Journal of the Research Institute of Industrial Technology. Vol. 23, 2006

GML과 SVG를 사용한 웹 기반 전자해도 시스템의 개발

감승철* · 이성대* · 곽용원* · 박휴찬**

*한국해양대학교 대학원, **한국해양대학교 IT공학부 교수

Development of Web-Based Electronic Navigational Chart System using GML and SVG

*S. C. Kam** · *S. D. Lee** · *Y. W. Kwak** · *H. C. Park***

**Graduate school of Korea Maritime University, Busan 606-791, Korea*

***Div. of Information Technology, Korea Maritime University, Busan 606-791, Korea*

요 약 : 최근 네트워크의 발달과 인터넷의 보편화에 따라 지리정보는 많은 분야에서 활용되고 있으며, 그 범위가 해양으로까지 넓혀지고 있다. 하지만 전자해도 등의 해양 지리정보는 선박의 항해와 같은 특수한 목적으로만 사용되어져 왔고 전용의 표시시스템을 사용해야만 하기 때문에 범용적이지 못하다는 단점을 가지고 있다. 또한 기존의 지리정보시스템들은 각각 독자적인 포맷으로 데이터를 표현하고 관리해왔기 때문에 시스템 상호간의 호환성이 떨어지고 데이터의 교환이 용이하지 않았다. 이를 극복하기 위하여 OGC에서는 지리정보를 효과적으로 표현할 수 있는 XML 기반의 GML 표준을 발표하였고, W3C에서는 벡터 그래픽의 표준인 SVG를 제안하였다. 이에 본 논문에서는 선박의 항해에만 제한적으로 사용되던 전자해도를 다양하게 활용하고 비전문가도 이용할 수 있도록 GML과 SVG에 기반한 전자해도 관리시스템을 개발하였다. 즉, GML로 표현된 전자해도를 데이터베이스화하고 사용자의 질의에 부합하는 전자해도를 검색하여 이를 SVG로 변환하여 웹 브라우저를 통하여 서비스를 제공할 수 있는 시스템을 설계하고 구현하였다.

핵심용어 : S-57, ENC, GML, SVG, XSLT, 지리정보 시스템

ABSTRACT : *With increasing Internet use and evolving network services, geographical information is widely used in various systems, including marine and coastal geographical information systems. Although marine geographical information such as Electronic Navigational Chart (ENC) has been applied to navigation of ships successfully, there is a drawback that it can be only used through some specialized systems. Also, it is difficult to keep compatibility to interchange geographical data among the systems because some existing geographical information systems represent and manage them in specific format. To overcome these limitations, OGC published GML standard based on XML to represent geographical information efficiently and W3C proposed vector graphic standard called SVG. In this paper, we design and implement a Web-based system using the GML and SVG technologies for the services of Electronic Navigational Charts through the Web. The system first accepts ENCs coded in S-57 format, and then adopts GML as a coding and data transporting mechanism, database as a storage and query system, and SVG as a visualization means on the Web. The proposed system can be easily used by ordinary people, while the existing systems are mainly used for specific purpose like the navigation of ships.*

KEY WORDS : S-57, ENC, GML, SVG, XSLT, Geographical Information Systems

1. 서 론

지리정보의 효율적인 관리를 위한 지리정보 시스템 (GIS: Geographic Information System)은 다양한 방법을 통하여 서비스되어 지고 있다. 하지만 다양한 형태의 지리정보 데이터는 표준화되지 못했고, 이를 표현하는 방법 또한 독자적인 그래픽 저작 도구에 의존하고 있다. 이에 따라 지리정보 시스템 간에 공유할 수 있는 표준화된 지리 데이터와 특정 장치의 호

* kamsc@ohsunghq.co.kr

** hcpark@hhu.ac.kr 051)410-4573

환성에 구애받지 않고 정교한 그래픽을 구현할 수 있는 그래픽 표준이 필요하게 되었다.

