

공학석사 학위논문

S-57 전자해도의 GML 및 SVG 변환 기법에 관한 연구

A Study on Transforming S-57 ENC into GML and SVG

지도교수 박 휴 찬

2007년 8월

한국해양대학교 대학원

컴퓨터공학과

유 흥 섭

본 논문을 유홍섭의 공학석사 학위논문으로 인준함

위원장 공학박사 김 재 훈



위 원 공학박사 이 장 세



위 원 공학박사 박 휴 찬



2007년 6월 29일

한국해양대학교 대학원

컴퓨터공학과

목 차

그림 목차	ii
표 목차	iii
Abstract	iv
제 1 장 서론	1
제 2 장 관련 연구	3
2.1 전자해도	3
2.2 GML 스키마와 응용 스키마	12
2.3 XML 데이터베이스	15
2.4 XML 관련 기술	20
제 3 장 GML 기반의 전자해도	26
3.1 S-57 의 GML 변환	27
3.2 변환된 GML 문서의 XML 데이터베이스 저장	33
3.3 클라이언트 뷰어	36
제 4 장 전자해도의 GML 및 SVG 변환 기법	39
4.1 변환 시 유의할 점	39
4.2 카달로그 파일의 작성	47
4.3 질의 처리	52
제 5 장 결론 및 향후 연구 과제	60
참고 문헌	62

그림 목차

그림 2.1 S-57 전자해도, GB5X01NW.000의 실제 표현.....	5
그림 2.2 S-57 객체 클래스 다이어그램.....	8
그림 2.3 Galdos.inc의 GML 응용스키마 구조.....	15
그림 3.1 시스템 구조.....	26
그림 3.2 S-57 전자해도의 GML 문서 변환 과정.....	28
그림 3.3 S-57을 GML로 변환하는 알고리즘.....	29
그림 3.4 변환된 GML 문서(Light 객체 클래스 부분).....	32
그림 3.5 GML 저장 구조.....	34
그림 3.6 XML 데이터베이스 저장 모듈 순서도.....	36
그림 3.7 클라이언트 뷰어의 형태.....	37
그림 4.1 GB5X01NW.000 실제 전자해도의 표현.....	41
그림 4.2 -90도 회전된 SVG 문서 GB5X01NW.svg 표현.....	41
그림 4.3 셀 외곽의 빈 공간.....	42
그림 4.4 No coverage available 타입의 커버리지 객체.....	43
그림 4.5 Coverage available 타입의 커버리지 객체.....	44
그림 4.6 S-57 카달로그 파일.....	48
그림 4.7 GML 카달로그.....	51
그림 4.8 질의 처리 과정.....	52
그림 4.9 위경도 질의 처리 과정.....	54
그림 4.10 사용자 질의 XQuery 문.....	55
그림 4.11 사용자 질의 결과.....	56
그림 4.12 다중 레이어 표현.....	57
그림 4.13 다중 레이어를 표현하고 있는 SVG 문서.....	58

표 목차

표 2.1 전자해도의 특징	6
표 2.2 항해목적별 분류	6
표 2.3 S-57 파일이름 구조	7
표 2.4 특징객체의 네 가지 형식	9
표 2.5 각 객체 클래스가 포함하고 있는 정보.....	9
표 2.6 실세계 모델 요소들과 데이터 구조 요소들 간의 관계.....	11
표 2.7 데이터 구조의 특징.....	11
표 2.8 Galdos.inc의 GML 응용 스키마 구성	14
표 2.9 XML 문서 저장 시스템의 비교	17
표 2.10 특징에 따른 XML 데이터베이스의 비교	18
표 2.11 XIS 특징	20
표 2.12 XSLT의 두 가지 처리 과정	22
표 2.13 SVG와 다른 그래픽 표준 비교.....	24
표 2.14 SVG 구성 요소.....	25
표 3.1 GML 변환 프로그램의 함수들.....	30
표 3.2 XIS 서버 저장시스템의 구조 요소 설명.....	34
표 3.3 XIS에 저장 가능한 파일 종류.....	35
표 3.4 사용자 질의 요소들	38
표 4.1 S-57 전자해도의 viewBox 좌표	47
표 4.2 기준점 (0,0)으로 변경된 viewBox.....	47
표 4.3 S-57 카달로그 파일의 요소들	49
표 4.4 GML 카달로그 파일 요소들.....	50

A Study on Transforming S-57 ENC into GML and SVG

Hung-Sub Ryu

Department of Computer Engineering
Graduate School, Korea Maritime University, Busan, Korea

Advised by Hyu-Chan Park

Abstract

S-57 is the international standard of Electronic Navigational Charts (ENCs) which is published by the International Hydrographic Organization (IHO). Although S-57 ENCs are standardized from paper charts, there is no easy way to access S-57 ENCs through a Web browser such as Internet Explorer. Several studies proposed Web-based systems for accessing ENCs, which use the Geography Markup Language (GML) and the Scalable Vector Graphics (SVG). However, the studies have lossess for searching time because of a very large capacity of GML documents in databases. Besides, the studies are impossible to search by coordinates of latitude and longitude such as GPS position data because objects of GML ENCs are queried

generally.

In this thesis, we propose two methods to cope with these problems. First, we propose a method for building catalogue files coded in XML in order to reduce search time and to query a coordinate of latitude and longitude. The catalogue files are retrieved by a query as a coordinate and then a query processor extracts relevant GML documents from the catalogue files. Second, we propose a method for transforming S-57 into GML and SVG. The GML documents extracted through the first method are transformed into SVG documents, which are viewed on the user interface. As the result, Web-based systems for ENC's are developed easily and the transformation of S-57 ENC's into SVG documents makes the systems stable and efficient to access from Internet.

제 1 장 서론

기존의 전자해도(ENC: Electronic Navigational Chart) 시스템과 포맷들은 다양한 분야에서 사용하기 보다는 전문적인 분야에서 주로 사용되어 왔고 나라별로 서로 다른 포맷의 전자해도가 작성되어 국제 표준이 어려운 문제점이 있었다. 국제수로기구(IHO: International Hydrographic Organization)에서는 국제적 표준으로 S-57 전자해도를 발표하였고 이로써 각 나라별로 같은 규칙과 포맷으로 전자해도를 개발할 수 있는 장이 구축되었다[1]. 또한 W3C(World Wide Web Consortium)의 XML(eXtensible Markup Language)처럼 다양한 플랫폼에서의 표준화 시스템이 가능한 마크업(Markup) 기술들이 발표되었고, 지리정보 시스템의 표준화를 원활하게 하기 위해 OGC(Open Geospatial Consortium)가 XML 기반의 GML(Geography Markup Language) 기술을 표준으로 발표하여 전자지도뿐 아니라 전자해도에도 활성화를 일으켰다[2,3].

S-57과 관련하여 이미 많은 시스템들이 개발되었지만 전자해도의 경우 아직도 전문 분야에서 종사하는 사람들만이 전자해도를 주로 다루고 있고 시스템 자체도 고가의 정책을 유지하고 있다. 그러나 인터넷의 발달은 일반인들로 하여금 전문 분야에 대한 이해를 높이고 다양한 분야로의 발전이 요구되는 만큼, 전자해도 또한 다양한 분야에서 연구가 이뤄지고 있으며 GPS의 보급이 활성화 되면서 낚시터나 바다 휴양지처럼 일반인들의 접근이 가능한 해양영역으로의 길 안내 등과 같은 시스템들이 개발 중이다[4].

이런 시스템들에 적용하는 하나의 방법으로 XML을 활용하는 방법이 연구되고 있다 [4]. 여러 플랫폼으로의 확장과 재활용성이 높은 XML S-57 전자해도를 GML, SVG

(Scalable Vector Graphics), XSLT(eXtensible Stylesheet Language Transformations), XQuery, XPath 등의 XML 기반 기술들만을 사용하여 표현한다[5-8]. 이를 위해서는 몇 가지 변환 기법이 요구된다.

본 논문에서는 XML 전자해도 시스템에서 사용자로부터 하나의 위경도 좌표를 질의로 입력 받아, 위경도 좌표 질의가 전자해도 뷰(View)의 중점이 되는 SVG 문서를 생성해내는 과정을 다룬다. 전체 과정에서 일어나는 변환 과정들과 위경도 질의가 XQuery와 XPath를 통해 데이터베이스의 GML 카탈로그에 포함된 셀 파일들의 정보들을 검색하고 처리되는 과정들을 제안한다. 또한 S-57 전자해도로부터 변환된 GML 문서에서 질의에 맞는 문서를 추출하여 SVG 문서로 변환될 때 S-57 위경도 좌표와 SVG의 스크린 좌표간 이상 현상과 같은 몇 가지 유의할 점과 해결방안을 제안하여 XML 전자해도 시스템의 쉬운 접근과 방향성을 제시한다.

본 논문의 2장에서는 S-57 국제표준 전자해도, GML 지리 정보 언어, XML 저장 데이터베이스, SVG 벡터 그래픽 표준 등 XML 관련 기술들에 대하여 설명한다. 3장에서는 S-57 전자해도의 GML로의 변환과 변환된 GML 문서들의 저장 방법, 클라이언트 뷰어의 형태들을 설명하고, 4장에서는 변환 시 유의할 점들을 제시하여 변환 과정에서 처리해야 할 요소들을 살펴보고 GML 카탈로그의 생성과 XML 기반 전자해도 시스템에서 위경도 좌표 질의가 처리되는 과정과 방법들을 제안한다.

제 2 장 관련 연구

2 장에서는 IHO에서 제안하고 있는 전자해도 S-57의 데이터 모델과 구조에 대해 설명하고, XML 기술을 이용하여 S-57 전자해도를 일반 웹 브라우저를 통해 접근할 수 있도록 하기 위해 OGC에서 제안하는 GML, XSLT, XPath, XQuery와 XML 문서를 저장하는 전문 데이터베이스 등 XML 기반 표준화 기술들의 구조 및 특징들에 대해 알아본다.

