웹 기반의 전자해도 시스템을 위한 프레임워크

이성대* · 박휴찬**

A Framework for Electronic Navigational Chart Systems Based on the Web

Seong Dae Lee* · Hyu Chan Park**

요 약

전자해도는 연안과 바다에 대한 다양한 정보가 저장되어 있는 디지털 지도이다. 이러한 전자해도에는 일반인들도 관심과 흥미를 가질 수 있는 많은 정보가 포함되어 있지만, 특수한 데이터 포맷과 전용의 표시시스템을 사용하기 때문에 일반인들이 접근하기에는 많은 어려움이 있다. 이러한 어려움을 극복하기 위해서는 누구나 손쉽게 전자해도를 활용할 수 있는 시스템을 개발해야 한다.

이를 위하여, 본 논문에서는 웹 기반의 전자해도 시스템에 적합한 프레임워크를 제안한다. 이 프레임워크에서는, 우선 S-57 포맷으로 코딩되어 있는 전자해도를 GML로 변환하여 데이터베이스에 저장한다. 이 데이터베이스에 대하여 사용자가 자신이 원하는 정보를 요청하게 된다. 이러한 사용자의 요청에 부합하는 정보를 검색한 후, SVG 포맷의 그래픽으로 변환하여 웹에 디스플레이하게 된다. 이러한 프레임워크는 기본적으로 웹에 기반하고 있기 때문에 일반인들도 손쉽게 전자해도를 활용할 수 있다. 뿐만 아니라, 범용 표준인 XML에 기반한 GML과 SVG를 사용하기 때문에 다른 시스템과의 호환성이 우수하고, 데이터베이스에 의한 효율성도 증대된다. 이러한 프레임워크의 효용성을 검증하기 위하여 프로토타입 시스템을 구현하고 시험하였다.

ABSTRACT

Electronic Navigational Charts (ENCs) are digital charts which contain a great variety of data on coast and sea regions. Although they contain much information that ordinary people are interested in, there is no easy way to access because they are coded in the specialized data forma, and can be visualized by using specialized system. Therefore, supporting systems need to be developed for ordinary people to easily access ENCs.

This paper proposes a framework for the ENC systems based on the Web. It relies on quite general standards such as Geography Markup Language (GML) and Scalable Vector Graphics (SVG). In the framework, ENCs coded in S-57 format are first translated into GML to be stored in a database. Once the database is built, users can request to the database what they want. According to the user request, relevant data are retrieved, and then translated into SVG to be displayed on the Web browser. By using the framework, ordinary people may easily access coast and sea information contained in the ENCs. Furthermore, the framework may provide interoperability by virtue of XML-based standards such as GML and SVG, and efficiency by virtue of database. To validate the feasibility of proposed framework, a prototype system is developed and tested.

키워드

전자해도, 웹, S-57, ENC, GML, SVG

* 한국해양대학교 대학원 컴퓨터공학과

** 한국해양대학교 IT공학부 (교신저자)

접수일자 : 2006. 3. 22

Ⅰ. 서 론

인터넷과 웹의 발전으로 많은 분야에서 웹에 기반한 지리정보 시스템이 개발되어 사용되고 있다. 이러한 지리정보 시스템의 개발을 좀 더 촉진하기 위하여 OGC(Open Geospatial Consortium)에서는 지리정보를 체계적으로 표현할 수 있는 GML(Geography Markup Language)을 제안하였고[1], W3C(World Wide Web Consortium)에서는 그래픽 정보를 효과적으로 표현할 수 있는 SVG(Scalable Vector Graphics)를 제안하였다[2]. 이 두 가지는 모두 XML(eXtensible Markup Language)에 기반한 표준 포맷이다. 이러한 기술의 발전으로 인터넷과 웹상에서 시간과 공간에 구애 받지 않고 육상의 지리정보에 손쉽게 접근할 수 있게 되었다.

하지만, 지구표면의 70 퍼센트를 차지하고 있는 바다에 대한 정보를 얻기란 쉽지가 않다. 한 원인으로, 바다가 일부 특정한 사람과 선원에게만 관심이 있는 것이라고 여겨졌기 때문이다. 결과적으로, 관련된 정보 시스템들은 전문적인 데이터 포맷, 접근 방법, 표시 방법을 사용하여 개발되어 왔다. 이러한 흐름의 한 축에 전자해도(ENC: Electronic Navigational Chart)가 있다.

전자해도는 기존의 종이해도를 대체할 수 있는 항해용 디지털 지도로서 종이해도에 비하여 다양한 부가적인 정보와 기능을 제공한다. 전자해도는 국제수로기구(IHO: International Hydrographic Organization)에서 수로 데이터의 코딩과 교환을 위하여 제안한 S-57이라는 특별한 데이터 포맷으로 작성된다[3]. 전자해도는 기존의 종이해도를 점진적으로 대체하고 있으며, 더 많은 데이터와 기능들이 추가될 것이다. 현재, 많은 나라에서 그들의 연안과 바다에 대하여 전자해도를 개발하고 있으며 주기적으로 갱신하고 있다. 이렇게 개발된 전자해도는 전자해도 표시시스템(ECDIS: Electronic Chart Display and Information System)을 통하여 선박의 안전 운항에 성공적으로 활용되고 있다.

비록 전자해도가 선박의 안전한 운항을 위해서 개발되었지만, 많은 사람들이 관심과 흥미를 가질 수 있는 다양한 정보도 포함하고 있다. 즉, 전자해도에는 바다의 해안선, 수심, 부두, 등대에서부터 연안지역의 도로, 강, 육상 목표물에 이르기까지 수많은 정보가 표현되어 있다. 하지만, 이러한 정보를 조회하기 위해서는 전문적인 시스템을 통해서만 가능하기 때문에, 많은 사람들이 관심을 가지고 있음에도 불구하고 이러한 정보에 접근하기란 쉽지가 않다. 더욱이, 특수한 포맷인 S-57을 사용하기 때문에 다른 시스템과의 데이터 교환을 위한 호환성도 떨어진다. 이러한 단점을 극복하기 위해서는 일반인들도 쉽게 접근할 수 있는 웹과 일반적인 표준 포맷을 채택한 시스템을 개발하여야 한다.