이러한 요구사항을 만족시키기 위해 OGC (OpenGIS Consortium)는 표준화된 지리 정보 표현을 위한 GML (Geography Markup Language)을 제안하였고, W3C (World Wide Web Consortium)는 그래픽에 대한 논리적인 구조를 기술하고 비트맵의 단점을 극복할 대안으로 벡터 기반의 SVG (Scalable Vector Graphics)를 제안하였다.

본 논문에서는 GML로 표현된 전자해도 데이터를 XSLT (eXtensible Stylesheet Language Transformations)를 이용하여 SVG 문서로 변환한 후 이를 웹 브라우저를 통해 서비스할 수 있는 시스템을 설계 및 구현한다.

2. 관련 연구

본 장에서는 S-57 전자해도 표준 포맷의 구조에 대해서 설명하고, 지리 정보 표준 교환 포맷으로 제안된 OGC의 GML에 대해서 살펴본다. 그리고 2차원 그래픽을 표현하기 위한 W3C의 XML 그래픽 표준인 SVG에 대해서 살펴보고, XML 문서를 다른 형식의 문서로 변환하기 위한 W3C의 XSLT에 대하여 알아본다.

2.1 S-57

S-57은 전자해도 등 해양 데이터의 표현 및 교환을 위한 표준으로서 국제수로기구 (IHO: International Hydrographic Organization)가 2000년 11월에 버전 3.1을 공표하였다. S-57은 지도상의 등대, 항구 등과 같은 특징(Feature) 객체 및 위치 표현을 위한 공간(spatial) 객체로 구성되며, 각 객체는 객체식별자(identifier)와 에트리뷰트(attribute)로 구성된다[1].

현실세계의 실체, 즉 객체는 피쳐 객체(feature object)와 공간 객체(spatial object)로 나누어진다. S-57에서는 실세계의 객체를 피쳐 정보와 공간 정보로 표현하며, 피쳐 정보와 공간 정보는 피쳐 객체와 공간 객체로 표현된다. 피쳐 정보는 해당 객체를 식별하기 위한 식별자와 점, 선, 면의 지형속성을 갖는다. 또한 피쳐 정보는 객체 간의 주종 관계(master-slave), 참조하는 공간 정보에 대한 포인터, 다국적 언어를 위한 속성값을 포함한다. 공간 정보는 피쳐 정보가 참조할 수 있도록 식별자와 공간 정보에 필요한 속성값을 갖는다. 피쳐 정보는 공간 정보의 유무에 관계없이 존재할 수 있으나 공간 정보는 반드시 피쳐 정보와 관련을 가져야 한다.

S-57은 기본적으로 2진(binary) 형식으로 표현되기 때문에, 사용하는 목적에 따라 적절한 형태의 포맷으로 변환하여야 한다. 본 논문에서는 S-57 데이터를 GML 문서로 변환한 후 데이터베이스에 저장하는 방식을 채택하였다.

2.2 GML

GML은 웹 환경에서 지리공간 정보의 저장 및 전송을 위해 데이터를 구조화된 문서인 XML로 표현한다. OGC에서는 2000년 5월에 버전 1.0이 발표된 이후로, 2001년에 XML 스키마에 기반을 둔 GML 2.0을 제시하였으며, 2003년에 26개의 스키마를 가진 GML 3.0을 채택하였다[2].

GML 3.0은 피쳐 스키마와 지오메트리(geometry) 스키마를 기본 스키마로 하고, 이 외의 부가적인 기능을 하는 스키마들이 참조된다. 26개의 코어(core) 스키마는 어플리케이션에서 바로 참조할 수 없다. 따라서 코어 스키마의 구조를 어플리케이션에 적합하도록 재정의한 것이 응용 스키마이다.

피쳐는 도로, 강, 사람, 운송 수단, 행정 경계와 같은 의미를 가지는 객체로서, 피쳐 스키마는 GML 피쳐나 피쳐 콜렉션(feature collection)을 생성하기 위한 프레임워크를 제공하는 스키마이다. 지오메트리 스키마는 공간 데이타를 표현하기 위한 스키마로서 지오메트리 타입 요소 및 에트리뷰트를 정의하고 있으며, 모든 지오메트리 요소는 (x, y)로 표현되는 좌표 참조 기법에 기반하여 피쳐의 위치 정보를 표현한다. GML 응용 스키마는 특정 영역이나 분야에 맞게 실제 GML 문서에서 참조할 XML 스키마이다. GML에서 제공하는 기본 스키마들을 토대로 GML 문서의 구조를 정의함으로써 XML 문서를 GML로 해석할 수 있는 기반을 제공한다. GML 응용 스키마는 GML 응용 스키마 생성 규칙을 따르는 W3C의 XML 스키마로 표현된다.

