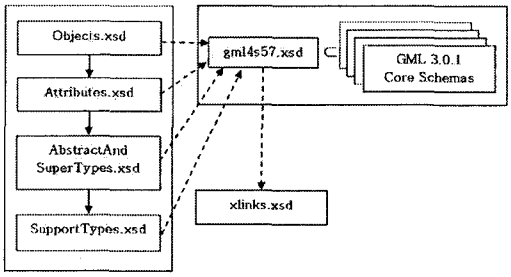

그림 5. S-57 응용 스키마의 구조
Fig. 5 Structure of S-57 Application Schema

이러한 응용 스키마 중에서 Objects.xsd 파일의 Anchor Berth 오브젝트에 대한 정의는 그림 6과 같다. 이 오브젝트는 S-57의 ACHBRT라는 피쳐 객체를 정의하고 있으며 어트리뷰트로 catach, datend 등을 정의하는 것을 볼 수 있다.

```
<!--============== element ==================-->
<element name="AnchorBerth" type="s57:AnchorBerthType"
substitutionGroup="gml:_Feature">
 <annotation>
  <documentation>The "AnchorBerth" feature corresponds
to the s57 object "ACHBRT". A designated area of water where
a single vessel, sea plane, etc... may anchor.
  </documentation>
 </annotation>
</element>

<!--============== complexType ================-->
<complexType name="AnchorBerthType">
 <complexContent>
  <extension base="s57:AbstractFeatureType">
   <sequence>
    <element name="acronym" type="string"
fixed="ACHBRT"/>
    <element ref="s57:catach" minOccurs="0"/>
    <element ref="s57:datend" minOccurs="0"/>
```

```
        <element ref="s57:datsta" minOccurs="0"/>
        <element ref="s57:nobjnm" minOccurs="0"/>
        <element ref="s57:objnam" minOccurs="0"/>
        <element ref="s57:perend" minOccurs="0"/>
        <element ref="s57:persta" minOccurs="0"/>
        <element ref="s57:radius" minOccurs="0"/>
        <element ref="s57:status" minOccurs="0"/>
        <element ref="s57:inform" minOccurs="0"/>
        <element ref="s57:ninfom" minOccurs="0"/>
        <element ref="s57:ntxtds" minOccurs="0"/>
        <element ref="s57:scamin" minOccurs="0"/>
        <element ref="s57:txtdsc" minOccurs="0"/>
        <element ref="s57:sordat" minOccurs="0"/>
        <element ref="s57:sorind" minOccurs="0"/>
        <choice>
         <element name="position"
type="s57:NodePropertyType" minOccurs="0"/>
          <element name="extent"
type="s57:FacePropertyType" minOccurs="0"/>
        </choice>
       </sequence>
      </extension>
     </complexContent>
    </complexType>
```

그림 6. S-57 응용 스키마의 예
Fig. 6 Example of S-57 Application Schema

2.3 XML 데이터베이스

XML은 W3C에서 제정한 표준으로 사용이 간편하고 재사용성 및 확장성이 뛰어나다[8]. 이러한 장점으로 인해 XML은 전자거래, 전자민원서비스 등 많은 분야에서 활용되면서 웹상에서 유통되는 많은 정보들이 XML 전자문서로 생성되고 있다. 이에 따라 이러한 XML 전자문서들을 효과적으로 저장, 관리할 수 있는 방법이 요구되고 있다[9,10].

XML 문서를 관리하는 전통적인 방법으로는 파일시스템 방식, 통합 RDBMS 방식, 분할 RDBMS 방식이 사용되고 있다. 파일시스템 방식은 개별 XML 문서를 파일의 형태로 특정 폴더에 저장하는 것으로, 문서 전체에 대한 추가 작업이 파일단위로 이뤄져 상대적으로 저장 및 추출 속도가 빠르나 XML 문서를 추출할 때마다 XML 파싱(Parsing)이 필요하다는 단점이 있다. 통합 RDBMS 방식은 XML 문서를 RDBMS의 BLOB(Binary Large Object)나 CLOB(Character Large Object)의 형태로 저장하는 것으로, 문서의 저장 및 추출 속도가 빠르나 문서의 부분 수정이나 검색에 많은 시간이 필요하다는 단점이 있다. 분할 RDBMS 방식은 XML 문서를 구성하는 각 요소 단위로 분할하여 테이블의 필드에 저장하는 것으로, 문서의 부분 수정이나 검색을 빠르게 수행할 수 있으나 문서를 추출할 때마다 여러 테이블에 대한 조인 과정을 거치므로 속도가 느리다는 단점이 있다.