2.1 전자해도

해양지도는 해상교통과 해양자원조사, 해양연구, 어업관련 등 많은 곳에서 전문적으로 사용되어 왔다. 그러나 해양에 대한 접근 범위가 커지면서 일반인들의 해도 사용과 국제적 표준 해도의 사용이 불가피해 해도의 디지털화가 진행되었고, 이 연구가 시작된 이후로 S-57이라는 국제 표준 전자해도 포맷이 국제수로기구에 의해 작성되었다 [1,4]. 즉, S-57은 국제수로기구에서 작성한 전세계 전자해도 표준의 명칭이다.

(1) 국제 표준 전자해도 S-57

바다를 이용하는 기술의 발달과 지구온난화 등으로 해양에 대한 인적 위험성과 경제적 손실, 해양환경오염 피해가 증가하면서 이에 대한 방안으로 국제수로기구와 국제해사기구(IMO: International Maritime Organization)에서 1990년대 초반부터 전자해도를 연구하기 시작했으며, 1996년 전자해도 제작 국제기준 S-57 3 버전을 발표했다[1].

S-57은 전자해도 제작에서 국제적 기준으로 참조할 수 있는 표준으로서 각국의 정부기관에 의해 전자해도가 제작된다. 우리나라의 경우 국립해양조사원에서 2000년부터 3.0 버전을 시작으로 현재는 S-57 3.1 버전을 표준으로 제작, 서비스하고 있는 중이다.

(2) S-57 ENC(Electronic Navigational Chart) 데이터

S-57 전자해도 데이터를 표현하거나 이용하는 기존 방식들은 모두 고가의 전용 장비나 브라우저를 통해 가능할 뿐 아니라 S-57 전자해도 데이터 그 자체도 각 나라별로 고가의 제품으로 판매되고 있으며, 실제 전세계의 데이터를 갖추기 위해서는 고가의 비용을 지불해야만 한다. 그림 2.1은 GB5X01NW.000 데이터를 무료 제공되고 있는 HydroService의 dKart Look 4.0 뷰어*를 통해 표현한 것이다. 본 논문의 S-57 데이터들은 실제 존재하지 않는 가상의 해양지도로서 테스트만을 위해 IHO에서 제공하는 총 6개의 전자해도 데이터들로서 S-57에 속한 대부분의 객체들을 포함한다.

* <http://www.hydroservice.no>

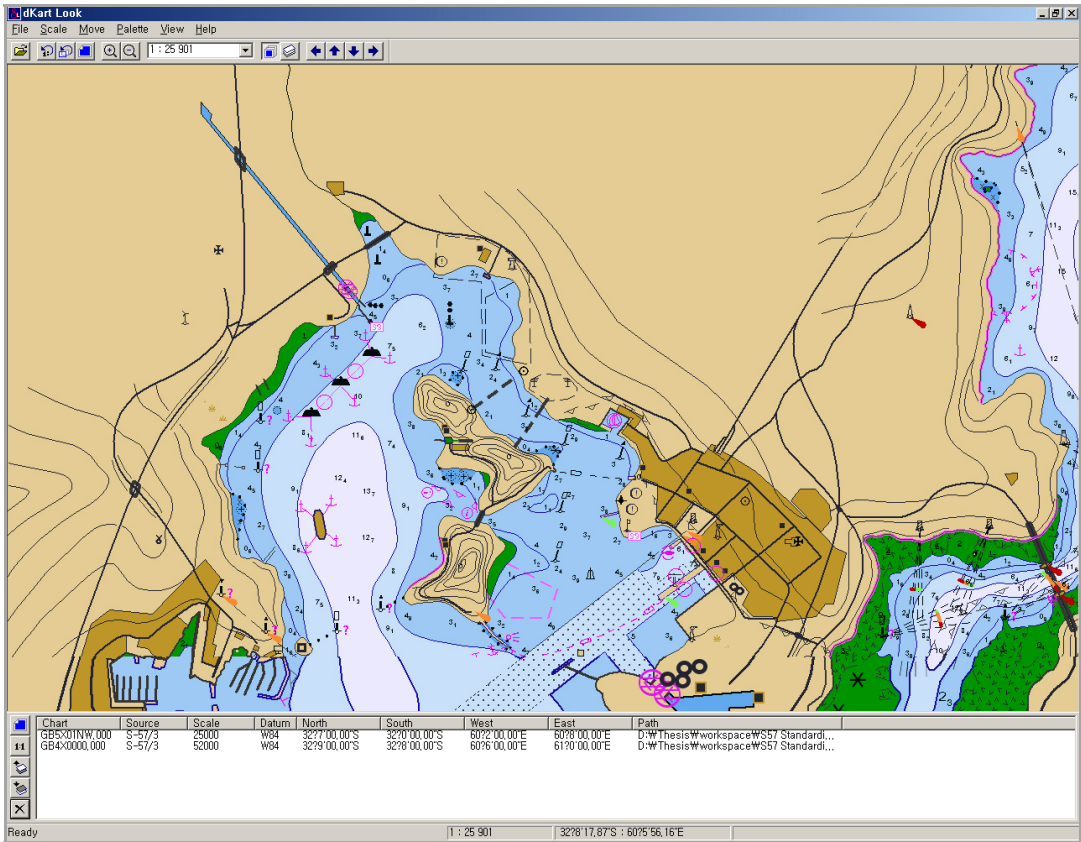


그림 2.1 S-57 전자해도, GB5X01NW.000의 실제 표현
 Fig. 2.1 The real representation of S-57 ENC, GB5X01NW.000

S-57 ENC 파일은 바이너리(Binary)로 구성된 아스키(ASCII) 형태로 구성되어 있다. 각 파일마다 축적(Scale)을 가지고 있으며 모든 파일들은 직사각형으로 구성된 셀(Cell) 형태로 표현된다. 특히 S-57은 위도, 경도를 사용해 셀과 위치를 표시하고 거리는 해상 마일로, 수심과 높이는 미터를 사용한다. 아래의 표 2.1과 2.2는 한국해양개발(주)*에서 작성한 한국해양지도 S-57의 특징과 항해목적별 분류 및 분류별 축적을

* <http://www.chartkorea.com>

나타낸다.

표 2.1 전자해도의 특징

Table 2.1 The descriptive characteristics of ENC

전자해도의 특징	
1.	각 셀은 직사각형으로 구성되고 위도, 경도로 표시한다.
2.	하나의 셀은 자료크기가 5M bytes 이하이어야 한다.
3.	각 셀은 항해목적별로 6가지로 구분한다.
4.	항해목적에 같은 셀은 서로 중복이 되어서는 안된다.
5.	수평좌표계는 WGS-84 기준점을 사용한다.
6.	어떠한 투영법도 적용하지 않는다.
7.	위치는 위도, 경도로 표시한다.
8.	거리는 해상 마일로 표시한다.
9.	수심과 높이의 단위는 미터를 사용한다.

표 2.2 항해목적별 분류

Table 2.2 The catalog from navigation-related information

부여번호	항해목적분류	축척 범위
1	총도 (Overview)	500,000 ~
2	항양도 (General)	100,000 ~ 499,000
3	해안도 (Coastal)	50,000 ~ 99,999
4	항만접근도 (Approach)	25,000 ~ 49,999
5	항박도 (Harbour)	3,000 ~ 24,999
6	계류도 (Berthing)	~ 2,999

S-57 ENC 파일은 해당 셀이 가진 특징을 파일 이름에 포함한다. 예를 들어 부산 영도구와 부산항을 포함하고 있는 KP520100.000 ENC 파일은 첫 번째 두 알파벳에서 KP라는 우리나라 국가 코드를 표시하고 세 번째 번호 5는 항해목적분류의 1~6까

지의 부여번호 중 축적 3,000에서 24,999에 속하는 항박도를 표시한다. 그 뒤 다섯 자리 숫자 중 앞 세 자리는 전자해도 제작 국가가 부여하는 번호로 우리나라의 경우 국립해양조사원*에서 세 자리 수로 되어 있는 기존 종이해도 번호를 그대로 사용하기로 정해 종이해도 번호 201을 전자해도 번호에 부여한다. 따라오는 두 숫자는 전자해도 번호이지만 조금 다르게 쓰인다. 우리나라 종이해도 번호 방식에서 실제 201 앞에는 W나 E, FW 등이 붙어 있고 세 자리 숫자 뒤에 연관된 해도에 대해 알파벳 하나가 붙거나 '-1' 등으로 표시한다. 이런 관계에 대해 또는 하위 해도에 대한 번호를 마지막 두 자리 숫자가 표현한다. 맨 뒤의 확장자 표시 000은 실제 S-57 ENC 데이터의 확장자이면서 업데이트 번호를 표현한다. 000은 초기 파일이며 001, 002처럼 새 버전에 대한 번호가 붙는다. 표 2.3은 S-57 전자해도 파일 이름의 구조와 실제 부산항 서부 전자해도 파일의 예제를 나타내는 표이다[1,7].