2.3 SVG

SVG는 XML을 기반으로 한 그래픽 표준 언어로서 2차원 그래픽을 표현하기 위해 W3C에 의해 2003년 1월에 권고안이 발표되었다[3]. SVG는 웹 관련 기술로 해상도에 관련 없이 변형 없는 확대와 축소가 가능하고 XML의 문법으로 기술되기 때문에 무한한 확장성을 가지고 있다. 또한 SVG는 다른 XML 문서에 의해 참조될 수 있으며 다른 SVG 그래픽 내부에도 포함될 수 있다. SVG는 벡터 그래픽 형태, 이미지, 텍스트의 3가지 그래픽 객체 타입을 가지며, 각 그래픽 객체들은 이미 표현된 객체를 그룹화하고, 스타일을 적용할 수 있으며, 변형 또는 합성할 수 있다. 또한 XML의 장점을 모두 수용하며, SMIL (Synchronized Multimedia Integration Language), GML 등 다른 XML 기반 언어들과 결합시켜 다양한 웹 어플리케이션으로 응용할 수 있다. 또한 SVG는 벡터 기반의 문서나 이미지를 웹 브라우저에서도 그대로 읽을 수 있으며, 일반 벡터 그래픽의 장점을 그대로 살려 해상도에 따른 픽셀 변화에도 전혀 영향을 받지 않기 때문에 방대한 양의 이미지를 표현하기에 적합하다.

2.4 XSLT

우리가 사용하는 브라우저들은 HTML(Hypertext Markup Language)과 같이 이미 정의된 태그를 해석하여 출력하게 된다. 그러나 HTML과 달리, GML은 미리 정의된 태그

(predefined tag)를 사용하지 않고, 사용자에 의해 만들어진 태그를 사용한다. 이러한 GML의 특성으로 인해 브라우저는 GML 문서를 어떻게 출력해야 하는지 알 수가 없다.

그래서, GML 문서를 표현하기 위해 GML 문서가 어떤 방식으로 출력 되어야 하는지를 결정하기 위한 메커니즘이 필요하다. 이러한 메커니즘의 하나로 CSS(Cascading Style Sheet)와 XSL(eXtensible Stylesheet Language)이 있다. 이 중 XSL은 HTML과 함께 사용되는 CSS보다 훨씬 복잡하고 다양한 기능을 지원한다.

XSLT는 XML 문서의 변환(transformation) 기능을 수행하는 언어로 입력 문서에 스타일시트 문서의 기술된 명령(instruction)을 적용함으로써 다른 문서 형식으로 변환하거나 재구성할 수 있다[4].

3. 시스템 설계

그림 1은 본 논문에서 제안하는 시스템의 구성도이다. 본 논문에서는 시스템 구현시 데이터베이스 서버로 MS SQL 서버를 사용하였고, 웹 서버로는 MS Windows 2000 Server를 사용하였다. 사용자 인터페이스는 MS Visual Studio .NET의 웹 폼(Web Form)을 사용하여 작성되었다.