본 논문에서는 웹 기반의 전자해도 시스템을 개발할 때 적용할 수 있는 프레임워크를 제안한다. 제안하는 프레임워크에서는 원본 데이터로 S-57을 입력받지만, 실제 전자해도 데이터의 처리를 위해서는 범용의 표준들을 사용한다. 즉, 데이터 변환과 표현 메커니즘으로 GML을, 데이터 저장과 질의를 위해 데이터베이스를, 웹에서의 디스플레이를 위한 그래픽으로는 SVG를 채택한다. 데이터의 흐름에 따라 살펴보면, 먼저 S-57 포맷의 전자해도 데이터를 입력 받아, 이를 GML 포맷의 데이터로 변환하고, 데이터베이스에 저장해 둔다. 이후, 사용자가 원하는 검색 조건을 제시하면, 이 데이터베이스에서 관련된 GML 데이터가 검색된다. 검색된 GML 데이터는 SVG 포맷의 그래픽으로 변환된 후, 최종적으로 웹 브라우저에 디스플레이 된다.

제안하는 프레임워크에 따라 개발된 시스템을 사용하면, 일반인들도 전자해도에 포함된 연안이나 바다에 대한 정보를 쉽게 얻을 수 있다. 또한, GML과 SVG 같은 XML 기반의 표준을 사용하기 때문에 외부 시스템과의 호환성을 높일 수 있고, 인터넷과 웹을 통한 접근성이 증대된다. 나아가, 데이터베이스에 의한 대용량 데이터의 효율적인 관리도 가능해진다. 제안하는 프레임워크의 효용성을 검증하기 위하여 프로토타입 시스템을 구현하고, 샘플 전자해도를 사용하여 시험하였다.

Ⅱ. 개요 및 관련 연구

본 논문에서 채택한 기반 기술인 S-57, 전자해도, GML, SVG, XSLT에 관하여 간략하게 살펴본 후, 본 논문과 관련이 있는 기존의 연구에 대하여 살펴본다.

2.1. S-57과 전자해도의 개요

해도의 내용과 품질은 바다에서의 안전한 활동을 위한 기본이므로, 국제적인 규약에 의해 엄격하게 통제되어야 하고 관련된 제품은 국제적인 인증 표준을 만족해야 한

다. 이 요구사항을 만족시키기 위해, 국제수로기구(IHO)는 S-57이라는 국제 표준을 발표했다[3]. S-57은 '특별 간행물 57번, 디지털 수로 데이터의 교환 표준'을 의미하며, 최종 버전인 3.1이 2000년 11월에 공표되었다.

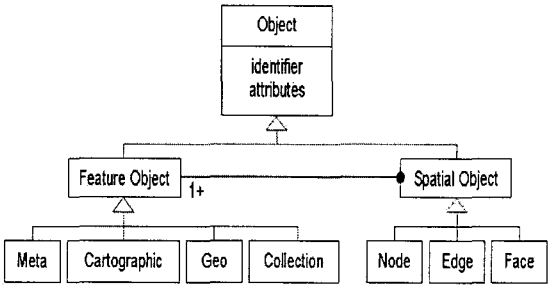


그림 1. S-57 데이터 모델
Fig. 1 S-57 Data Model

그림 1에서 보는 바와 같이, S-57에서 각 객체(Object)는 식별자(identifier)와 속성(attribute)을 가지며, 피쳐 객체(Feature Object)와 공간 객체(Spatial Object)로 분류된다. 피쳐 객체는 실세계의 개체(entity)에 대한 서술적 정보를 기술하고, 공간 객체는 그것의 위치 정보를 기술한다.

피쳐 객체로는 부이(buoy), 표지(beacon), 라이트(light) 등 180여개가 정의되어 있고, 각 객체는 고유 식별자와 몇 개의 속성을 갖는다. 예를 들면, 부이라는 객체는 모양, 색상 등과 같은 속성을 갖는다. 이러한 피쳐 객체는 다시 네 가지의 세부 객체로 분류된다. 즉, 다른 객체들에 대한 공통 정보를 기술하는 메타(Meta) 객체, 실세계 개체의 지도 표현에 대한 정보를 포함하는 지도(Cartographic) 객체, 실세계의 개체를 서술하는 지리(Geo) 객체, 그리고 다른 객체간의 관계를 서술하는 집합(Collection) 객체로 구성된다.

공간 객체는 벡터 모델(vector model), 래스터 모델(raster model), 행렬 모델(matrix model)로 분류될 수 있는데, 버전 3.1에서는 벡터 모델만이 지원된다. 공간 객체는 노드(Node), 에지(Edge), 면(Face)의 3가지 세부 객체로 분류된다. 노드는 하나의 좌표 쌍으로 표현된 공간 포인트이며, 에지는 두개 이상의 좌표 쌍으로 구성되는 선 또는 다중선이다. 면은 동일 노드에서 시작되고 끝나는 연속된 에지들로 구성된다. S-57 파일에서 하나의 공간 객체는 여러 피쳐 객체에 의해 공유될 수 있다. 예를 들면, 부이와 그것에 부착되어 있는 라이트는 서로 다른 피쳐 객체로 정의되지만, 동일한 곳에 위치하기 때문에 하나의 공간 객체를 참조하게 된다.

이러한 S-57은 국가의 수로기관, 선원, 다른 사용자간에 원활한 수로 데이터 교환을 위해 사용될 수 있는데, 가장 중요한 생산물이 전자해도이다. 전자해도는 국가의 수로기관에서 작성한 공식적인 항해용 해도로서 기존의 종이해도와 동등한 지위를 가지며, 추가적인 항해정보를 포함할 수 있다. 이렇게 작성된 전자해도는 전자해도 표시 시스템(ECDIS)을 통하여 선박의 안전 운항에 성공적으로 활용되고 있다[4].

2.2. GML, SVG, XSLT의 개요

1) GML(Geography Markup language) : GML은 지리정보의 모델링, 저장, 교환을 위한 표준으로 OGC에 의해 개발되었다[1]. GML에서 강, 다리, 건물과 같은 실세계의 개체(entity)는 객체(Object)로 모델링된다. 각 객체는 위치나 범위와 같은 공간 프로퍼티(Property), 색상과 높이와 같은 비공간 프로퍼티로 기술된다.