표 2.3 S-57 파일이름 구조
Table 2.3 The S-57 file name structure

파일이름 구조 CCPXXXYY.EEE			부산항서부 예제 KP5201B0.002		
이름	길이	내용	이름	길이	내용
CC	2	국가코드	KP	2	대한민국 국가코드
P	1	항해목적별분류 축척 범위	5	1	항박도를 위한 해도 1/10,000
XXX	3	해도 번호	201	3	종이해도 W201
YY	2	하위 번호	B0	2	연관해도 W201B
.	1	파일구분자	.	1	파일구분자
EEE	3	업데이트 번호	002	3	두번 업데이트된 해도

* <http://www.nori.go.kr>

(3) S-57 객체와 속성

그림 2.2에서 나타내는 것처럼, S-57은 실세계의 객체를 특징(Feature) 객체와 공간(Spatial) 객체로 표현한다. 특징 객체는 실세계 엔티티들의 특징과 주종관계 등 지리적 요소들을 포함하지 않은 모든 요소들을 기술하고 공간 객체는 실세계 엔티티들의 위치와 모양에 대한 지리적 요소들을 포함한다. 각 객체는 객체식별자(Identifier)와 속성(Attribute)으로 구성된다는 것을 보여준다. 각 객체들의 특징들, 객체의 속성에 대한 정보, 객체 사용에 관련된 정보 등 특징 객체는 좌표 정보를 가지는 공간 객체와 완전히 구별된 부분이다.

실세계 엔티티에서 위치 정보를 제외한 나머지 정보들을 S-57의 특징 객체로 효과적으로 바꿔주기 위해 S-57은 표 2.4와 같은 네 가지 형식을 제공한다[1].

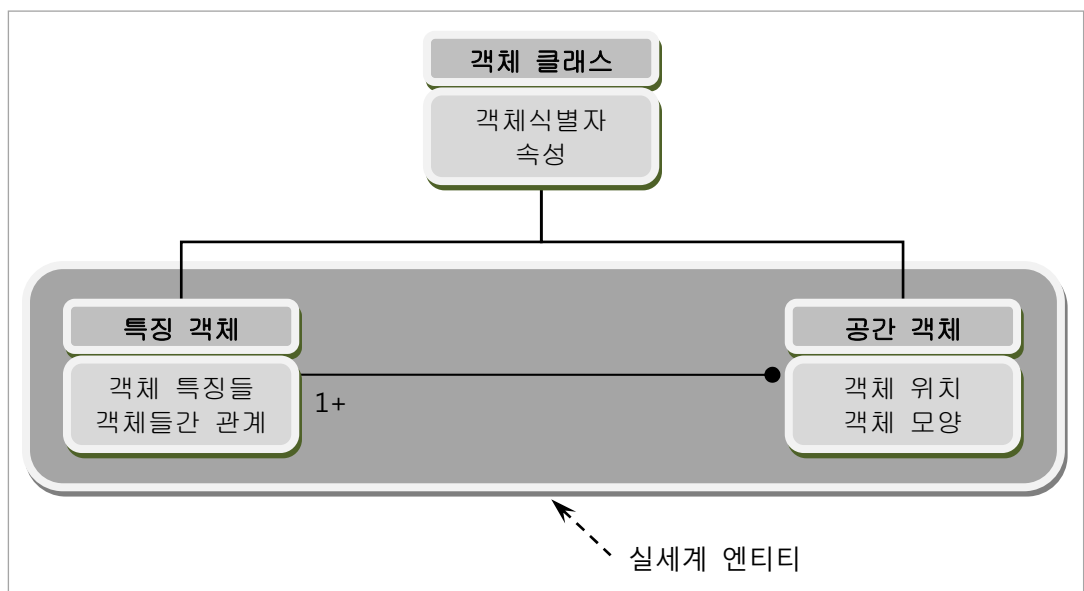


그림 2.2 S-57 객체 클래스 다이어그램
Fig. 2.2 Diagram of S-57 object classes

표 2.4 특징객체의 네 가지 형식

Table 2.4 Four types of feature objects

특징 객체 형식	내용
메타(Meta)	다른 객체들에 대한 정보
지도제작 (Cartographic)	실세계 엔티티들의 텍스트를 포함한 지도제작 표시 관련 정보
지리(Geo)	실세계의 각 엔티티들을 나타내는 특징 정보
컬렉션(Collection)	다른 객체들간의 관계를 표시

모든 형식의 특징 객체들은 객체클래스 이름, 객체의 긴 이름을 알파벳과 특수문자로 구성된 여섯 자리로 줄여 놓은 코드(Acronym), 데이터에서 사용되는 객체의 숫자 코드, 객체클래스의 세 가지 속성 집합을 가진다. 표 2.5는 각 객체 클래스가 포함하고 있는 정보를 나타낸다[1].

표 2.5 각 객체 클래스가 포함하고 있는 정보

Table 2.5 The information containing in each object class

영문 정보 이름	내용
Object Class	객체 클래스 이름
Acronym	객체 클래스의 여섯자리 문자 코드
Code	데이터에서 사용되는 숫자형 코드
subset 'Attribute_A'	객체의 각 특징들
subset 'Attribute_B'	데이터 사용과 관련된 정보
subset 'Attribute_C'	속성이 속한 데이터와 객체들의 관리 정보

공간 객체의 객체 식별자는 특징 객체에서 명시하고 있는 객체 식별자와 연결되고 해당 공간 객체는 Node(점), Edge(선), Face(면)으로 구성된 정보들을 가지며 실제 데이터에서는 Point(점), Line(선), Area(면)으로 표현된다. S-57에서 면은 식별자를 가

지는 선들을 포함하고, 선은 식별자를 가진 점들과 좌표를 포함한다, 즉, 면과 선은 참조 식별자를 가지고 좌표는 선과 점에만 표현된다.

(4) S-57 데이터 구조

논리적인 현실세계를 물리적인 구조로 표현하기 위해서는 규칙이 필요하고 거기에 따라 시스템 상호간에 교환도 가능해진다. 전자해도의 정보를 표현하기 위해서는 먼저 실세계를 모델화하여 단순화시키는 작업이 필요하고 그 결과, 모델은 이름을 가지는 레코드와 필드 등으로 구조화 작업을 한다. 구조화 작업이 끝나면 다른 컴퓨터 시스템과의 교환을 위해 물리적인 변환 표준인 ISO/IEC 8211을 사용해 구조화된 S-57을 압축한다[1].

IHO는 표 2.6과 표 2.7에서처럼 실세계 모델을 데이터 구조로 만들기 위해 모델과 구조 간의 관계를 정립해 놓았고 전자해도의 정보를 표현하고 교환하기 위한 데이터 구조를 네 가지로 요약한다[1].

표 2.6 실세계 모델 요소들과 데이터 구조 요소들 간의 관계

Table 2.6 The relationship between model elements and structure elements

실세계 모델의 요소들	데이터 구조의 요소들
특징 객체	특징 레코드
메타 특징 객체	메타 특징 레코드
지도제작 특징 객체	지도 제작 특징 레코드
지리 특징 객체	지리 특징 레코드
컬렉션 특징 객체	컬렉션 특징 객체
공간 객체	공간 레코드
벡터 객체	벡터 레코드
고립된 점(Node) 객체	고립된 점 벡터 레코드
연결된 점 객체	연결된 점 벡터 레코드
선(Edge) 객체	선 벡터 레코드
면(Face) 객체	면 벡터 레코드
래스터(Raster) 객체	래스터 레코드
매트릭스(Matrix) 객체	매트릭스 레코드
속성	특징 속성 필드나 공간 속성 필드
특징 객체 간 관계	컬렉션 특징 레코드나 포인터 필드
특징객체와 공간객체 간 관계	포인터 필드

표 2.7 데이터 구조의 특징

Table 2.7 Features of data structure

데이터 구조의 특징
하나의 변환 집합(Exchange set)은 하나 이상의 파일을 가진다.
하나의 파일은 하나 이상의 레코드를 가진다.
하나의 레코드는 하나 이상의 필드를 가진다
하나의 필드는 하나 이상의 하위 필드를 가진다.

2.2 GML 스키마와 응용 스키마

HTML과 함께 SGML에서 파생되어온 XML은 표준화 작업의 일환으로서 많은 관심을 불러일으켰을 뿐 아니라 유비쿼터스 시대에 필요한 요소로서 존재하고 있다. 그러나 XML은 프로그램을 만들어내는 개발 도구 언어가 아니기 때문에 실제 사용하기 위해서는 사용자가 직접 작성해야 하는 작업들이 많다. 그 중 지리적 문서에 대한 부분은 개인 수준에서 다루기에 어려운 부분들이 많아 많은 인력이 포함된 프로젝트 수준의 작업이 필요하다. 이에 OGC에서는 XML에 지리적인 부분만을 추가한 GML을 발표하였다[2,8].

(1) GML 스키마

GML은 XML과 마찬가지로 DTD나 스키마를 통해 GML에 대한 구조를 정의한다. GML 3.0은 피쳐 스키마(Feature Schema), 지오메트리 스키마(Geometry Schema), 이 외 부가적인 기능을 하는 스키마로 구성된다[2]. 26개의 코어 스키마는 어플리케이션에서 바로 참조할 수 없어 코어 스키마의 구조를 어플리케이션에 적합하도록 재정의한 것이 다음 절에 설명할 GML 응용 스키마이다.

피쳐(Feature)는 S-57의 특징 객체처럼 건물, 등대, 도로, 강, 운송 수단, 행정 경계 등과 같은 객체를 의미하지만 여기서의 피쳐는 지형이라는 뜻에 가깝다. 피쳐 스키마는 GML 피쳐나 피쳐 콜렉션(Feature collection)을 생성하기 위한 프레임워크를 제공하는 스키마이다. GML에서 피쳐는 항상 XML 요소로 표현되며, 피쳐 특징을 표현하는 특성(Property)들과 속성(Attribute)의 집합으로 기술된다. 피쳐 스키마는 지오메트리 스키마를 포함함으로써 피쳐의 위치 정보를 표현한다.

모든 GML 피쳐 스키마는 `gml:location`과 `gml:boundedBy` 특성을 선택적으로 가진다. `gml:location` 특성은 피쳐의 위치를 정의하고, `gml:boundedBy` 특성은 피쳐 인스턴스(Instance)를 둘러싸는 경계를 정의한다. 그리고 모든 피쳐는 고유한 식별자인 `gml:id` 속성을 가진다.