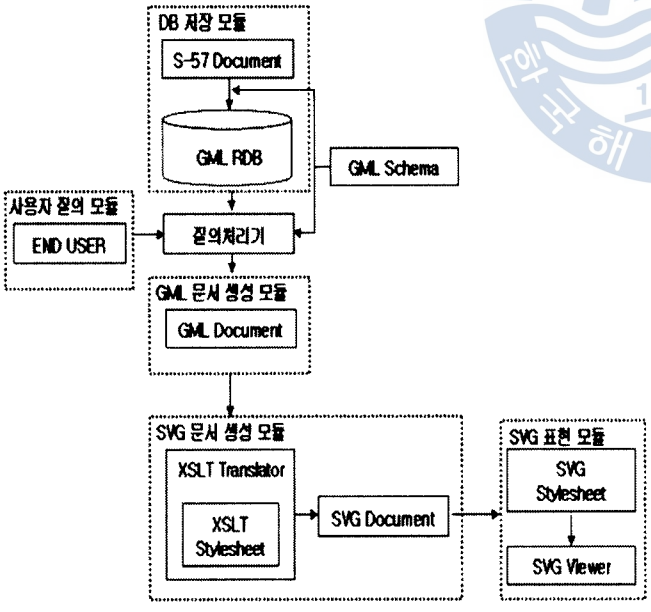

Fig. 1 System Architecture

3.1 S-57 문서의 데이터베이스 저장 모듈

전자해도를 위한 데이터 표준인 S-57은 객체(object)들의 집합으로 구성된다. 이러한 객체는 등대, 항구 등과 같은 특징 객체(feature object)와 그들의 위치를 표현하기 위한 공간 객체(spatial object)로 나누어진다. 각각의 객체는 고유의 식별자(identifier)와 구체적인 데이터를 표현하기 위한 에트리뷰트(attributes)로 구성되며, 대부분의 데이터가 문자열이 아닌 숫자로 인코딩(encoding) 되어 있다[8].

이러한 S-57 전자해도를 GML로 변환하여 데이터베이스에 저장하게 된다. 우선, GML로의 원활한 변환을 위하여 숫자 형식의 코드를 대응되는 문자열로 변환한 후 각각의 객체 단위로 분할한다. 분할된 각각의 객체를 GML 스키마 구조에 맞게 변환한 후 XML 파서(parser)를 통해 데이터베이스에 저장한다.

데이터베이스는 GML 문서의 저장과 검색의 효율성을 고려하여 관계형 및 객체지향형 데이터베이스의 장점을 이용할 수 있는 객체 관계형 데이터베이스를 사용하여 시스템을 구성하였다. 데이터베이스는 단일 테이블로 구성하였으며, 객체 클래스에서 지도 명, 객체 명, 객체 내용 별로 데이터를 추출하여 저장하였다.

3.2 사용자 질의 모듈

사용자 질의는 웹 서버와 데이터베이스 서버, 클라이언트 간의 동작으로 이루어진다. GML 문서가 저장된 데이터베이스에서 정보를 검색하는 방법에는 크게 구조 기반 검색과 내용 기반 검색이 있다[7]. 본 논문에서는 검색 인터페이스를 통하여 지역과 객체 단위의 구조 기반 검색을 웹 서버에 지시한다. 웹 서버는 사용자 질의를 데이터베이스 서버에 SQL (Structured Query Language) 형태로 전달한 후 결과 집합을 돌려 받게 된다. SQL은 데이터베이스에서 데이터를 선택, 추가, 삭제, 갱신하는 등의 역할을 하는 구조화된 질의 언어로서, 본 논문에서는 선택 질의 만을 사용한다. 선택 질의는 다음과 같은 형태를 가진다. SELECT 절은 검색할 필드를 지정하고, FROM 절은 검색할 필드를 포함하는 테이블을 지정하며, WHERE 절은 검색할 필드의 조건을 지정한다.

```
SELECT [field1, field2, ...]
FROM [table1, table2, ...]
WHERE [Condition1, Condition2, ...]
```

3.3 GML 문서 생성 모듈

반환된 결과는 GML의 피쳐 스키마와 지오메트리 스키마를 기반으로 한 응용 스키마를 만족하는 GML 문서로 생성된다. 그림 2는 GML 응용 스키마의 구조를 보여준다. GML 응용 스키마는 피쳐 객체의 집합으로 구성되며, 각 피쳐는 요소 선언부와 타입 선언부로 구분된다. 요소 선언부에서는 피쳐의 타입을 정의하고, 타입 선언부에서는 피쳐를 구성하는 에트리뷰트, 속성 요소, 공간 객체 요소를 정의한다.