이러한 문제점을 해결하기 위하여 최근에는 XML 데이터베이스가 활용되고 있다. XML 데이터베이스는 그 특성에 따라 XML 가능 데이터베이스와 XML 전용 데이터베이스로 구분할 수 있다. XML 가능 데이터베이스는 기존의 데이터베이스에 XML 문서를 위한 확장된 기능이 부가된 데이터베이스이다. 대표적인 솔루션으로는 IBM의 DB2 XML Extender, Informix의 Web DataBlade, Microsoft의 XML for SQL Server 2005, Oracle의 Oracle 10g 등이 있다. XML 전용 데이터베이스는 XML 문서에 대한 모델을 정의하고, 그 모델에 따라 XML 문서를 저장하고 추출할 수 있는 데이터베이스이다. 대표적인 솔루션으로는 Software AG의 Tamino, X-Hive사의 X-Hive/DB, eXcelon사의 eXtensible Information Server(XIS), Ipedo사의 Ipedo 등이 있다. 본 논문에서는 eXcelon 사의 XIS를 이용하여 GML 전자해도용 데이터베이스를 구축하였다.

Ⅲ. XML 데이터베이스를 이용한 전자해도의 관리

S-57 전자해도를 보다 손쉽게 이용하기 위해서는 전자해도용 GML 응용 스키마를 따르는 GML 문서로 변환하여 데이터베이스로 관리하는 것이 필요하다. 본 장에서는 이러한 변환 프로그램과 변환된 GML 문서를 저장하고 관리하는 XML 데이터베이스의 구조를 제안한다. 또한 데이터베이스에 저장된 전자해도의 검색을 위한 사용자 질의 처리 방안을 제안한다. 그림 7은 본 논문에서 제안하는 시스템의 전체적인 구조를 보여준다.

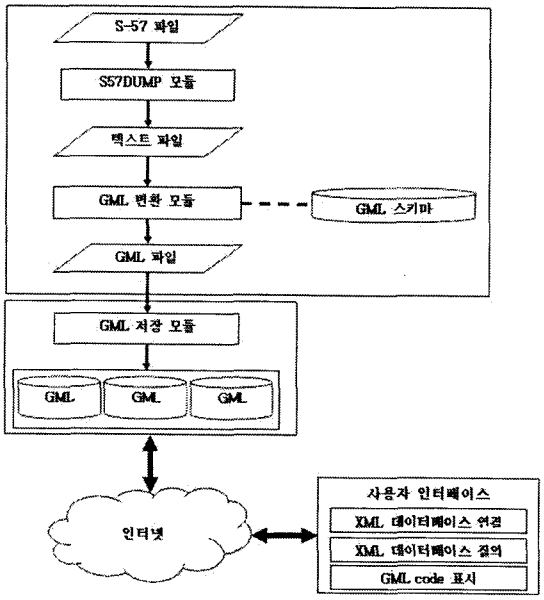

그림 7. 전자해도용 시스템의 구조
Fig. 7 Overall Structure of ENC System

3.1 S-57 전자해도의 GML 변환

전자해도를 위한 데이터 표준인 S-57은 객체의 집합으로 구성된다. 객체는 지도상의 등대, 부이 등과 같은 지리적 피쳐 객체와 위치 정보를 나타내기 위한 공간 객체로 구성된다. 각 객체는 고유한 식별자와 속성들을 기술하기 위한 어트리뷰트들로 구성된다. S-57 전자해도는 이진 형식의 파일로 만들어 졌으며 객체 단위의 순차적인 구조를 갖는다. 이러한 S-57 전자해도를 GML로 변환하는 과정은 그림 8과 같다. 먼저, 이진 형식의 S-57 전자해도 파일을 입력으로 받는다. 입력 받은 S-57 전자해도를 gdal-1.1.5 프로그램[12]을 사용한 S57DUMP 모듈이 객체 단위로 분리하여 텍스트 파일로 출력한다. 이 텍스트 파일을 GML 변환 모듈이 GML 스키마를 참조하여 GML 파일로 변환하게 된다.