2003년에 공표된 GML 3.0은 XML 스키마(XML Schema) 문법으로 작성된 28개의 코어 스키마(Core Schema)로 구성되어 있다. 그 중에서 피쳐(Feature) 스키마와 지오메트리(Geometry) 스키마는 실세계의 지리정보를 기술하기 위한 핵심적인 객체들을 제공한다. GML 응용 스키마(Application Schema)는 응용 분야에 적합한 객체를 정의하기 위해, 코어 스키마 안에 정의된 객체를 확장하거나 제한하여 만들어진 스키마이다. 즉, 코어 스키마는 '_Feature'와 같은 기본 객체를 제공하고, 응용 스키마는 이를 확장하여 'Bridge'와 같은 객체를 정의한다. 이러한 응용 스키마에 따라 실제 GML 문서가 만들어지는데, '부산대교'와 같은 실세계의 개체에 대한 정보를 기술한다. GML에서 각 객체는 엘리먼트(element)와 대응되는 타입(type)으로 작성된다[5].

2) SVG(Scalable Vector Graphics) : SVG는 2차원 벡터와 벡터/래스터가 혼합된 형태의 그래픽을 XML로 기술하기 위해 W3C에 의해 개발되었다[2]. SVG는 벡터 그래픽, 이미지, 텍스트의 세 종류의 그래픽 객체를 지원한다. 또한 중첩된 변환(nested transformation), 클리핑 패스(clipping path), 알파 마스크(alpha mask), 필터 효과(filter effect)와 템플리트 객체(template object)를 지원한다.

SVG는 이름이 지칭하듯이 벡터 그래픽 및 확대/축소가 가능한 그래픽이라는 두 가지 중요한 특징을 가진다. 즉, 벡터 그래픽은 선과 커브와 같은 기하학적 객체를 포함하며, 모든 픽셀에 대한 정보를 저장해야 하는 래스터 형식보다 더 많은 유연성을 가진다. 또한, 해상도에 따라 확대/축소가 가능하기 때문에 같은 웹 페이지에 서로 다른 크기로 표시될 수도 있고, 다른 페이지에 다른 크기로 재이용될 수도 있다. SVG 그래픽은 다른 SVG 그래픽에 참조되거나 포함될 수 있기 때문에 부분별 그래픽을 통합하여 복잡한 그래픽을 구성할 때 유용하게 사용될 수 있다. 또한, XML 기반이기 때문에 다른 파일이나 그래픽으로의 하이퍼링크와 같은 XML의 기능을 사용할 수 있다[2].

3) XSLT(Extensible Stylesheet Language Transformation) : XSLT는 W3C에 의해 개발되었고, XML 문서를 다른 문서로 변환하기 위한 메커니즘이다[6]. 따라서, XSLT는 XML 기반인 GML 문서를 SVG 문서로 변환하는데 사용될 수 있다. 즉, GML은 지리정보의 내용만을 기술하고 어떻게 표시할 것인지는 지정하지 않기 때문에, 웹 브라우저에 표시하기 위해서는 GML 지리정보를 XSLT를 통하여 대응되는 SVG 그래픽으로 변환하여야 한다. 이 때 변환규칙을 지정한 문서를 스타일시트(Stylesheet)라 한다.

2.3. 관련 연구

S-57 및 전자해도와 관련된 연구로는 활용, 표시 시스템, 응용 프로그램 등에 대한 연구가 있다. S-57은 주로 전자해도를 코딩하기 위해 활용되고 있다[7]. 한국의 국립해양조사원, 미국의 NOAA(National Oceanic and Atmospheric Administration)와 같은 많은 나라의 국가 기관에서 그들의 연안과 바다에 대하여 전자해도를 개발하고 있으며, 정기적으로 갱신하고 있다[8]. 이러한 전자해도를 디스플레이하기 위해 SevenCs사의 SeeMyDEnc와 같은 전문적인 뷰어가 개발되었다[9]. 또한, 전자해도를 활용한 전자해도 표시시스템(ECDIS)은 선박의 안전한 운항을 지원하는 시스템으로서 다양한 시스템이 개발되고 있다[4].

S-57 데이터를 분석하여 다른 포맷으로 변환하기 위한 툴로서는 GDAL(Geospatial Data Abstraction Library)의 S-57 지원 툴킷이 있다[10,11]. ESRI사의 ArcView는 S-57 데이터를 임포트할 수 있는 인터페이스를 제공하고 있다[12]. Courley[13]은 S-57을 이용하여 Multibeam 데이터를 처리하는 방법을 제안하였다. [14]에서는 S-57을 위한 XML 스키마를 제안하고, S-57 데이터를 변환하는 프로토타입을 개발하였다. 최근에 Burggraf[15]는 S-57을 위한 좀 더 일반화된 응용 스키마를 제안하였다.

GML 및 SVG와 관련된 연구로는 저장, 변환, 표시 등에 대한 연구가 있다. Corcoles[16]는 GML에 대한 질의를 처리하기 위해 공간 연산자를 갖는 질의어를 제안하였고, GML 문서를 관계형 데이터베이스에 저장하기 위한 여러 방법을 분석하였다. [17]에서는 객체관계형 데이터베이스에 기반하여 XML 문서를 저장하고 검색하는 방법을 제안하였는데, GML에도 적용할 수 있다.

Guo[18]는 GML 문서를 SVG 문서로 변환하되, 사용자가 정의한 DTD나 스키마를 따르는 SVG 문서로 변환하는 방법을 제안하였다. Tennakoon[19]은 XSLT를 이용한 GML 데이터의 시각화에 대한 방법을 제안하였다. Peng[20]은 코딩 언어로 GML을 사용하고, 웹에서 GML 데이터를 디스플레이하기 위해 SVG를 사용하고, 피쳐 레벨의 데이터에 접근하기 위해 WFS(Web Feature Service)를 사용하는 방법을 제안하였다. 어도비사의 SVG 뷰어[21]와 같이 웹 브라우저에 플러그인할 수 있는 뷰어들도 있다.