생성된 GML 문서에서 점(point)은 위도와 경도 좌표를 가지

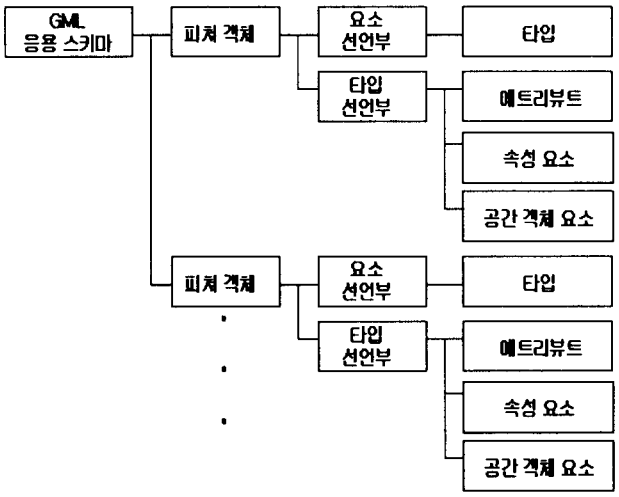

Fig. 2 Structure of GML Application Schema

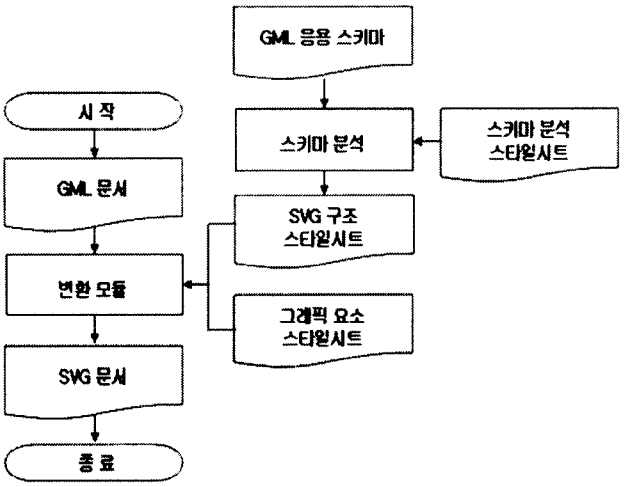

Fig. 3 SVG Document Generation Module

는 위치 정보이며, 선(line)은 점과 점의 연결, 면(area)은 연속된 점들의 연결로 구성된 다각형(polygon)으로 표현된다.

3.4 SVG 문서 생성 모듈

GML 문서를 SVG 문서로 변환하기 위해서는 응용 스키마를 분석하기 위한 스타일시트, GML 문서의 지오메트리 요소를 SVG의 그래픽 요소로 변환하기 위한 스타일시트, GML 문서의 구조적인 변환을 위한 스타일시트로 구성된다. XSLT 변환기는 GML 문서의 요소와 스타일시트에 정의된 변환 규칙을 패턴 매칭(pattern matching)함으로써 SVG 문서를 생성한다. 표 1은 GML 문서의 요소와 SVG 문서의 요소의 대응 관계를 보여준다.

Table 1 GML Elements and SVG Elements

구분	GML	SVG
점	Point	Rect, Circle, Path
선	LineString	Polyline, Path
다각형	Polygon	Path
박스	Box	Rect
심볼	Symbol	Path, Raster image

그림 3은 SVG 문서 생성에 필요한 각 스타일시트 간의 관계를 보여준다. 응용 스키마를 분석하여 생성된 SVG 구조 스타일시트와 그래픽 요소 스타일시트를 변환 모듈이 참조하여 GML 문서를 SVG 문서로 변환한다.

3.5 SVG 표현 모듈

변환된 SVG 문서는 어도비사(Adobe Inc.)의 SVG 플러그인을 통해 사용자에게 보여진다. 현재 버전 3.02인 SVG 뷰어(viewer)는 윈도우와 매킨토시 플랫폼에서 플러그인을 제공하며, 대부분의 웹 브라우저를 지원한다. 또한 플러그인은 출력된 SVG 문서에 대해 자체적으로 줌(zoom), 패닝(panning), 복사, 소스 보기 등의 기능을 제공한다.