지오메트리 스키마는 공간 데이터를 표현하기 위한 스키마로서 지오메트리 형식의 요소 및 속성을 정의하며, 모든 지오메트리 요소는 (x, y)로 표현되는 좌표 참조 기법에 기반하여 피쳐의 위치 정보를 표현한다.

모든 지오메트리 요소들은 추상적인 슈퍼타입(Supertype)으로부터 직접적 또는 간접적으로 파생된다. 따라서 지오메트리 요소는 식별자(`gml:id`), 이름(`name`), 설명(`description`)을 가진다.

GML에서는 지오메트리 정보를 표현하기 위해 `Point`, `LineString`, `Box`, `LinearRing`, `Polygon`과 여기서 확장된 `MultiPoint`, `MultiLineString`, `MultiPolygon` 등의 형식을 제공한다.

(2) GML 응용 스키마

GML 응용스키마는 GML 스키마와 S-57 스키마와는 별개로 GML 문서를 작성하기 위해 S-57 표준에 맞는 GML 규칙을 나열한다. GML 응용스키마는 개인적으로 제작하기엔 시간적인 측면과 인력적인 측면에서 어려운 점이 많다[8]. 일반적으로 XML 전자해도 시스템에서 사용되는 S-57 전용 GML 응용스키마는 Galdos.inc에서 제작·발표한 S-57 응용스키마라는 이름의 스키마가 있으며 본 논문에서도 응용스키마로 사용된다[9-11]. Galdos.inc의 S-57 응용스키마는 표 2.8과 같은 네 가지 XSD 문서들로 구성되고 그림 2.3은 XSD 간 관계 구조를 나타낸다[6].

표 2.8 Galdos.inc의 GML 응용 스키마 구성

Table 2.8 The GML application schemas by Galdos.inc

XSD	설명
Objects.xsd	모든 유효한 S-57 전자해도 객체들을 포함한다. (지도제작 관련 객체들과 C_ASOC 객체 제외)
Attributes.xsd	모든 유효한 S-57 전자해도 속성들을 포함한다. (지도제작 관련 속성들 제외)
AbstractAndSuperTypes.xsd	모든 추상적 기반 타입들과 높은 레벨의 특징 컬렉션, 대응하는 메타데이터를 포함한다.
SupportTypes.xsd	S-57에서 지원하는 단순타입을 포함하며 각 타입들은 S-57 스키마 엘리먼트들로 구성되어 있다.

Galdos.inc의 백터레코드는 지오메트리와 토폴로지 모델로 나누며 지오메트리는 노드, 에지, 페이스를 나타내고 토폴로지는 점, 선, 면을 나타낸다. 지오메트리와 토폴로지의 요소들은 xlink:href 참조와 같은 백포인터(Backpointer)를 사용하여 연결한다.

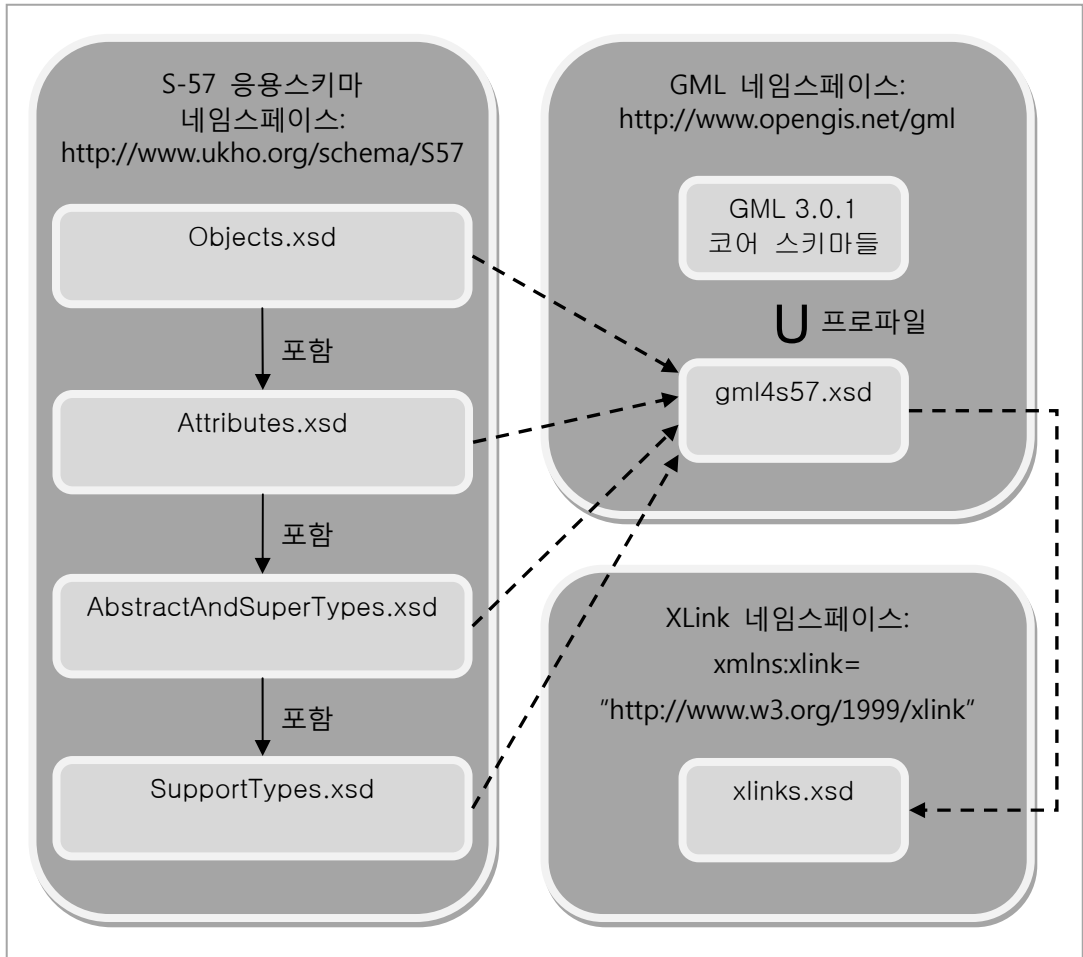


그림 2.3 Galdos.inc의 GML 응용스키마 구조

Fig. 2.3 The structure of GML application schemas by Galdos.inc

2.3 XML 데이터베이스

XML 기술을 사용할 때 가장 중요하면서 기본적인 요소는 저장 시스템의 구조이다. 저장 시스템 구조는 XML 전체 시스템의 속도와 안정성, 보안, 동시성, 비용 등에 큰

영향을 끼친다[11,12].

(1) XML 저장 시스템의 선택

일반적으로 XML을 다루는 저장 시스템이라면 관계형 데이터베이스, 객체지향형 데이터베이스, 그리고 XML 전용 데이터베이스 세 가지 시스템으로 나눌 수 있다. 이 세 가지 데이터베이스는 각각 장점과 단점을 가지고 있으며 XML 기술을 사용하는 목적에 가장 적절한 시스템을 선택하는 것이 선행되어야 할 것이다. 실제로는 파일 시스템 기반의 방식도 있지만 웹 기반의 시스템에서는 입지가 좁아지고 있는 기술이므로 빼도록 한다.

관계형 데이터베이스의 경우, 구조적이지 않은 XML 데이터 모델을 구조적인 관계형 모델로 옮기면서 발생하는 데이터 손실의 문제가 있으며, 큰 크기의 객체(Lob, Large Object) 처리 부분에서 발생하는 파싱 시간 등의 속도에 부하가 있을 수 있다. 그러나 XML 데이터와 관계형 데이터의 동시 사용이 가능하다는 점과 성능상의 문제 해결로 인해 주로 사용되고 있으며 90% 이상의 시장 점유율을 가지고 있어 다양한 분야에서 활용되고 있다[13,14].

객체 지향형 데이터베이스는 XML 데이터 모델과 객체 지향형 모델과 유사하기 때문에 데이터 손실이 없이 XML 그대로 저장할 수 있는 장점이 있지만, 시장 확보율이 낮고 ODMG(Object Database Management Group)의 표준을 확실히 따르지 않고 있는 단점이 있다[15].

XML 전용 저장시스템의 경우는 XML만을 사용하고자 할 때 다른 저장시스템에 비해 효과적이지만, 개발 비용이 많이 들고, 기존 데이터와의 호환이 문제가 될 수 있다 [16]. 표 2.9에서는 관계형, 객체지향형, XML전용, 세 가지의 XML 문서 저장 시스템

을 비교, 분석한다[13-16].

표 2.9 XML 문서 저장 시스템의 비교

Table 2.9 Comparisons on systems to save XML documents

저장시스템	장점	단점
관계형	<ul style="list-style-type: none"> · 시장점유율이 높다 · XML 데이터와 관계형 데이터의 동시 사용이 가능하다 · 성능상의 문제 해결이 쉽다 · 문서의 저장 및 추출이 빠르다 	<ul style="list-style-type: none"> · 반구조적 XML 모델을 구조적 관계형 모델로 사상할 때 데이터손실이 발생할 수 있다 · LOB의 파싱 처리 시간 등의 속도 부하가 발생할 수 있다 · 문서의 부분 수정이나 검색에 많은 시간이 필요하다
객체지향형	<ul style="list-style-type: none"> · XML 모델과 객체지향형 모델이 유사하여 데이터 손실이 없다 	<ul style="list-style-type: none"> · 시장점유율이 낮다 · ODMG표준을 확실히 따르지 않는다
XML전용	<ul style="list-style-type: none"> · XML 메시지를 하나의 처리 단계로부터 다음 처리 단계에 넘겨줄 때 XML 텍스트 파일의 생성과 파싱 과정이 불필요하다 · XML을 사용할 때 가장 효과적인 저장시스템이다 	<ul style="list-style-type: none"> · XML만을 사용할 수 있다 · 개발 비용이 높다 · 기존 데이터와의 비호환 문제가 발생할 수 있다

S-57 전자해도의 GML 변환 문서는 전체 시스템에서 LOB 형태의 큰 크기를 처리해야 하기 때문에 관계형과 객체지향형 데이터베이스를 사용하기에는 무리가 있다. 따라서 본 논문에서는 XML 전용 데이터베이스를 사용해 XML 관련 기술들을 다루도록 하였다. 그러나 대용량의 비구조적인 XML 데이터를 저장하고 가져오기 위해서는 데이터 전송 소프트웨어나 관계형 데이터베이스와 같은 서드파티 미들웨어가 구축되거나, XML 전용 데이터베이스와 별개로 XML 비호환 데이터에 대한 하위 서버를 구축하는 것을 고려해야 한다.