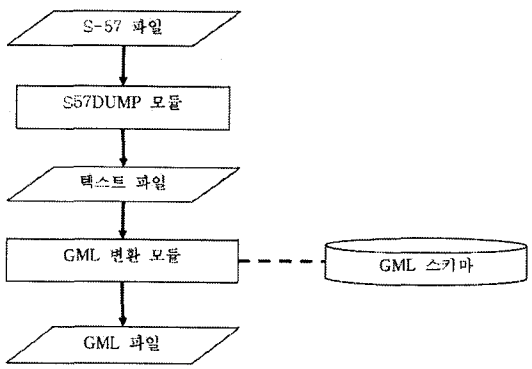

그림 8. S-57의 GML 변환
Fig. 8 Conversion of S-57 to GML

GML 변환 모듈의 변환 알고리즘을 그림 9에서 보여주고 있다. 텍스트 파일로 변환된 S-57 전자해도 데이터를 'T'라고 정의하고 입력으로 사용하게 된다. 알고리즘에서 사용되는 함수들의 기능을 살펴보면 다음과 같다.

- get_line(): 입력 'T'에서 1줄씩 내용을 읽어와 read_line에 넣는다.
- tokenizer(): read_line 1줄을 '*' 단위로 자르고 token 배열에 저장한다.
- isobject(): token이 오브젝트인지 판별한다.
- isfeature_object(): 피쳐 객체인지 판별한다.
- feature_pattern_matching(): 피쳐 객체라고 판별된 token을 GML 응용 스키마의 피쳐 스키마를 참조해서 GML로 만들고 GML_token에 값을 넘겨준다.
- spatial_pattern_matching(): 공간 객체라고 판별된 token을 GML 응용 스키마의 지오메트리 스키마를 참조해서 GML로 만들고 GML_token에 값을 넘겨준다.
- GMLfile.write(): GML_token을 최종 GML파일에 출력한다.

위의 feature_pattern_matching(), spatial_pattern_matching() 함수에서 사용되는 스키마는 전자해도용 GML 응용 스키마로 Objects.xsd, Attributes.xsd, Support.xsd, AbstractAndSuperTypes.xsd, gml4s57.xsd, xlinks.xsd 파일을 참조하여 적합한 GML 문서로 변환한다.

```
Algorithm S-57 to GML

//입력: S57DUMP를 이용하여 변환된 S-57 텍스트 파일
('T')
//출력: 변환된 GML 파일 ('GMLfile')

begin
 while (T == EOF) {
  //텍스트 파일을 1줄씩 읽어들인다.
  read_line = get_line() ;
```

```
//1라인을 *단위로 자른다.
token[] = tokenizer(read_line, '*') ;
for (i = 0 ; i < token.size() ; i++) {
 if (isobject (token[i])) { //객체인 경우
  if (isfeature_object(token[i])) { //피쳐 객체인 경우
   //텍스트 토큰을 GML 피쳐 토큰으로 변환
   GML_token = feature_pattern_matching(token[i]) ;
   //출력에 GML_token을 추가
   GMLfile.write(GML_token) ;
  } else { //공간 객체인 경우
   //텍스트 토큰을 GML 지오메트리 토큰으로 변환
   GML_token = spatial_pattern_matching(token[i]) ;
   //출력에 GML_token을 추가
   GMLfile.write(GML_token) ;
  }
 }
}
end
```

그림 9. S-57의 GML 변환 알고리즘
Fig. 9 Algorithm for Converting S-57 to GML

이러한 과정을 통하여 변환된 GML 전자해도의 예로 'RIVERS' 객체의 코드를 그림 10에서 보여준다. 코드에서 알 수 있듯이 다양한 프로퍼티로 구성되어 있으며 S-57 전자해도가 가지고 있던 어트리뷰터 값을 가지고 있다. 변환 과정에서 데이터 값이 없는 S-57의 빈 어트리뷰트는 삭제된다.

```
<RIVERS>
 <gml:metaDataProperty>
  <FeatureRecordIdentifier>
   <group>2</group>
   <objectLevel>114</objectLevel>
   <recordVersion>2</recordVersion>
  </FeatureRecordIdentifier>
 </gml:metaDataProperty>
 <gml:metaDataProperty>
  <FeatureObjectIdentifier>
   <agency>280</agency>
   <featureIdentificationNumber>792874</featureIden
   <featureIdentificationSubdivision>1</featureIdentifi
  </FeatureObjectIdentifier>
 </gml:metaDataProperty>
 <STATUS>
  <Status>
   <code>149</code>
   <idList>1</idList>
   <value>permanent</value>
  </Status>
 </STATUS>
 <SCAMIN>
  <ScaleMinimum>
   <code>133</code>
   <value>50000</value>
  </ScaleMinimum>
 </SCAMIN>
 <gml:element>
  <gml:Polygon>
   <gml:pos>'129.20164000';'35.44349000'</gml:pos>
   <gml:pos>'129.20119000';'35.44310000'</gml:pos>
   <gml:pos>'129.20164000';'35.44349000'</gml:pos>
  </gml:Polygon>
 </gml:element>
</RIVERS>
```