위와 같이 부분적으로 관련된 연구는 기존에도 있었지만, 본 논문에서 제안하는 것과 같은 S-57, GML, SVG, 데이터베이스가 전체적으로 연동되는 웹 기반의 전자해도 시스템은 없었다.

III. 웹 기반의 전자해도 시스템

본 논문에서 제안하는 웹 기반의 전자해도 시스템에 적용할 수 있는 프레임워크는 그림 2와 같이 네 단계의 구조로 구성된다.

첫째, S-57 포맷으로 코딩되어 있는 전자해도 문서를 GML 데이터로 변환한다. 이 때 변환된 GML 데이터의 적합성을 보장하기 위하여 S-57 응용 스키마가 참조된다. 이렇게 변환된 GML 데이터는 데이터베이스에 저장되어, 이후의 다양한 전자해도 활용의 중심이 된다.

둘째, 이 데이터베이스에 대하여 사용자는 웹 인터페이스를 통하여 자신이 원하는 정보를 요청하게 된다. 데이터베이스로부터 사용자의 질의에 부합하는 데이터가 검색되고, 검색된 데이터는 다시 GML 문서로 재구성된

다. 이 때 재구성된 GML 데이터의 적합성을 보장하기 위하여 S-57 응용 스키마가 참조된다.

셋째, 재구성된 GML 문서는 XSLT 변환기를 사용하여 SVG 문서로 변환된다. 이 때 GML과 SVG간의 변환규칙을 지정한 스타일시트가 참조된다.

마지막으로, SVG 문서는 프리젠테이션 스타일시트에 따라 SVG 뷰어에 의해 웹 브라우저에 디스플레이된다.

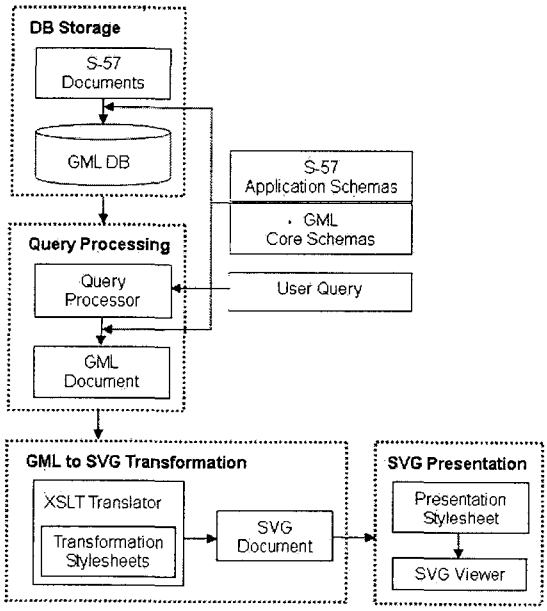


그림 2. 웹 기반 전자해도 프레임워크
Fig. 2. Framework for Web-based Electronic Navigational Charts

3.1. S-57 문서의 데이터베이스 저장

전자해도를 위한 S-57 문서의 내용은 두 종류의 레코드(record)로 구성된다. 피쳐 레코드는 부이(buoy)나 라이트(light)와 같은 실세계 개체의 비공간 속성을 표현하며, 공간 레코드는 실세계 개체의 공간 정보를 서술한다. 그림 3은 부이를 위한 피쳐 레코드와 그 위치를 나타내는 공간 레코드로 구성된 예를 보여준다. 그림에서 알 수 있듯이, S-57 문서의 코딩방식은 정보교환을 위한 명세인 ISO/IEC 8211에 따라 작성되어 있고, 대부분의 데이터는 문자 대신에 숫자 코드가 사용된다[3].

```
019003LE1  0900319  !  5504
00000016300000000100044400163FRID0011400207FOID0007400321AT
TF0006000395NATF0006900450FFPC0008900524...▽
0000;&        △0001FRIDFRIDFOIDFRIDATTFFRIDNATF...▽
0100;&        ISO 8211 Record Identifier△△(I(5))▽
...
00245  D          00109       5504
00010000600000FRID0002600006FOID0001800032ATTF0004100050N
ATF0002900091FSPT0001600120▽
00001▽
FE0000000001P002000180011▽
NL000000000100001▽
000044D000753,1△000763△00116North  Sea  1△▽
00301·N·o·o·r·d·z·e·e·  ·1·△·▽
VI0000000001NNN▽
00110  D          00067       5504
00010000600000VRID0001700006SG2D0002000023▽
00002▽
VI0000000001001I▽
52.10475△4.3004833△▽
```

그림 3. S-57 문서의 예
Fig. 3. An example of S-57 document

이와 같은 S-57 데이터는 GML 데이터로 변환되어 데이터베이스에 저장된다. 이 변환을 위해서는 이전의 연구[14]에서 제안한 바와 같이, GDAL의 S-57 지원 툴킷[11]과 같은 분석 툴킷을 확장하여 사용할 수 있다. 이 때 변환된 GML 데이터는 응용 스키마에 적합해야 한다. 이러한 응용 스키마를 새롭게 정의할 수도 있지만, 본 논문에서는 영국의 UKHO(United Kingdom Hydrographic Office)와 Galdos사가 개발한 S-57 응용 스키마[15]를 채택하였다. 이 응용 스키마는 GML 코어 스키마로부터 파생된 객체, 어트리뷰트, 타입을 정의하고 있다. 그림 4는 'colour'과 같은 비공간 어트리뷰트와 'position'와 같은 공간 어트리뷰트를 포함하는 'Light' 객체의 정의를 보여준다.

```
<!--============== element ====================-->
<element name="Light" type="s57:LightType"
substitutionGroup="gml:_Feature">
  <annotation>
    <documentation> The "Light" feature corresponds to the s57
object "LIGHTS". A luminous or lighted aid to navigation.
    </documentation>
  </annotation>
</element>

<!--============ complexType ==================-->
<complexType name="LightType">
  <complexContent>
    <extension base="s57:AbstractFeatureType">
      <sequence>
        <element name="acronym" type="string" fixed="LIGHTS"/>
        <element ref="s57:catlit" minOccurs="0"/>
        <element ref="s57:colour" minOccurs="0"/>
        <element ref="s57:datsta" minOccurs="0"/>
        ...
        <element ref="s57:sordat" minOccurs="0"/>
        <element ref="s57:sorind" minOccurs="0"/>
        <element name="position" type="s57:NodePropertyType"
minOccurs="0"/>
      </sequence>
    </extension>
  </complexContent>
</complexType>
```