(2) XML 전용 데이터베이스

XML 기술이 많은 분야에 활용되기 시작하면서 업체들과 조직에서 XML 문서를 효과적으로 저장하고 관리할 수 있는 데이터베이스의 필요성을 인식하게 되었다. 기존의 XML 문서는 크게 데이터와 문서 중심의 두 가지 문서로 나눌 수 있는데, 데이터 중심의 XML 문서는 온줄 구매 주문서, 온줄 민원 신청서, 부서간 데이터와 메시지 교환 등과 같은 데이터 전송을 위한 것이다. 문서 중심의 XML 문서는 웹에서 서버의 부하를 줄이기 위한 클라이언트 처리 부분으로서 온줄 프로그램의 설정 등에 사용되거나 학술 분야에서 논문 및 연구보고서를 통합 문서로 다루기 위해 사용되는 등 웹과 관련된 클라이언트 최종 사용자에게 보여주기 위해 작성된 XML 문서를 의미한다. 표 2.10은 특징에 따른 XML 데이터베이스로 나누어 비교한다[6].

표 2.10 특징에 따른 XML 데이터베이스의 비교

Table 2.10 Comparisons on XML databases by their characteristics

XML 호환 데이터베이스	XML 전용 데이터베이스
XML 문서와 문서구조간 데이터 전송 데이터 전송 확장 제공	XML 문서에 대한 논리적 모델을 정의 모델에 따라 XML 문서를 저장 및 추출
데이터 중심의 어플리케이션 데이터 중심의 문서용	논리적 저장단위의 XML 문서 (물리적 저장 모델을 요구하지 않는다.)
IBM DB2 XML Extender Informix Web DataBlade Microsoft XML for SQL Server 2005 Oracle 10g	Software AG Tamino X-Hive/DB eXcelon XIS lpedo

문서중심의 XML 문서를 저장 및 추출하기 위해 XML 데이터베이스 또는 콘텐츠 관리 시스템이 필요하다. 이런 시스템을 제공하는 XML 데이터베이스 어플리케이션은

많은 업체에서 다양하게 제공하고 있는데, 이런 어플리케이션은 특징에 따라 나눌 수 있으며 XML 가능 데이터베이스 와 XML 데이터베이스로 나눌 수 있다.

(3) eXcelon의 XIS(eXtensible information Server)

본 논문에서는 XML 전용 데이터베이스로서 eXcelon의 XIS를 사용한다[16]. XIS는 eXcelon社에서 최근 Sonic社로 바뀌어 현재는 Sonic XML Server라는 명칭으로 제품 이름도 변경되었다. XIS는 데이터 및 콘텐츠의 통합, 관리, 변환 등의 기능과 확장성을 제공하고 산업 표준으로서의 XML 기반으로 통합되고 관리되며 대용량의 XML 정보를 저장하고 다수의 사용자가 사용할 수 있도록 확장성이 고려되어 설계되었다.

XML 데이터베이스의 특징과 함께 XIS는 XML 문서를 DOM 형태로 저장하기 때문에 XML 메시지를 하나의 처리 단계로부터 다음 처리 단계에 넘겨줄 때 XML 텍스트 파일의 생성과 parsing 과정이 불필요하다는 장점을 갖는다. 표 2.11은 XIS의 특징들을 나타낸다[16].

표 2.11 XIS 특징

Table 2.11 XIS features

XIS 특징	
XML 문서를 DOM 형태로 저장	<ul style="list-style-type: none"> • 필요할 때마다 XML을 파싱할 필요가 없어 빠름 • XML 전체뿐만 아니라 일부분의 정보도 추출 가능 • 캐쉬(Cache)에 XML 정보를 올려 빠르게 서비스 • 대용량의 XML 데이터 서비스가 가능
Well-formed XML 문서를 저장	<ul style="list-style-type: none"> • 다양한 구조의 XML 문서를 지원 • 엘리먼트나 속성 단위로 수정 가능 • XML스키마의 수정이 쉬워 스키마 설계와 구현이 빠름 • DTD 및 XML 스키마 승인(Validation)을 지원하여 데이터의 동기화를 유지
산업 표준을 지원	<ul style="list-style-type: none"> • W3C XML 1.0 (네임스페이스 포함) • W3C DOM Level 1, 2 • XPath 1.0 (질의 언어) • XSLT 1.0 (Java 확장 함수 포함) • 세계 언어를 위한 표준 문서 인코딩
빠른 XSLT 기능	<ul style="list-style-type: none"> • XSL도 XML 문서임으로 DOM 구조로 저장 • XSLT 수행 중 재 파싱 없이 변환을 수행 • 캐쉬에 이러한 기능이 분산되어 성능 확장성이 보장
기관 시스템과의 통합이 용이	<ul style="list-style-type: none"> • LDAP 서버와의 연동을 통해 사용자를 관리 • J2EE 서버와의 쉬운 통합 (JCA 제공) • RDB와 데이터 교환이 용이 (XSLT 사용)

2.4 XML 관련 기술

XML은 HTML과 같이 데이터의 기술과 프리젠테이션을 동시에 기술하는 언어가 아니라 이를 분리함으로써 데이터의 유용성과 재사용성을 제공함으로써 각광받고 있다 [4]. XML에서 사용하는 태그들은 사용자에게 의해 그 데이터의 목적만을 정의하기 때문

에 이런 재사용성이 높은 데이터가 표현을 필요로 할 경우는 그 데이터를 알아보기 쉬운 것으로 바꾸어줄 필요가 있다.

이런 기능을 제공하는 것이 XML 언어를 기반으로 XML의 장점을 포함하는 XSLT, SVG, XPath, XQuery와 같은 XML 관련 기술들이다. 이렇게 복잡하게 얽혀있는 XML을 사용하는 목적은 데이터의 목적이 뚜렷하고 재사용이 높은 데이터로 하나의 인터페이스를 만들었을 때 다양한 플랫폼에서 하나의 XML 문서를 통해 사용 가능하기 때문이다. 예를 들어, XML을 기반으로 웹 사이트 인터페이스를 만든다면, 플랫폼은 웹 브라우저 기반의 클라이언트, WML(Wireless Markup Language)을 사용하는 모바일 클라이언트, VoiceXML을 이용한 음성 기반의 브라우저에 이르기까지 다양한 플랫폼을 소화할 수 있다.

우리의 실생활에서도 인터넷과 컴퓨터를 자주 사용하는 사람들이 주로 사용하는 요즘의 메신저는 주소록, 사용자 정보, 메모와 같은 작은 크기의 텍스트 문서, 메신저 설정 등 그 외 많은 기능들을 가지고 있고 이런 정보들을 메신저가 XML로 저장한다면, 이 정보들을 인터넷이 연결되어 있는 곳이면 어디서든 다양한 프로그램과 플랫폼에 이식시킬 수 있게 된다.

(1) XSLT (eXtensible Style Language Transformation)

XSLT는 DSSSL(Document Style Semantics and Specification language)과 CSS(Cascading Style Sheet) 두 개의 고전적인 기술들을 기반으로 한다[5]. 일반적으로 XSLT 하나만으로 문서를 표현하지 않고 아직은 CSS와 함께 문서를 표현한다. XSLT가 더 구체적이고 다양한 기능을 표현할 수 있지만 그만큼 CSS보다 복잡한 구조를 가진다. 일반적으로 XSLT는 문서의 구조를 표현하고 CSS는 문서의 서식을 표

현한다고 볼 수 있다.

XSLT는 그 자체로는 출력이나 프로그래밍할 수 없는 XML과 GML 문서를 다른 언어로 변환하여 HTML이나 다양한 문서들로 생성해주는 역할을 한다. XML과 GML은 사용자 정의 태그로서 브라우저는 이것을 분석할 수 없다. 이런 GML 문서를 표현하기 위해서는 GML 문서가 어떤 방식으로 출력 되어야 하는지를 결정하기 위하여 CSS와 XSL(eXtensible Stylesheet Language)와 같은 메커니즘이 필요하다.

XSLT는 XML 문서의 변환 프로세서로서 인터넷익스플로러 6.0 이상 버전부터 자체 내장되어 있는 마이크로소프트사의 MSXML 파서*, Apache의 Xalan-C++**과 Xalan-Java 프로세서***, IBM의 LotusXSL****, Micheal Kay의 Saxon*****, James Clark의 XT***** 등 다양한 프로세서들이 존재한다. XML 문서를 여러가지 다양한 다른 문서로 변환하는데 필요한 XSLT 프로세서는 일련의 기술된 명령문을 적용함으로써 다른 문서 형식으로 변환하거나 재구성한다[5]. XSLT의 처리 과정은 표 2.12에서 처럼 크게 두 가지로 분리하여 나타낼 수 있다.