그림 10. 변환된 GML 전자해도의 예
Fig. 10 Example of Converted GML ENC

3.2 XML 데이터베이스 저장

변환된 GML 전자해도의 효율적인 관리를 위해서는 데이터베이스에 저장하는 것이 바람직하다. 하지만, 기존의 관계형 데이터베이스는 XML과는 핵심 데이터 모델이 상이하기 때문에 XML 형식의 데이터를 저장하는데 많은 어려움이 있다. 이러한 어려움을 극복하기 위하여, 본 논문에서는 XML 데이터베이스인 eXcelon사의 XIS를 이용하여 GML 전자해도 데이터베이스를 구축하였다[13]. 이러한 XML 데이터베이스는 GML 전자해도를 체계적으로 저장할 수 있을 뿐만 아니라 사용자의 검색 질의를 효율적으로 처리할 수 있다. XIS를 이용하여 설계한 GML 저장 과정을 그림 11에서 보여준다.

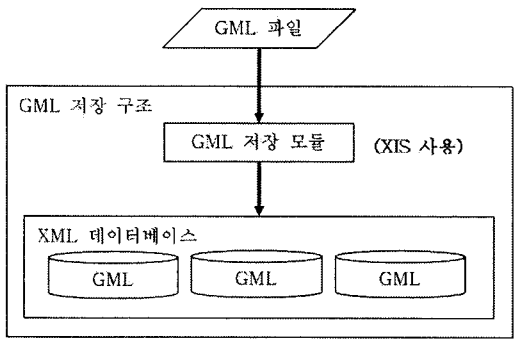

그림 11. GML 저장 과정
Fig. 11 Process of GML Storage

GML 저장 과정 중에서 GML 저장 모듈은 GML 데이터를 스키마와 비교하여 문법적으로나 구조적으로 오류가 있는지는 확인하는 기능을 한다. 그림 12는 이러한 저장 모듈을 순서도로 나타낸 것이다.

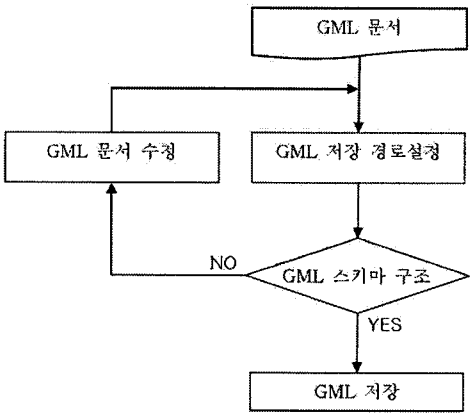

그림 12. GML 저장 모듈의 순서도
Fig. 12 Flowchart of GML Storage Module

GML 전자해도를 실제로 저장하고 있는 XML 데이터베이스의 구조를 세부적으로 보면 그림 13과 같다. 기본적으로 Partition, GML Store, Directory 형태의 계층적 구조로 데이터가 저장되며, 각각을 설명하면 다음과 같다.

- Partition: 물리적인 저장 위치와 운영체제의 저장 위치(Directoy)가 연결되어 있는 것으로 GML Store 그룹이 저장되는 장소이다. Partition의 이름은 서버 상에서 중복될 수 없다.

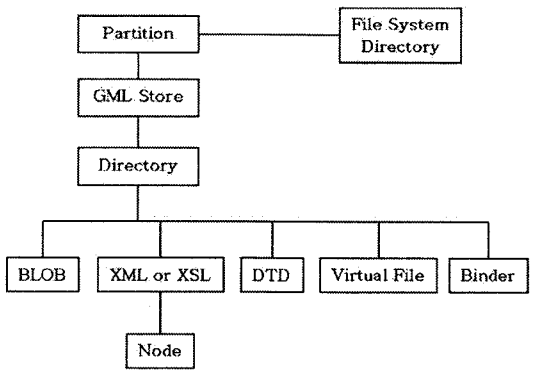

그림 13. XML 데이터베이스의 구조
Fig. 13 Structure of XML Database

- GML Store: 계층적 구조로 저장된 데이터의 최상위 Root 역할을 수행한다. GML Store의 이름은 XIS 서버 상에서 중복될 수 없다. 생성될 GML Store의 개수는 응용프로그램이 처리해야 하는 업무 및 데이터의 양과 그 설계에 의존적이다.
- Directory: Directory 내에는 Child Directory 또는 파일이 저장된다.