그림 4. S-57 응용 스키마 정의 ('Light')
Fig. 4. Definition of S-57 Application Schemas ('Light')

전자해도 데이터를 데이터베이스에 저장하기 위하여, 본 논문에서는 S-57 대신에 GML을 채택하였다. 이 방식의 장점으로는, GML이 이후의 단계에서 사용될 SVG와 같은 XML 기반이기 때문에 일관성을 증대시킬 수 있다. 더욱이 GML의 범용성 덕분으로 다른 정보시스템과의 상호 운용에 효과적일 수 있다. 데이터베이스 설계를 위해, 본 논문에서는 이전에 연구했던 XML 저장 기법[17]을 채택하였다. 데이터베이스는 해도 이름, 객체 타입, 객체 내용으로 구성된 단일 테이블로 설계되었다.

3.2. 질의 처리와 GML 데이터 검색

일단 데이터베이스가 구축되면, 사용자는 원하는 정보를 요청할 수 있고, 이러한 요청은 SQL(Structured Query Language) 질의 형태로 데이터베이스에 전달된다. 현재 지원되는 질의 형태는 특정 해도의 특정 오브젝트만 검색할 수 있다. 따라서, 이러한 질의에 대한 처리 과정은 기존의 데이터베이스의 단순 선택질의와 동일한 과정을 거치게 된다. 이 과정을 거쳐 검색된 데이터는 S-57 응용 스키마에 적합한 GML 문서로 재구성된다. 그림 5는 하나의 'Light' 객체에 대한 GML 문서의 예를 보여준다.

```
<Light gml:id="_540_2135148864_687">
  <gml:metaDataProperty>
    <FeatureObjectIdentifier>
      <agency>540</agency>
      <featureIdentificationNumber> 213514886
      </featureIdentificationNumber>
    </FeatureObjectIdentifier>
  </gml:metaDataProperty>
  <acronym>LIGHTS</acronym>
  <catlit>
    <CategoryOfLight>
      <code>37</code>
      <value>air obstruction light</value>
    </CategoryOfLight>
  </catlit>
  <colour>
    <Colour>
      <code>75</code>
    </Colour>
  </colour>
  <position>
    <gml:Node gml:id="N120_668">
      <gml:pointProperty>
        <gml:Point gml:id="P120_668">
          <gml:pos>-32.498195 60.895926</gml:pos>
        </gml:Point>
      </gml:pointProperty>
    </gml:Node>
  </position>
</Light>
```

그림 5. GML 문서의 예 ('Light')
Fig. 5 An example of GML document ('Light')

3.3. GML 데이터의 SVG 변환

GML 데이터를 웹 브라우저에 디스플레이하기 위해서는 적절한 그래픽 데이터로 변환해 주어야 한다. 본 논문에서는 이러한 그래픽 데이터로 SVG를 채택하였다. 또한, 변환규칙을 지정하기 위하여 지오메트리 스타일시트와 피쳐 스타일시트를 정의하였다. 지오메트리 스타일시트의 정의를 위해, 우선 GML과 SVG의 지오메트리 엘리먼트 간의 대응 관계를 분석하였다. 표 1의 분석 결과와 같이, SVG가 더 많은 지오메트리 엘리먼트를 지원하기 때문에 GML과 SVG 간에는 1:M의 관계가 형성된다. 하지만 SVG에는 점(point)에 대한 엘리먼트가 없으므로, 본 논문에서는 작은 사각형이나 원으로 대체하였다.

표 1. GML과 SVG의 지오메트리 엘리먼트 간의 대응 관계
Table. 1 Correspondence between geometry elements of GML and SVG

Geometry type	Document type	Element format
Point	GML	<gml:Point> <gml:pos>x1,y1</gml:pos> </gml:Point>
	SVG	<rect x="x1" y="y1" width="w1" height="h1"/>
		<circle x="x1" y="y1" r="r1"/>
Line	GML	<gml:LineString> <gml:pos>x1,y1 x2,y2 ···</gml:pos> </gml:LineString>
	SVG	<line points="x1,y1 x2,y2 ···" />
		<polyline points="x1,y1 x2,y2 ···" />
		<path d="Mx1,y1 Lx2,y2 ···Z" />
Polygon	GML	<gml:Polygon> <gml:pos>x1,y1 x2,y2 x3,y3 ··· x1,y1</gml:pos> </gml:Polygon>
	SVG	<path d="Mx1,y1 Lx2,y2 Lx3,y3···Z" />
Symbol	GML	<gml:Object1> <gml:Point> <gml:pos>x1,y1</gml:pos> </gml:Point> </gml:Object1>
	SVG	<image x="x1" y="y1" "width="w1" height="h1" xlink:href="image path"/>
		<symbol id="symbol1" viewbox="x1 y1 x2 y2"/>
		<use x="x1" y="y1" width="w1" height="h1" xlink:href="symbol1">

분석한 대응 관계에 따라, GML의 지오메트리 를 어떻게 SVG로 변환할 것인지를 지정한 지오메트리 스타일시트를 정의하였다. 그림 6은 'LineString'과 'Polygon'에 대한 스타일시트를 보여준다. GML의 'LineString'은 SVG의 'polyline'으로 변환되고, 'Polygon'은 'path'로 변환된다.

```
<!--========== LineString ================-->
<xsl:template match="gml:LineString">
  <xsl:param name="ID"> </xsl:param>
  <xsl:element name="polyline">
    <xsl:attribute name="id">
      <xsl:value-of select="$ID"/>
    </xsl:attribute>
    <xsl:attribute name="points">
      <xsl:value-of select="normalize-space(gml:pos)"/>
    </xsl:attribute>
  </xsl:element>
</xsl:template>

<!--============ Polygon ==================-->
<xsl:template match="gml:Polygon">
  <xsl:param name="ID"> </xsl:param>
  <xsl:element name="path">
    <xsl:attribute name="id">
      <xsl:value-of select="$ID"/>
    </xsl:attribute>
    <xsl:attribute name="d">
      <xsl:for-each select="*/gml:LineString">
        <xsl:text>M</xsl:text>
        <xsl:value-of select="translate(normalize-space(gml:pos),'
',L')"/>
        <xsl:text>Z</xsl:text>
      </xsl:for-each>
    </xsl:attribute>
  </xsl:element>
</xsl:template>
```