표 2.12 XSLT의 두 가지 처리 과정
Table 2.12 Two processes of XSLT

처리 과정	내용
구조적 변환 단계	입력된 XML 문서의 구조는 요구되는 출력 형태를 반영하는 구조로 데이터의 변환이 이루어진다
XSL 포매팅	입력된 XML 문서의 구조는 HTML이나 PDF 등의 형식을 가지는 출력 형태를 갖는다

* <http://support.microsoft.com/kb/269238>

** <http://xml.apache.org/xalan-c/>

*** <http://xml.apache.org/xalan-j/>

**** <http://www.alphaworks.ibm.com/tech/LotusXSL>

***** <http://www.saxonica.com/>

***** <http://www.blnz.com/xt>

(2) SVG (Scalable Vector Graphics)

SVG는 XML 기반의 언어로서 W3C에 의해 제안된 XML 2차원 그래픽 표현 표준이다[3]. SVG는 XML의 장점들을 포함하면서 SMIL, GML, MathML 등 다른 XML 언어들과 결합하여 다양한 웹 어플리케이션으로 응용 가능하다. 실시간 데이터로부터 고품질의 다이나믹한 그래픽을 만들어 낼 수 있어 전자상거래, 교육, 광고, 출력, 특히 지리정보 시스템 분야에서 큰 역할을 담당할 수 있다. 따라서 본 논문에서는 Macromedia社의 Flash나 기타 웹 기반의 벡터 그래픽을 사용하지 않고 XML 언어를 기반으로 하는 SVG를 사용한다[9,10].

SVG는 이미지, 텍스트, 벡터 세 가지 형식의 그래픽을 지원한다. SVG에서 이미지는 문서 내에 포함하거나 그렇지 않은 두 가지 방법을 취하고 있다. Adobe社의 일러스트레이터 프로그램이나 SVG로의 변환이 가능한 기타 프로그램을 사용하면 레스터 이미지를 SVG 코드로 바꾸어줄 수 있고 레스터 이미지인 만큼 압축 코드로 작성하여 SVG 문서로 만들 수도 있다. 이미지를 포함하지 않는 방법은 HTML에서 이미지를 추가하는 방법과 마찬가지로 논리적인 절대주소나 상대주소를 이용해 이미지 자체를 연결하는 방법으로 이미지를 표현한다[3,17].

또한 SVG는 기존 웹 스크립트 언어들을 지원한다. Javascript, Java, ASP, JSP, Visual Basic 등 기존 웹 스크립트 언어들을 사용해 사용자와 상호작용함으로써 동적으로 그래픽 문서를 제작할 수 있다. SVG의 가장 큰 특징 중 하나는 스타일시트 CSS 문서를 지원한다는 점이다. S-57의 경우 객체 단위로 구성되어 있기 때문에 객체별 스타일을 작성하면 XSL의 기본 스타일을 쉽게 확장하여 표현할 수 있고 문서의 레이아웃과 내용을 분리, 그래픽 요소 및 속성을 효과적으로 제어함으로써 유지 관리 비용을 줄이고 업데이트를 쉽게 할 수 있는 장점을 가진다[17]. 표 2.13은 SVG와 기존의 그

래픽 기술들을 비교하고 있다[9,10,17].

표 2.13 SVG와 다른 그래픽 표준 비교

Table 2.13 Comparisons of SVG and other graphic standards

비교	SVG	Flash	JPEG	GIF
이미지 형식	Vector	Vector	Raster	Raster
해상도 조절 지원	O	O	X	X
이벤트 스크립트 지원	O	O	X	X
HTML 표준 지원	O	O	O	O
DB 연동 지원	O	O	X	X
XML 기반 지원	O	X	X	X
이미지 내 텍스트 검색	O	X	X	X
애니메이션 지원	O	O	X	O

SVG는 XML과 마찬가지로 구조적인 정의가 가능하고 데이터베이스와 연동하여 동적인 그래픽 생성이 가능하다. 또한 모든 도형 및 그래픽 요소들은 좌표시스템을 기본으로 하기 때문에 지리 정보 시스템을 구성하는데 용이하고 위경도 뿐만 아니라 미터, 인치 등 대부분의 단위계를 지원한다. 표 2.14는 SVG 문서를 구성하는 주요 구성 요소를 나타낸다[3].

표 2.14 SVG 구성 요소

Table 2.14 Composition elements of SVG

구조	요소	내용
루트	svg	SVG 문서의 부분 정의, 중첩 가능
설명	title	SVG 문서의 제목
	desc	SVG 문서의 내용 설명
그룹	g	그래픽 요소들을 그룹으로 묶는다
기본 도형	line	두 개의 좌표로 이루어진 선
	polyline (path)	좌표의 나열로 다중선 그리기
	circle	중점 좌표와 반지름으로 표현하는 원
	rectangle	두 끝 좌표로 표현하는 사각형
	polygon	시작점과 끝점이 같은 폐쇄형 선. 다각형
심볼	symbol	그래픽 요소 심볼화
	def	참조될 요소 정의
	use	참조 가능한 요소 사용
텍스트	text	텍스트 속성 지정
링크	a	하이퍼링크 제공
스크립트	script	이벤트 스크립트 지정

제 3 장 GML 기반의 전자해도

일반적인 XML 전자해도 시스템은 그림 3.1과 같이 변환 모듈과 문서 저장소, 질의 처리기, 사용자 인터페이스로 구성된다[9-11,18-20].

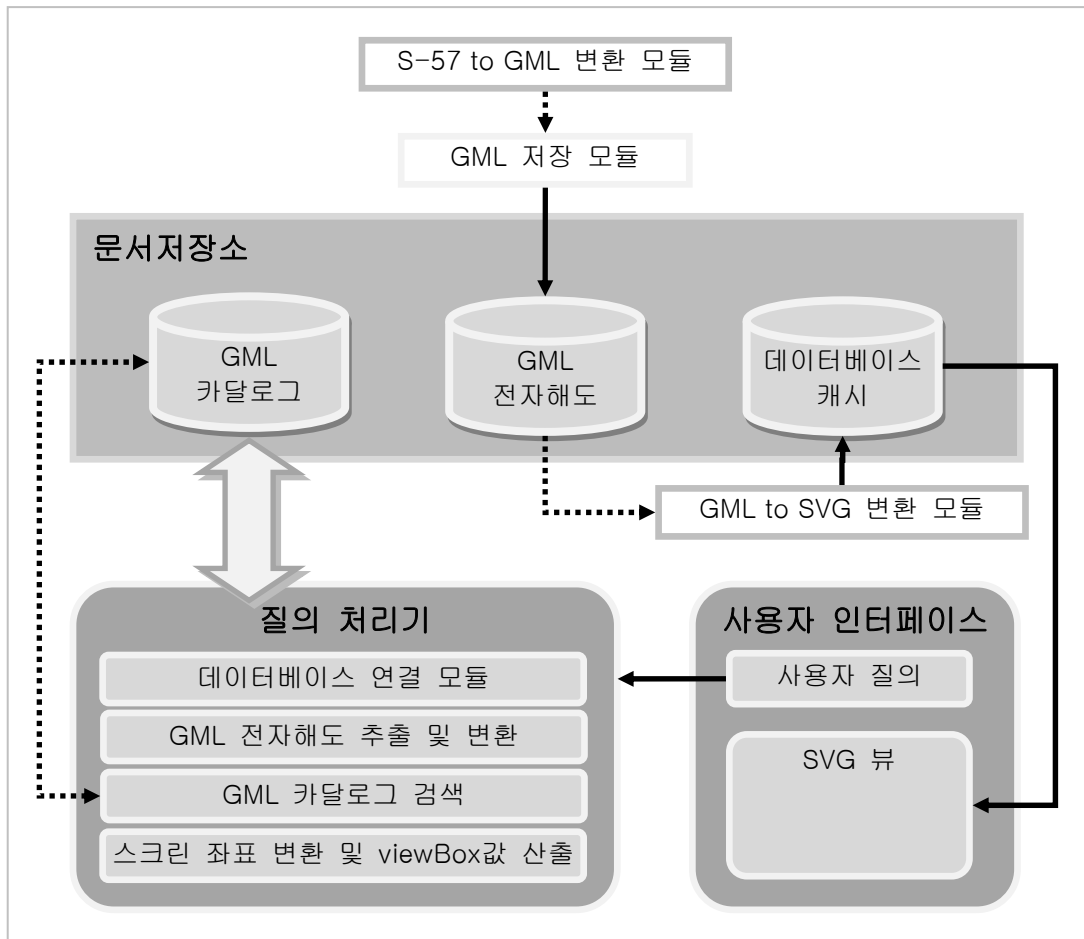


그림 3.1 시스템 구조

Fig. 3.1 The system structure

기존 XML 기반 전자해도 시스템은 사용자 질의나 셀렉트 톨과 같은 객체의 선택으로 검색된다. 본 논문에서는 입력 형태의 위경도 질의와 선택 형태의 축척 질의에 의해 검색하는 일반적인 전자해도 질의 형식을 가진다.

가로, 세로 길이 등 뷰어 관련 질의를 포함하여 몇 가지의 사용자 질의가 입력 또는 선택되면 질의 처리기는 전송된 질의에서 SVG의 viewBox 설정값을 데이터베이스 캐시에 먼저 저장하고 다음 작업을 수행한다. 전송된 위경도 질의는 데이터베이스에 저장되어 GML 전자해도 셀들의 정보를 요약해 놓은 GML 카탈로그를 검색 및 비교하여 해당하는 GML 전자해도 셀들을 추출한다. 추출된 셀들은 SVG 문서로 변환되는데 SVG 변환 모듈을 통해 S-57 위경도 좌표 포맷을 스크린 좌표로 바꿔주고 viewBox 값을 계산하여 설정하여 데이터베이스 캐시에 변환된 문서를 저장한다. 먼저 저장된 viewBox 설정값을 변환된 SVG 문서의 최 상위 SVG 태그에 삽입하여 최종 SVG 문서를 만들어내어 클라이언트에게 전송하고 사용자 인터페이스인 뷰어에서 SVG 문서를 표현한다.