3.3 사용자 질의 처리

GML 전자해도에 대한 검색 질의는 사용자 인터페이스와 XML 데이터베이스가 연동되어 처리하게 된다. 먼저, 그림 14에서 알 수 있듯이 사용자는 인터넷을 통하여 웹으로 서버에 접근하게 된다. 이 과정에서 XML 데이터베이스에 연결하기 위하여 다음과 같은 파일이 사용된다.

- Client Session: dxeclient.jar
- Local Session: dxeserver.jar
- Java Binding: dxebind.jar
- J2EE Connector: dxeconnector.jar

위의 파일 중에서 dxeclient.jar와 dxeserver.jar은 XML 데이터베이스와 사용자를 연결해주는 역할을 한다. 즉, dxeclient.jar 파일에는 XML 데이터베이스 사용자 API가 포함되어 있고, 쿼리 및 업데이트 등을 서버에 송신하고 그 결과를 수신하는 Java 모듈로 구성되어 있다. dxeserver.jar 파일은 서버 API가 포함되어 있고, 트랜잭션 관리 및 사용자의 요구를 처리하는 작업을 수행한다.

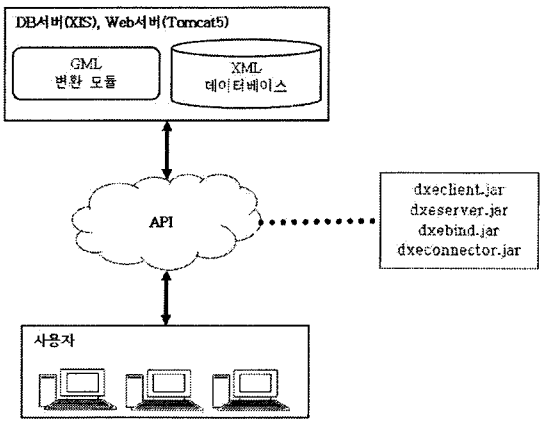

그림 14. XML 데이터베이스와의 연결
Fig. 14 Connection to XML Database

다음으로, XML 데이터베이스에서 검색 질의를 처리하는 과정을 살펴본다. 검색 질의는 XQuery와 XPath를 이용하여 처리하게 되는데, 그 과정을 그림 15에서 보여주고 있다. 먼저, 질의문이 들어오면 이것을 실행이 가능한 트리 형태로 변환하고, 다시 이 트리를 검색하고자 하는 문서에 적합하게 최적화시키게 된다. 이후, 최적화된 트리를 실행하고, 그 결과를 XML 구조에 맞추어 출력하게 된다.

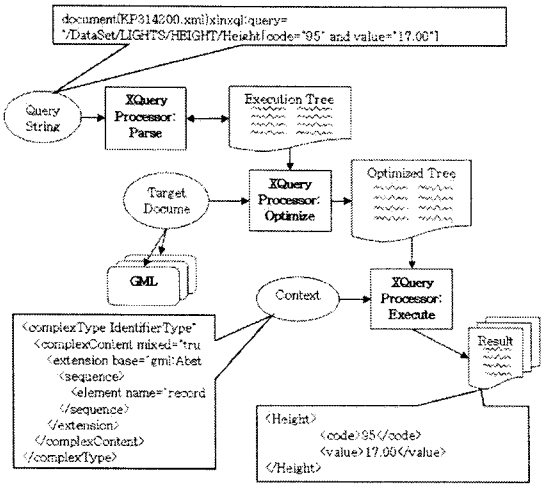

그림 15. GML 질의 처리
Fig. 15 GML Query Processing

그림 16은 GML 전자해도에 대한 검색 과정을 순차적으로 표현한 것이며, 사용자가 XML 데이터베이스에 연결하고 질의를 해서 원하는 결과를 가져오는 과정을 보여주고 있다. 그 순서는 다음과 같다.

① 사용자는 웹을 사용하여 검색하고자 하는 특정 전자해도와 객체를 선택한다.

② Query1.jsp는 선택된 전자해도와 객체를 DBelement.java에 넘겨준다.

③ DBelement.java는 넘겨받은 값을 XML 질의어로 변환하여 XML 데이터베이스에 질의한다.

④ DBCgml.java는 검색된 결과를 데이터베이스로부터 받아서 GML 형식으로 변환한다.

⑤ gmlview.jsp는 GML 형식의 검색 결과를 웹에서 볼 수 있도록 변환한다.

⑥ 웹상에 GML 코드를 표시한다.

위의 순서로 진행되는 과정에서 사용되는 각 모듈의 기능을 설명하면 다음과 같다.