그림 6. 지오메트리 스타일시트 ('LineString' 및 'Polygon')
Fig. 6. Geometry stylesheet ('LineString' and 'Polygon')

또한, GML의 피쳐 엘리먼트를 어떻게 SVG로 변환할 것인지를 지정한 피쳐 스타일시트를 정의하였다. 기본적으로 GML의 피쳐 엘리먼트는 대응되는 SVG로 변환할 수 있다. 하지만 각각의 GML 피쳐 엘리먼트는 부(sub-) 엘리먼트로 몇몇의 지오메트리 엘리먼트를 가질 수 있다. 따라서, 피쳐 엘리먼트를 변환할 때, 그것의 지오메트리 부 엘리먼트도 함께 변환해야 한다. 그러므로, 그림 7의 'Light' 엘리먼트를 위한 스타일시트에서 보는 바와 같이, 피쳐 스타일시트는 지오메트리에 대한 템플릿을 포함하고 있다.

```
<!--============ Light element  ==============-->
<xsl:element name="g">
  <xsl:attribute name="id">L ght</xsl:attribute>
  <xsl:attribute name="class">Light</xsl:attribute>
  <xsl:for-each select="//*[name()='Light']">
    <xsl:apply-templates select=".//gml:Polygon">
      <xsl:with-param name="ID">
        <xsl:value-of select="./*[name()='acronym']" />
      </xsl:with-param>
    </xsl:apply-templates>
    <xsl:apply-templates select=".//gml:LineString">
      <xsl:with-param name="ID">
        <xsl:value-of select="./*[name()='acronym']" />
      </xsl:with-param>
    </xsl:apply-templates>
    <xsl:apply-templates select=".//gml:Point">
      <xsl:with-param name="ID">
        <xsl:value-of select="./*[name()='acronym']" />
      </xsl:with-param>
    </xsl:apply-templates>
  </xsl:for-each>
</xsl:element>
```

그림 7. 피쳐 스타일시트 ('Light')
Fig. 7. Feature stylesheet ('Light')

두 개의 스타일시트를 통해, MSXML과 같은 XSLT 변환기에 의해 GML 문서는 SVG 문서로 변환된다. 그림 8은 변환된 SVG 문서의 예를 보여준다.

```
<?xml version="1.0" encoding="ISO-8859-1" standalone="no"?>
<?xml-stylesheet type="text/css" href="gml2svg_styles.css"?>
<svg width="100%" height="100%" viewBox="-32.500000 60.866667
0.049999999999716 0.100000000000142">
  <desc> Converted to SVG using GMLtoSVG.xsl</desc>

  <g id="DepthArea" class="DepthArea">
    <path id="DEPARE" d="M-32.493758 60.928285 -32.493763
60.932597 -32.493758 60.933328 -32.493182 60.933529
-32.493308 60.933328z" />
    <path id="DEPARE" d="M-32.500000 60.884642 -32.500000
60.883333 -32.499688 32.500000 60.884642 -32.499882
60.884628z" />
  </g>

  <g id="LandArea" class="LandArea">
    <path id="LNDARE" d="M-32.499722 60.898657 -32.498594
60.898996 -32.498594 60.898657z" />
    <path id="LNDARE" d="M-32.497387 60.914768 -32.499326
60.916941 -32.499326  60.914922 -32.497251 60.914922
-32.497387 60.914768z" />
  </g>

  <g id="Light" class="Light">
    <rect x="-32.498195" y="60.895926" height="0.001"
width="0.001"/>
    <rect x="-32.495481" y="60.892104" height="0.001"
width="0.001"/>
    <rect x="-32.491039" y="60.931863" height="0.001"
width="0.001"/>
  </g>
</svg>
```

그림 8. SVG 문서의 예
Fig. 8 An example of SVG document

3.4. SVG 데이터의 디스플레이

변환된 SVG 데이터는 SVG 플러그인을 통해 웹 브라우저 상에 디스플레이될 수 있다. 이러한 플러그인은 기본적인 그래픽 출력 외에 줌, 패닝과 같은 부가적인 기능도 제공한다. 본 논문에서는 각 객체의 공통적인 출력 형태를 지정하는 프리젠테이션 스타일시트를 정의하였다. 그림 9와 같이 각 객체의 채움 및 외곽선의 색상, 외곽선의 너비 등을 지정하고 있다

```
.DepthArea
 {  fill: blue;
    stroke: blue;
    stroke-width: 0.2%;  }
.LandArea
 {  fill: tan;
    stroke: tan;
    stroke-width: 0.2%; }
.Light
 {  fill: yellow;
    stroke: red;
    stroke-width: 0.2%;  }
```

그림 9. SVG용 프리젠테이션 스타일시트
Fig. 9 Presentation stylesheet for SVG

SVG 데이터를 출력할 때 점(point) 객체를 작은 사각형이나 원으로 표시할 수도 있지만, 실제적인 느낌을 주기 위해서는 그 객체의 모양과 유사한 그래픽을 출력하는 것이 좋다. 하지만, 모든 객체에 대해 상세한 그래픽을 SVG 문서에 포함시킨다면, 문서의 용량은 급격하게 증가하게 된다. 이러한 단점을 극복하기 위해서는 각 객체 타입에 대한 그래픽을 심볼로 미리 정의해 둔 후, SVG 문서에서 참조하는 것이 효율적이다. 그림 10은 'Light' 객체를 심블로 정의한 것이다.

```
<?xml version="1.0" encoding="iso-8859-1"?>
<!DOCTYPE svg PUBLIC "-//W3C//DTD SVG 20000303
Stylable//EN"
"http://www.w3.org/TR/2000/03/WD-SVG-20000303/DTD/svg-20000
303-stylable.dtd">
<svg viewbox = "-32.500000 60.866667 0.04999999999716
0.10000000000142">

  <g id="Light">
    <image width="15.000000" height="33.000000"
xlink:href="
iVBORw0KGgoAAAANSUhEUgAAAA8AAAAhCAIAAACJEmZbAAAAfUlEQ
VR42mP4TwfRCDdVBORw0KGdANSR7ziBAAA8OKDSDFJAhCAqeFDCJ
EEZbGFKfUqwEVR12FWZCY5qTA04VWMajy+dYFVNbBokSjVdANSR7zi
BsmtiyipaqlRooKzUZhNOlVw0AvR7ziBtoGW4AAAAASUVORK5CYII=" />
  </g>
</svg>
```

그림 10. 심볼 정의 ('Light')
Fig. 10 Symbol definition ('Light')

Ⅳ. 구현 및 시험

본 논문에서 제안한 프레임워크의 효용성을 검증하기 위하여 프로토타입 시스템을 구현하고, 샘플 전자해도를 이용하여 시험하였다.