현재는 별도의 SVG 뷰어를 설치해야 하지만 XML 기술이 점점 보편화됨에 따라 모든 웹 브라우저에서 SVG 뷰어가 구현될 것으로 예상된다.

4. 시스템 구현

4.1 질의 처리

사용자는 웹 브라우저를 통해 서버에 검색을 요청할 수 있다. 구조 기반 검색, 내용 기반 검색 그리고 두 가지 검색 방식의 혼합인 구조+내용 기반 검색을 지원한다. 다음은 구조+내용 기반 검색의 질의 예를 보여주고 있다.

```
// 'object'의 자식노드로 'name'을 포함하고 그 값이
// 'light'인 object의 ID를 검색
SELECT Object_ID
FROM GML_Content
WHERE PathExp LIKE '%Object/Name%' AND
        Name = 'Light'
```

4.2 검색결과의 GML 표현

데이터베이스에 저장된 전자해도 정보는 사용자 질의에 의해 GML 3.0 코어(Core) 스키마와 응용 스키마를 따르는 GML 인스턴스(Instance) 문서로 표현된다.

그림 4는 작성된 GML 문서의 일부로서 오브젝트 L130_998, L130_999의 위치 정보는 선으로, 오브젝트 P110_1은 점으로 표현된 예를 보여주고 있다.

```
<gml:element>
  <gml:LineString gml:id="L130_998">
    <gml:pos>-32.493122 60.963369</gml:pos>
    <gml:pos>-32.493015 60.963681</gml:pos>
  </gml:LineString>
</gml:element>
<gml:element>
  <gml:LineString gml:id="L130_999">
    <gml:pos>-32.493015 60.963681</gml:pos>
    <gml:pos>-32.493043 60.963831</gml:pos>
    <gml:pos>-32.493172 60.963892</gml:pos>
  </gml:LineString>
</gml:element>
<gml:element>
  <gml:Point gml:id="P110_1">
    <gml:pos>-32.477568 60.961217</gml:pos>
  </gml:Point>
</gml:element>
```

Fig. 4 GML Instance Document

4.3 XSLT를 이용한 SVG 문서로의 변환

XSLT는 XSL에 정의된 규칙과 대상 GML 문서의 엘리먼트를 매칭시켜 SVG 문서를 생성한다. 대표적으로 GML의 엘리먼트가 SVG의 엘리먼트로 변환될 때 'Box' 엘리먼트는 'rect'로, 'LineString'이나 'Polygon'은 'path'로 변환된다.

그림 5는 GML 문서에서 다각형에 해당하는 gml:Polygon, 선에 해당하는 gml:LineString, 점에 해당하는 gml:Point 요소와 스타일시트의 서식이 매치되면 스타일시트에 선언된 명령을 수행하여 그래픽 요소를 추출하도록 하였다. 추출된 객체의 ID와 좌표 정보는 SVG 요소의 형식에 맞춰 변환된다. 그림 6은 GML 문서에서 객체 단위로 그래픽 요소 추출 스타일시트를 반복적으로 적용하여, 그래픽 요소의 배열을 재정의함으로써 그래픽 요소가 출력되는 순서를 지정하는 SVG 구조 스타일시트이다.

```
<!--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>

<!--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>

<!--Point-->
<xsl:template match="gml:Point">
  <xsl:param name="ID"> </xsl:param>
  <xsl:element name="rect">
    <xsl:attribute name="x">
      <xsl:value-of select="substring-before(normalize-
      space(gml:pos),' ')"/>
    </xsl:attribute>
    <xsl:attribute name="y">
      <xsl:value-of select="substring-after(normalize-
      space(gml:pos),' ')"/>
    </xsl:attribute>
    <xsl:attribute name="width">0.001</xsl:attribute>
    <xsl:attribute name="height">0.001</xsl:attribute>
  </xsl:element>
</xsl:template>
```

Fig. 5 Stylesheet for Graphic Element Extraction

```
<xsl:element name="g">
  <xsl:attribute name="id">Light</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:elemen>
```

Fig. 6 Stylesheet for SVG Structure Transformation

4.4 SVG 문서의 웹 브라우징

SVG 문서로 변환된 검색 결과는 어도비 플러그인을 통해 웹 브라우저에 표시된다.