- Query1.jsp: 웹에서 사용자가 전자해도와 객체를 선택할 수 있도록 Select Box를 제공하며 선택된 값을 DBelement.java로 넘겨주는 기능을 한다.
- DBelement.java: Query1.jsp에서 선택된 값을 7개의 클래스 파일 (com.exln.dxe, com.exln.dxe.client, com.exln.dxe.engine, com.exln.dxe.servext, com.exln.dxe.servlet, com.exln.dxe.filesystem, com.exln.dxe.dom)을 사용하여 XML 질의어로 변환하고 XML 데이터베이스 서버에 질의하는 기능을 한다.
- DBCgml.java: XML 데이터베이스 서버로부터 검색 결과를 넘겨받아 최종 GML로 재구성한 후 gmlview.jsp에 넘겨준다.
- gmlview.jsp: DBCgml.java에서 넘겨받은 GML을 사용자가 볼 수 있는 형태로 변환하여 웹브라우즈에 보여준다

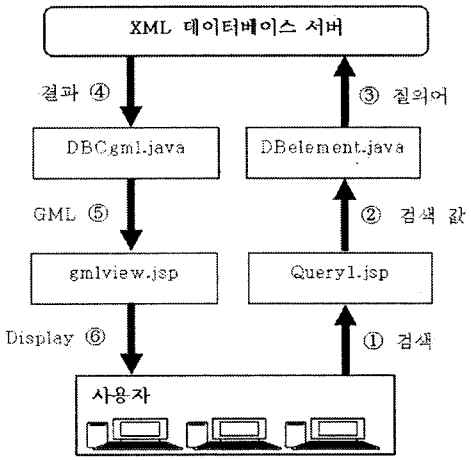

그림 16. GML 검색 과정
Fig. 16 GML Retrieval Process

Ⅳ. 시스템의 구현

S-57 전자해도를 GML로 변환하는 시스템, 변환된 GML 파일을 XML 데이터베이스에 저장하는 시스템, 저장된 데이터로부터 XQuery를 이용하여 검색하는 시스템의 구현에 대하여 설명한다. 시스템의 구현을 위한 환경은 표 2와 같다.

표 2. 구현 환경
Table. 2 Implementation Environment

구 분	구 성	
사용자	웹 브라우저	Internet Explorer 6.0
	운영체제	Microsoft Windows XP
서버	CPU	Intel PentiumIV 1.7Ghz
	Memory	768M
	운영체제	Microsoft Windows XP Professional SP2
	웹 서버	Tomcat5
	XML 데이터베이스 서버	eXcelon (XIS) Version 3.1
	웹 개발 언어	JSP
	GML 변환	JAVA NetBeans 5.0

구현된 시스템의 전체 구성도는 그림 17과 같다. 크게 GML 변환 시스템, GML 저장 시스템, 사용자 질의 시스템으로 구성되어 있다.

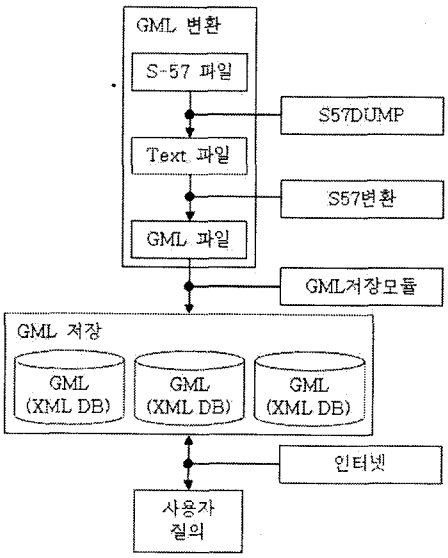

그림 17. 구현된 시스템의 구성도
Fig. 17 Structure of Implemented System

4.1 GML 변환 시스템

S-57은 기본적으로 2진 형식으로 표현되어 있기 때문에 사용하는 목적에 따라 적절한 형태의 형식으로 변환해야 한다. 본 논문에서는 우선 텍스트 파일로 변환한 후, 이를 다시 GML 파일로 변환하는 과정을 거친다. 그림 18에서는 2진 형식의 S-57 전자해도 데이터, 변환된 텍스트 파일, 최종적으로 변환된 GML 파일을 보여주고 있다.