4.1. 프로토타입 시스템의 구현

그림 11에서와 같이, 프로토타입 시스템은 클라이언트, 웹 서버, 데이터베이스 서버의 세 부분으로 구성되며, 각각은 세부적인 모듈로 구성된다. 구현환경은 클라이언트 운영체제로 Windows XP Professional, 서버 운영체제로 Windows 2000 Server를 사용하였다. 웹 브라우저는 인터넷 익스플로어 6.0을, 데이터베이스로는 SQL Server 2000을, 개발 도구로 Visual C#.Net 2003을 사용하였다.

클라이언트 측에는 세 종류의 모듈이 있다. 디스플레이 모듈은 SVG 플러그인이 설치되어 있는 웹 브라우저로서, 웹 서버로부터 반환되어 오는 SVG 데이터를 디스플레이한다. 컨트롤 모듈은 줌이나 팬과 같은 기능을 이용하여 디스플레이된 데이터를 컨트롤한다. 컴포넌트 모듈

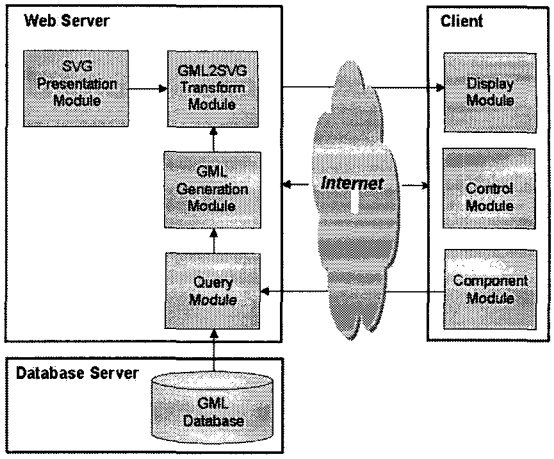


그림 11. 프로토타입 시스템의 구조
Fig. 11 Architecture of prototype system

은 사용자로부터 질의를 입력받아 웹 서버에 전달한다.

웹 서버는 네 종류의 모듈로 구성되어 있다. 질의 모듈은 클라이언트로부터 사용자 질의를 전달 받아, 데이터베이스로 요청을 보내 검색된 결과를 반환받는다. GML 생성 모듈은 반환된 결과를 GML 데이터로 재구성한다. GML2SVG 변환 모듈은 GML 데이터를 SVG 데이터로 변환하여 클라이언트로 전달한다. 이 때, SVG 프리젠테이션 모듈에 의해 최종적으로 디스플레이 될 그래픽의 색상, 선 굵기 등이 지정된다.

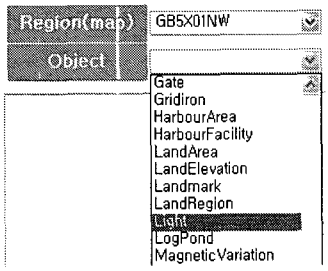
데이터베이스 서버에는 GML 전자해도 데이터가 저장된다. 웹 서버로부터 질의를 전달받아, 데이터베이스를 검색하여, 검색된 결과를 다시 웹 서버로 반환한다.

4.2. 시험 수행

시험 수행에 사용한 전자해도는 UKHO와 Galdos사가 공동 제작한 GML 전자해도로서, 각각의 크기가 60KB에서 27MB에 이른다[15]. 예를 들면, 'GB5X01NW' 전자해도는 파일의 크기가 19MB로서 축척이 25,000:1이고, 69개의 객체 타입을 사용하며, 611개의 피처 레코드와 1,899개의 지오메트리 레코드로 구성되어 있다.

이러한 전자해도를 사용하여 프로토타입 시스템을 테스트하였다. 그림 12는 그 예를 보여주고 있는데, 그림 12(a)는 'GB5X01NW' 전자해도에서 'Light' 객체를 검색하는 질의 화면을 캡쳐한 것이고, 그림 12(b)는 검색된 GML 데이터를 보여주고 있다. 그림 12(c)는 최종적으로 웹 브라우저에 디스플레이된 전자해도와 전체 화면을 캡

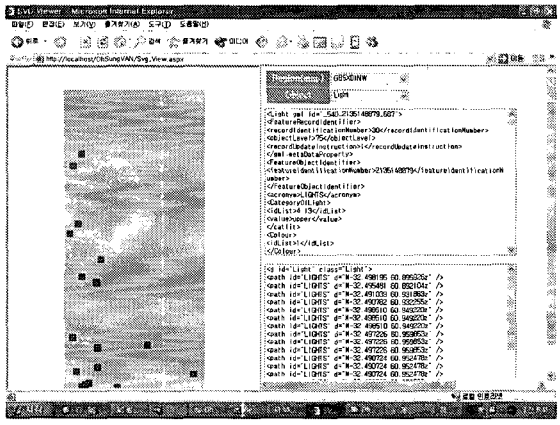
쳐한 것이다. 이 그림에서, 현재의 SVG 플러그인이 심볼 메커니즘을 정상적으로 지원하지 않기 때문에, 'Light' 심볼 대신에 작은 사각형으로 표시하였다.