3.1 S-57의 GML 변환

S-57 전자해도의 바이너리 파일 하나는 등대나 부표 등과 같은 객체 클래스들의 집합으로 구성된다. 객체 클래스는 각 객체들의 지리적 특징 객체와 위치, 모양 등의 좌표 정보들을 표현하는 공간 객체, 객체 식별자와 속성들로 구성된다. 그림 3.2는 S-57 전자해도의 GML 변환 과정을 나타낸다[4,11].

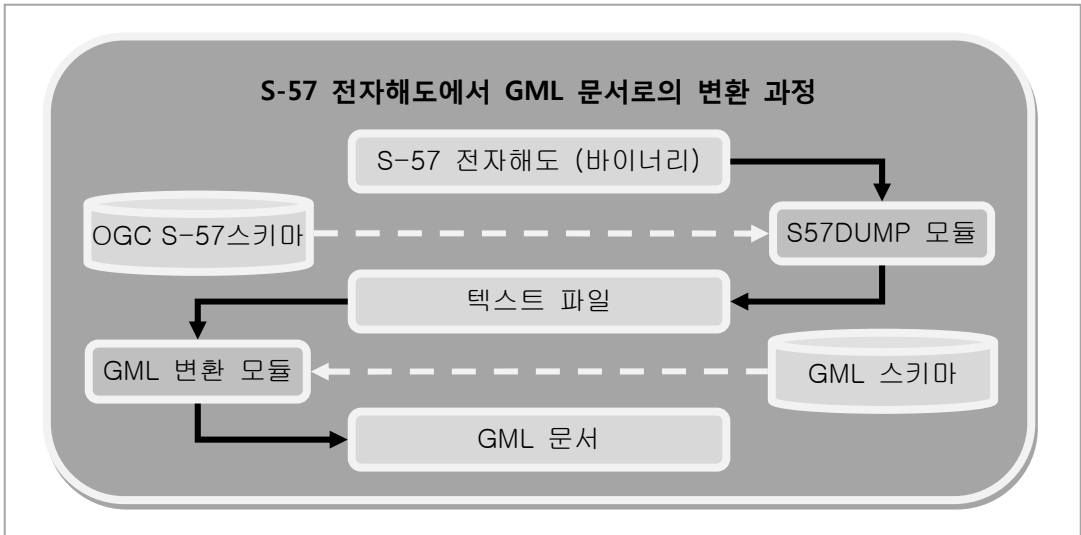


그림 3.2 S-57 전자해도의 GML 문서 변환 과정
 Fig. 3.2 The conversion processing from S-57 to GML

(1) 변환 알고리즘

변환 과정에서 입력 데이터는 S-57 전자해도 바이너리 파일이다. 입력 받은 S-57 전자해도가 객체 클래스 단위로 구성되므로 gdal 텍스트 변환 프로그램을 사용하여 클래스 단위의 텍스트 문서로 변환한다. 그런 다음 텍스트 파일을 GML로 변환하는 과정을 가지는데, 그림 3.3은 텍스트의 GML 변환 과정을 알고리즘으로 표현한 것이다[11].

그림 3.3은 S-57 전자해도 바이너리 파일을 S57DUMP 프로그램을 사용하여 텍스트 파일로 변환된 문서를 'T'라 정의하고 입력으로 사용된다. 표 3.1은 GML 변환 프로그램에서 사용된 함수들의 기능을 설명한다[11].

텍스트 파일의 GML 변환 알고리즘

```
1 Algorithm S-57 Into GML(T)
2
3 // 입력:S-57 파일을 S57DUMP 를 이용하여 변환된 텍스트 파일을 'T'라고 사용한다.
4 // 출력:GML 파일을 'GMLfile'이라고 사용한다.
5
6 begin
7   while (T == EOF) {
8     // 텍스트 파일을 1 줄씩 읽어들인다.
9     read_line = get_line() ;
10    // 1 줄을 *단위로 자른다.
11    token[] = Tokenizer(read_line, `*`) ;
12    for (i = 0 ; i < token.size() ; i++) {
13      if (isobject (token[i])) {
14        if (isfeature_object(token[i])) {
15          // 텍스트 토큰을 특징 GML 토큰으로 변환
16          GML_token = feature_pattern_matching(token[i]) ;
17          // 출력에 GML_token 을 추가
18          GMLfile.write (GML_token) ;
19        } else {
20          // 텍스트 토큰을 공간 GML 토큰으로 변환
21          GML_token = space_pattern_matching(token[i]) ;
22          // 출력에 GML_token 을 추가
23          GMLfile.write (GML_token) ;
24        }
25      }
26    }
27  }
28 End
```

그림 3.3 S-57을 GML로 변환하는 알고리즘

Fig. 3.3 An algorithm for transforming S-57 into GML

feature_pattern_matching(), space_pattern_matching() 함수에서 사용되는 스키마는 전자해도용 GML 응용 스키마로 Objects.xsd, Attributes.xsd, Support.xsd, AbstractAndSuperTypes.xsd, gml4s57.xsd, xlink.xsd를 적용하여 적합한 GML 문서로 변환한다. 변환 과정 중 위경도 좌표 시스템과의 왜곡 현상을 해결하기 위해 카탈로그 파일을 참조하여 메르카토르 투영법과 x, y 좌표 간 위치 변경,

S-57 셀 외곽의 빈 공간을 제거하기 위해 객체들의 좌표 추출 및 계산을 하여 커버리지에서 빈 공간을 제거할 수 있도록 하는 등의 좌표 변환 과정을 거쳐 변환된 GML 문서가 최종적으로 출력된다. 변환된 GML 문서는 다음절에서 자세히 설명한다.

표 3.1 GML 변환 프로그램의 함수들
Table 3.1 Functions for GML conversion

함수	기능 설명
get_line()	T에서 한줄씩 내용을 읽어오며 그 내용을 read_line으로 넣는다
Tokenizer()	read_line 한줄을 ‘*’ 단위로 자르고 token 배열에 저장한다
isobject()	token이 오브젝트인지 아닌지를 구별한다
isfeature_object()	특징 객체를 구별한다
feature_pattern_matching()	객체라고 구별된 token을 특징 객체의 스키마를 참조해서 GML로 만들고 GML_token에 값을 넘겨준다
space_pattern_matching()	객체라고 구별된 token을 공간 객체의 스키마를 참조해서 GML로 만들고 GML_token에 값을 넘겨준다
GMLfile.write()	GML_token을 최종 GML 파일에 내용을 삽입하는 역할을 한다

(2) 변환된 GML 문서

S-57 전자해도는 메타 객체, 지도제작 객체, 지리정보 객체, 콜렉션 객체 등 객체 클래스 단위의 순차적인 구조로 구성되므로 S-57 전자해도를 변환하기 위한 첫 단계는 S-57의 특성에 맞게 객체 클래스 단위로 분리하여 텍스트 문서로 변환한다. S-57 전자해도의 객체 구성과 마찬가지로 변환될 GML 문서도 하나의 객체 클래스에 특징 객체와 공간 객체로 구분된다.

특징객체는 객체 식별자와 세 개의 속성인 subset 'Attribute_A', subset 'Attribute_B', subset 'Attribute_C'에 대한 요소를 가져와 GML 문서로 변환한다. 특히 특징 객체의 세 가지 속성에 대해서는 미리 작성된 S-57 속성과 객체 테이블들을 참조하여 전자해도용 GML 응용 스키마 규칙에 적절한 GML 문서로 변환한다.

그림 3.4는 변환된 GML 문서 중 LIGHTS 객체 클래스 부분이다. 소스에서 34번째 줄까지가 실제 객체 클래스 부분이고 줄 35부터 마지막 줄까지가 실제 좌표 참조 부분에 속한다. 변환된 GML 문서의 전체 부분에서 좌표는 모든 객체 클래스 집합이 끝난 다음 줄부터 시작된다. 34번째 줄까지의 LIGHTS 객체 클래스 소스는 앞서 설명한 특징 객체와 공간 객체로 나뉘어진다. 18번째 줄까지는 식별자 표시를 포함하는 메타데이터 부분이고 19번째부터 32번째까지가 세 가지 속성에 속하는 부분이다. 모든 속성이 표시되지 않는 이유는 Null 값을 가진 속성을 변환 과정 중 표시하지 않도록 설정되어 있기 때문이다. 그리고 마지막의 33번째 줄이 좌표 참조 식별자를 가진 공간객체에 속한다[6,9-11].

변환된 GML 문서의 LIGHTS 객체 소스

```

1 <Light gml:id="_540_2135116304_688">
2   <gml:metaDataProperty>
3     <FeatureRecordIdentifier>
4       <recordName>100</recordName>
5       <recordIdentificationNumber>1116</recordIdentificationNumber>
6       <group>2</group>
7       <objectLevel>75</objectLevel>
8       <recordVersion>1</recordVersion>
9       <recordUpdateInstruction>1</recordUpdateInstruction>
10    </FeatureRecordIdentifier>
11  </gml:metaDataProperty>
12  <gml:metaDataProperty>
13    <FeatureObjectIdentifier>
14      <agency>540</agency>
15      <featureIdentificationNumber>2135116304</featureIdentificationNumber>
16      <fidSubdivision>688</fidSubdivision>
17    </FeatureObjectIdentifier>
18  </gml:metaDataProperty>
19  <acronym>LIGHTS</acronym>
20  <catlit><CategoryOfLight>
21    <code>37</code><idList>413</idList><value>air obstruction light</value>
22  </CategoryOfLight></catlit>
23  <Colour><code>75</code><idList>3</idList><value>red</value></Colour>
24  <height><Height><code>95</code><value>44</value></Height></height>
25  <litchr><LightCharacteristic>
26    <code>107</code><id>7</id><value>isophased</value>
27  </LightCharacteristic></litchr>
28  <SignalGroup><code>141</code><value>(1)</value></SignalGroup>
29  <SignalPeriod><code>142</code><value>6</value></SignalPeriod>
30  <valnmr><ValueOfNominalRange>
31    <code>178</code><value>18</value>
32  </ValueOfNominalRange></valnmr>
33  <position xlink:href="#N120_1746"/>
34 </Light>
35 <gml:Node gml:id="N120_1746">
36   <gml:pointProperty xlink:href="#P120_1746"/></gml:Node>
37 <gml:Point gml:id="P120_1746">
38   <gml:pos>-4608596 388197502</gml:pos> // 메르카토르 투영법을 통해 변환된 좌표
39 </gml:Point>

```

그림 3.4 변환된 GML 문서(Light 객체 클래스 부분)

Fig. 3.4 A GML document to be translated

3.2 변환된 GML 문서의 XML 데이터베이스 저장

XML 문서에 대해서 기존의 XML 가능 데이터베이스들이 XML의 특징을 제대로 사용하는데 비효율적이고 기본 데이터 모델이 XML과 상이하기 때문에 XML 형식의 데이터를 저장하는데 많은 어려움이 있다. 본 논문에서는 GML 전자해도의 저장과 관리, 사용자질의를 효율적으로 처리할 수 있는 XML전용 데이터베이스를 사용한다.