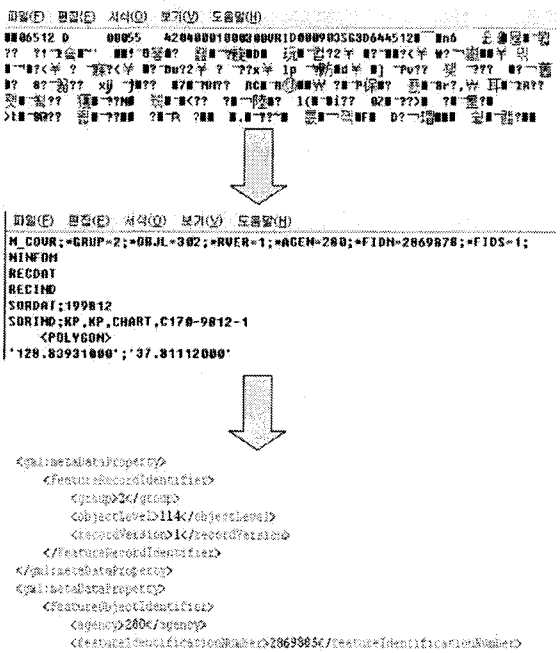

그림 18. S-57 전자해도의 GML 변환
Fig. 18 GML Conversion of S-57 ENC

본 논문에서 시험용으로 사용한 전자해도는 대한민국 연안을 표현한 전자해도 중에서 KP314200.000, KP420400.000, KP52180A.000, KP52180B.000 등이다. 이러한 전자해도에 대한 변환과정에서의 데이터 크기를 살펴보면 표 3과 같은데, 그 크기가 점점 커지는 것을 볼 수 있다. 즉, S-57의 2진 데이터를 텍스트 형태로 바꾸면 그 크기가 증가하게 되고, 다시 텍스트를 GML로 변환하면 모든 정보를 XML의 엘리먼트 형태로 세세히 기술해야 하므로 데이터의 크기가 더욱 커지게 된다. 이러한 문제는 압축 기법을 적용하여 GML 데이터를 압축하면 상당히 경감할 수 있을 것이다.

표 3. 시험 데이터의 크기
Table. 3 Size of Test Data

전자해도 이름	S-57 데이터 크기	텍스트 데이터 크기	GML 데이터 크기
KP314200.000	696KB	2,010KB	5,094KB
KP420400.000	1,448KB	3,902KB	9,827KB
KP52180A.000	135KB	436KB	1,037KB
KP52180B.000	79KB	265KB	640KB

4.2 GML 저장 시스템

변환된 GML 파일을 체계적으로 관리하기 위하여 XML 데이터베이스에 저장하게 된다. 그림 19에서는 XML 데이터베이스의 Partition, GML Store, Directory 형태의 계층적 구조를 보여준다.

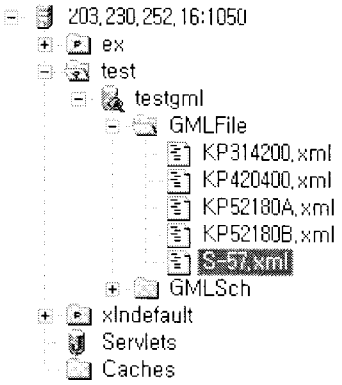

그림 19. XML 데이터베이스의 구조
Fig. 19 Structure of XML Database

4.3 사용자 질의 시스템

GML 전자해도에 대한 사용자 질의를 처리하는 인터페이스를 구현하였다. 먼저, 사용자는 이 인터페이스를 통하여 검색하고자 하는 특정한 전자해도 및 객체를 입력하게 된다. 이후, XML 데이터베이스로부터 검색결과를 전달받아 GML 형식으로 출력하게 된다. 그림 20은 질의 및 검색결과 화면을 보여준다. 그 외의 다양한 검색 기능은 향후에 추가 개발할 예정이다.

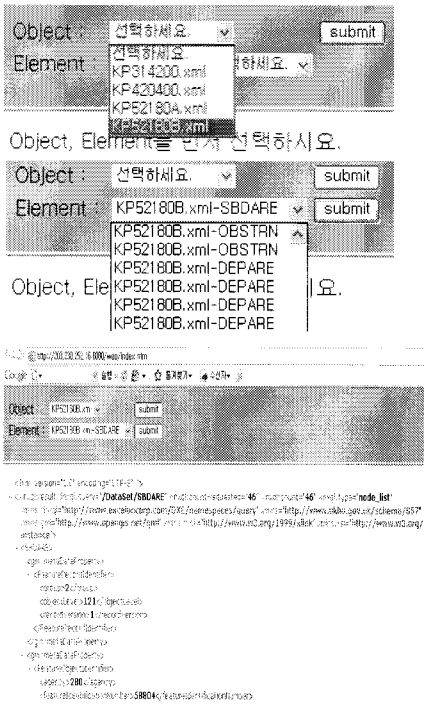

그림 20. 질의 및 검색결과 화면
Fig. 20 Query and Retrieval Result Screen

V. 결론 및 향후 연구과제

기존의 S-57 형식의 전자해도를 좀 더 효율적으로 활용하기 위해서는 지리정보를 체계적으로 기술하고 교환할 수 있는 GML로 변환할 필요가 있다. GML은 범용의 XML 구조를 따르기 때문에 사용이 간편하고 재사용성 및 확장성이 뛰어나다는 장점이 있어 많은 분야에 활용될 수 있다.