(a) 질의 화면

	ggml_zone	ggml_clas	ggml_cont
43	GB5X01NW	Light	<Light gml:id="_540_213514869 7_687">
44	GB5X01NW	Light	<FeatureRecordIdentifier>
45	GB5X01NW	Light	<recordIdentificationNumber>332</recordId...
46	GB5X01NW	Light	<objectLevel>75</objectLevel>
47	GB5X01NW	Light	<recordUpdateInstruction>1</recordUpdatel...
48	GB5X01NW	Light	</gml:metaDataProperty>
49	GB5X01NW	Light	<FeatureObjectIdentifier>
50	GB5X01NW	Light	<featureIdentificationNumber>213514869 7</f...

(b) 검색된 데이터



(c) 웹 디스플레이

그림 12. 시험 수행의 예
Fig. 12 An example of test runs

이러한 일련의 시험 수행을 통하여 웹에 기반한 전자해도 시스템의 가능성을 확인할 수 있었다. 즉, 사용자의 측면에서 보면, 단지 웹을 사용할 수 있는 환경만 갖추어지면 언제 어디서나 전자해도에 접근할 수 있음을 확인하였다. 시스템적인 측면에서 보면, S-57, GML, SVG간의 정보변환 과정과 DB와의 연동구조가 아주 자연스럽게 형성됨을 확인하였다. 검색 및 디스플레이 속도 등과 관련하여서는, 비교 대상이 될 만한 기존의 시스템이 없어 비교하지는 못하였다. 다만, SeeMyDEnc와 같은 전문적인 뷰어[9]와 비교하면 디스플레이 속도가 다소 떨어지는 것을 확인하였다. 이러한 속도저하는 주로 SVG 플로그인의 성능에 기인하는 것으로 이 성능이 향상되면 전체적인 시스템의 성능도 향상될 것이다.

V. 결론 및 향후 연구과제

전자해도가 주로 선박의 운항을 지원할 목적으로 개발되었지만, 많은 사람들이 관심을 가질 수 있는 연안이나 바다에 대한 다양한 정보를 포함하고 있다. 그러나, 전자해도가 특수한 포맷인 S-57로 코딩되어 있고 전용의 표시 시스템을 사용하기 때문에 일반인들이 접근하기란 쉽지가 않다. 이러한 어려움을 극복하기 위해서는, 좀 더 범용의 포맷으로 표현되어야 하고, 사용자에게 친숙한 방법으로 접근이 가능해야 한다.

이를 위해, 본 논문에서는 웹 기반의 전자해도 시스템에 적합한 프레임워크를 제안하였다. S-57로 코딩되어 있는 전자해도를 GML로 변환하여 데이터베이스에 저장한 후, 사용자 질의에 따라 관련 데이터를 검색하여 GML로 재구성하고, 이를 다시 SVG로 변환하여 웹 브라우저에 디스플레이한다. 제안한 프레임워크에 따라 구현한 프로토타입 시스템과 일련의 시험 수행을 통하여, 일반인들도 전자해도에 포함된 연안과 바다 정보에 쉽게 접근할 수 있음을 보였다.

현재, 성능 분석, 질의 패턴의 다양화, 사용자 인터페이스의 개선, 데이터베이스의 확장을 위한 연구를 진행 중이다. 향후, 육상의 지리정보 시스템 및 다른 해양정보 시스템과의 연계를 통한 통합 시스템의 개발도 요구되고 있다.

참고문헌

[1] Open Geospatial Consortium (OGC), Geography Markup Language (GML) Implementation Specification, Version 3.0, 2003. http://www.opengeospatial.org.

[2] World Wide Web Consortium (W3C), Scalable Vector Graphics (SVG) 1.1 Specification, 2003. http://www.w3.org/tr/svg11.

[3] International Hydrographic Bureau (IHB), IHO Transfer Standard for Digital Hydrographic Data, Edition 3.1, Special Publication No. 57, 2000. http://www.iho.shom.fr.

[4] Open ECDIS Forum (OEF), Resources for ECDIS. http://www.openecdis.org.

[5] R. Lake, D. S. Burggraf, M. Trninic and L Rae, Geography Markup Language: Foundation for the Geo-Web, John Wiley & Sons, 2004.

[6] World Wide Web Consortium (W3C), XSL Transformations (XSLT) Version 2.0, 2005. http://www.w3.org/tr/xslt20.

[7] M. B. Brown, "Developments in the NOAA Electronic Navigational Chart Program", Proceedings of the U.S. Hydrographic Conference, 1999.

[8] National Oceanic & Atmospheric Administration (NOAA), Resources for the NOAA Electronic Navigational Charts. http://nauticalcharts.noaa.gov/mcd/enc/index.htm.

[9] SevenCs, Software for SeeMyDEnc: Viewer for S-57, DNC and SENC Data. http://www.sevencs.com.

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

[11] Geospatial Data Abstraction Library (GDAL), Resources for S-57 Support of GDAL. http://home.gdal.org/projects/s57.

[12] ESRI, Resources for S-57 Support. http://support.esri.com.

[13] M. Gourley, P. Schwarzberg and G. Noll, "Processing Multibeam Data Through to S-57", Proceedings of the U.S. Hydrographic Conference, 2001.

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

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

[16] J. E. Corcoles and P. Gonzalez, "Analysis of Different Approaches for Storing GML Documents", Proceedings of GIS'02, pp. 11-16, 2002.

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

[18] Z. Guo, S. Zhou, Z. Xu and A. Zhou, "G2ST: A Novel Method to Transform GML to SVG", Proceeding of GIS'03, pp. 161-168, 2003.

[19] W. Tennakoon, Visualization of GML data using XSLT, International Institute for Geoinformation Science and Earth Observation, 2004.

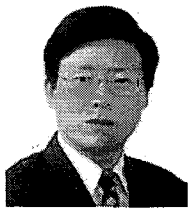
[20] Z. R. Peng and C. Zhang, "The Roles of Geography Markup Language (GML), Scalable Vector Graphics (SVG), and Web Feature Service (WFS) Specifications in the Development of Internet Geographic Information Systems (GIS)", Journal of Geographical Systems, pp. 95-116, vol. 6, 2004.

[21] Adobe Systems, Resources for SVG Support. http://www.adobe.com/svg

저자소개

이 성 대(Seong-Dae Lee)

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



박 휴 찬(Hyu-Chan Park)

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