그림 7은 등대 객체에 대해서 변환된 SVG 문서가 웹 브라우저를 통해 사용자에게 출력된 결과를 보여준다. 변환된 SVG 문서는 SVG 플러그인을 통해 사용자에게 보여지며, 검색된 GML 문서와 변환된 SVG 문서도 정보로 제공한다.

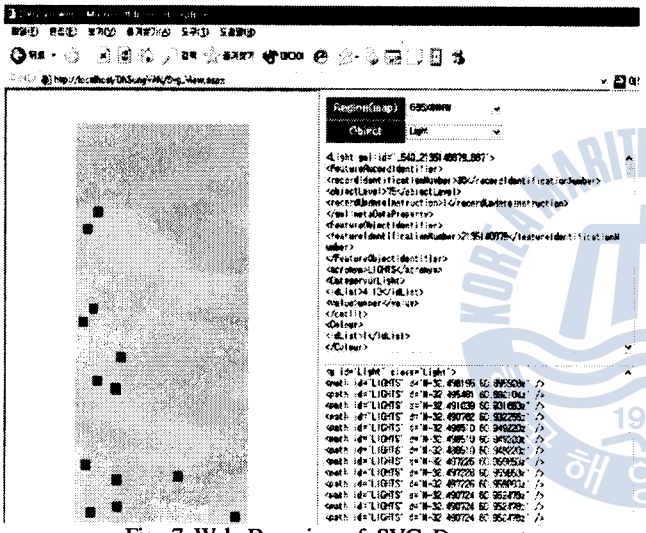

Fig. 7 Web Browsing of SVG Document

5. 결 론

인터넷의 발전으로 지리정보의 공유가 가능해짐에 따라 웹을 통한 서비스가 용이하도록 표준화된 시스템의 개발이 요구되고 있다. 이러한 요구에 따라 본 논문에서는 S-57, GML, SVG 등 3가지 표준을 따르는 전자해도 시스템을 설계하고 구현하였다.

S-57 표준을 따르는 전자해도를 데이터베이스로 구축하여 전자해도의 효과적인 관리와 검색이 용이하도록 하였다. 검색된 전자해도를 더욱 구조적으로 표현할 수 있는 표준인 GML로 재구성함으로써 다른 지리정보시스템과의 정보 교환을 용이하도록 하였다. 또한 재구성된 GML 문서를 XSLT를 이용하여 벡터 기반의 그래픽 표준인 SVG로 변환하였다. 이렇게 변환된 SVG는 기존의 레스터(Raster) 그래픽이 가지는 단점을 극복할 수 있고 웹에서 브라우징이 가능하게 된다.

향후에는 보다 빠른 검색 방법과 개선된 사용자 인터페이스를 제공할 수 있도록 연구하고, 전자해도 데이터의 갱신에 따른 데이터베이스의 수정과 다른 지리정보시스템과의 연계방안이 연구될 필요가 있다.

참 고 문 헌

[1] IHO, IHO Transfer Standard for Digital Hydrographic Data Version 3.1, http://www.iho.shom.fr, Nov. 2000.

[2] OpenGIS, Geography Markup Language (GML) Version 3.0, http://www.opengeospatial.org, Jan. 2003.

[3] W3C, Scalable Vector Graphics (SVG) Version 1.1, http://www.w3.org/TR/SVG11, Jan. 2003.

[4] W3C, XSL Transformations (XSLT) Version 2.0, http://www.w3.org/TR/xslt20, Apr. 2005.

[5] W.T.M.S.B.Tennakoon, Visualization of GML data using XSLT, International Institute for Geoinformation Science and Earth Observation, Feb. 2004.

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

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

[8] 한국전산원, "지리공간 정보 엔코딩(Encoding) 표준 개발에 관한 연구", Nov. 2002.

원고접수일 : 2005년 12월 30일
원고채택일 : 2006년 1월 3일