본 논문에서는 S-57 전자해도를 피쳐 객체와 공간 객체로 분류하여 Java 언어 및 Galdos사의 전자해도용 GML 스키마를 적용하여 GML로 변환하는 시스템을 개발하였다. 변환된 GML은 기본 형태가 트리 구조이므로 저장 및 관리를 위해 기존의 관계형 데이터베이스를 사용할 경우 속도나 활용성에 어려움이 있다. 이 문제점을 해결하고자 본 문문에서는 XML 데이터베이스를 채택하여 GML 전자해도를 효율적으로 저장할 수 있는 데이터베이스를 구축하였다. 또한, 웹을 통하여 이 데이터베이스에 쉽게 접근할 수 있는 사용자 인터페이스를 개발하였다.

향후에는, 좀 더 다양한 기능이 추가된 사용자 인터페이스를 개발하고 다른 지리정보시스템과의 연계 방안을 연구할 필요가 있다.

참고문헌

[1] International Hydrographic Organization (IHO), IHO Transfer Standard for Digital Hydrographic Data Version 3.1, Special Publication No. 57 (S-57), http://www.iho.shom.fr, 2000.

[2] 이성대, 강형석, 박휴찬, "전자 해도용 XML 스키마의 정의 및 변환", 한국해양정보통신학회논문지, 제8권, 제1호, pp. 200-212, 2004.

[3] Open Geospatial Consortium (OGC), Geography Markup Language (GML) Version 3.2.1, http://www.opengeospatial.org, 2007.

[4] 감승철, 이성대, 곽용원, 박휴찬 "GML과 SVG를 사용한 웹 기반 전자해도 시스템의 개발", 한국해양대학교 부설산업기술연구소 연구논문집, 제23집, pp. 83-88, 2006.

[5] D. S. Burggraf, S-57 Schema and Related Tools Manual, S-57/GML Project, http://www.ukho.gov.uk/b2b_gml_home.asp, 2004.

[6] ESRI, ArcGIS S-57 Converter 9.0 Beta Documentation, 2004.

[7] C-Navi, CnENC Ver. 1.0 Document, 2004.

[8] W3C, Extensible Markup Language (XML) 1.1, http://www.w3.org/TR/xml11, 2006.

[9] 이성대, 곽용원, 박휴찬, "객체관계형 데이터베이스에 기반한 XML 문서 저장 및 검색 시스템의 설계및 구현", 한국해양정보통신학회논문지, 제7권, 제2호, pp. 183-193, 2003.

[10] 권훈, 김정희, 곽호영, "저장 공간과 검색 효율을 위한 XML 문서의 RDB 스키마 모델", 한국콘텐츠학회논문지, 제6권, 제4호, pp. 19-28, 2006.

[11] W3C, XQuery: A Query Language for XML. W3C Working Draft, http://www.w3.org/TR/2003/WD-xquery-20030502/, 2003.

[12] Geospatial Data Abstraction Library(GDAL), Resources for GDAL, http://www.gdal.org.

[13] eXcelon, eXtensible Information Server Developer Guide Documentation, 2003.

저자소개

이 성 대(Seong-Dae Lee)

1999 한국해양대학교 컴퓨터공학과 (공학사)
2001 한국해양대학교 컴퓨터공학과 (공학석사)
2007 한국해양대학교 컴퓨터공학과 (공학박사)
1995~1996 미래정보CIM
2007~현재 한국해양대학교 산학협력단 전임연구원
※관심분야 : 데이터베이스, 데이터마이닝, 해양정보시스템, XML

박 휴 찬(Hyu-Chan Park)

1985 서울대학교 전자공학과 (공학사)
1987 한국과학기술원 전기및 전자공학과 (공학석사)
1995 한국과학기술원 전기및전자공학과 (공학박사)
1987~1990 금성반도체
1997~현재 한국해양대학교 부교수
※관심분야 : 데이터베이스, 해양정보시스템, 데이터마이닝, XML