(1) 데이터베이스 구조

eXcelon社의 XIS와 같은 XML 데이터베이스는 GML 전자해도를 체계적으로 저장할 수 있고 특히 사용자 검색 질의를 효율적으로 처리하는 장점을 가진다[16]. XIS는 XML 데이터를 포함해 다양한 종류의 데이터들을 저장할 수 있는데 텍스트 형식의 문서들뿐 아니라 바이너리 파일로 된 그림이나 문서들 또한 저장할 수 있다. 저장되는 파일들은 XIS의 저장소(Repository)에 저장되고 Partition, XMLStore, Directory, File 네 가지 순서대로의 계층적 구조를 가진다. 실제 데이터들은 XMLStore와 Directory에 저장되고 유일한 이름의 파티션 안에서 이루어진다.

이런 일련의 작업들은 XIS DXE Manager를 사용하여 수행되고 XML Partition, XMLStore, Directory, File의 생성과 수정, 추가, 추출, 삭제 등과 같은 작업들을 수행한다. 특히 XPath와 XLink, XQuery의 질의 처리도 가능하고 XIS가 제공하는 자바 컴퍼넌트를 이용해 JSP나 자바 프로그램을 작성할 수 있다. 만일 운영체제가 유닉스나 리눅스의 경우이거나 텔넷 등의 셸 프로그램을 이용하는 네트워크 작업을 주로 한다면 XIS 자체에서 제공하는 셸 명령어를 통해서도 이런 작업들이 가능하다. 그림 3.5는 XIS를 이용하여 설계한 GML 저장 구조를 나타낸다[11,16].

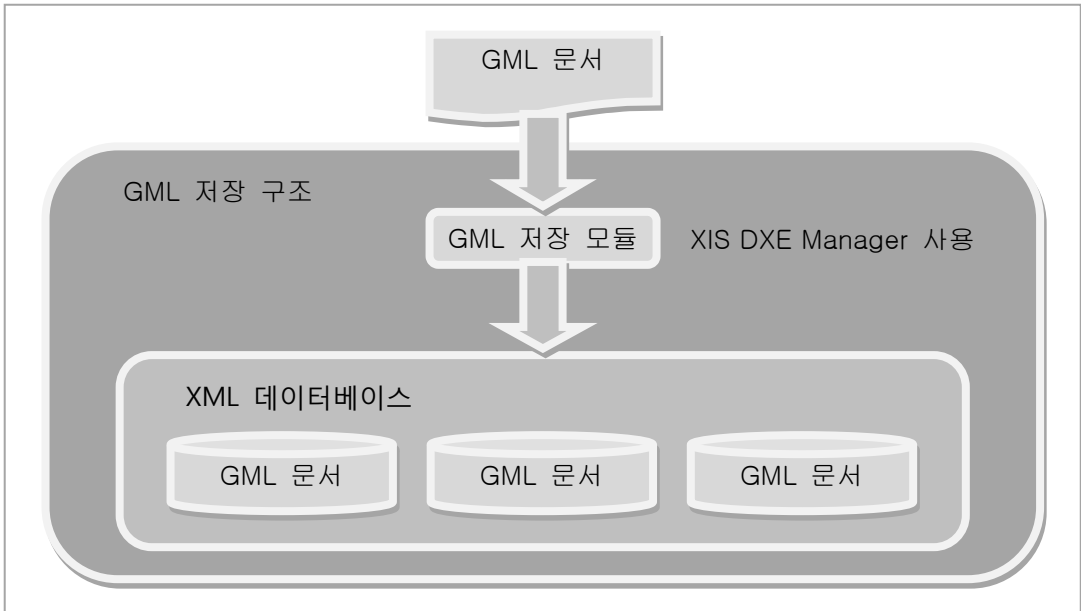


그림 3.5 GML 저장 구조

Fig. 3.5 The storage structure for GML

표 3.2와 표3.3은 이 네 가지 계층적 구조와 파일의 종류를 설명한다[16].

표 3.2 XIS 서버 저장시스템의 구조 요소 설명

Table 3.2 The factors of the storage structure within XIS server

구조	내용	저장
Partition	<ul style="list-style-type: none"> - 운영체제의 특정 파일시스템 위치(Directory)와 연결 - 주로 RAID 디스크 배열의 마운트 디렉토리가 사용 - Partition의 이름은 서버에서 중복 불가능 	XMLStore 그룹
XMLStore	<ul style="list-style-type: none"> - 계층적 구조로 저장된 데이터의 루트역할을 수행 - XMLStore의 개수는 처리 업무와 데이터 양, 그 설계에 의존적 	Directory, File, Metadata(하위 계층 접근정보)
Directory	<ul style="list-style-type: none"> - 일반 시스템의 디렉토리과 같은 역할 	Directory, File
File	<ul style="list-style-type: none"> - 다양한 종류의 데이터를 포함 	

표 3.3 XIS에 저장 가능한 파일 종류
Table 3.3 Various kinds of files for XIS

파일 종류	내용
XML 문서	일반적인 XML 문서
XSL 문서	스타일시트를 포함한 XML 출력을 위한 변환 규칙 문서
DTD 문서	XML 문서형식정의 문서
XML 스키마 문서	XML 문서형식정의 문서
가상 파일	XIS 서버 확장에 연결된 파일
Binder	여러 종류, 다수의 XML 파일들의 컬렉션
BLOB	XML을 제외한 데이터: GIF, JPEG, Word 문서, PowerPoint 문서 등 다양한 데이터 저장

GML 저장 모듈에서는 DTD나 스키마에 정의된 형식 및 규칙과 비교하여 XML 문서의 구조에 문법적인 오류 검사를 확인하는 기능을 가진다. 그림 3.6은 저장 모듈을 순서도로 나타낸 것이다[11,16].

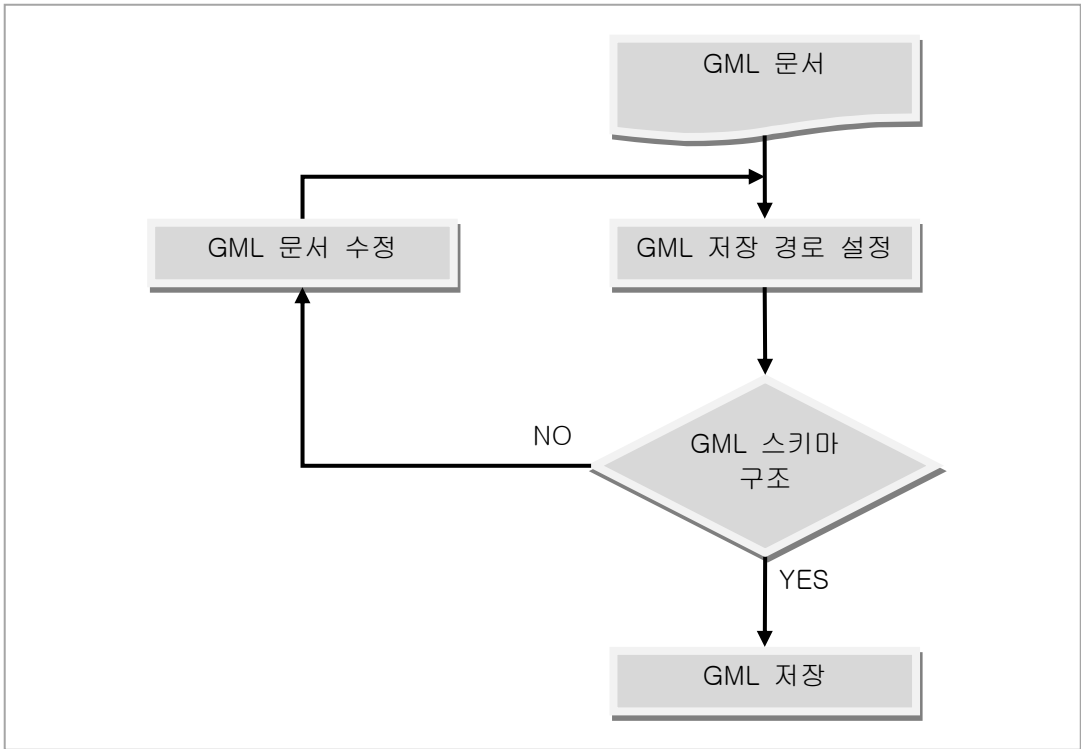


그림 3.6 XML 데이터베이스 저장 모듈 순서도
 Fig. 3.6 The flowchart of the storage module for the XML database

3.3 클라이언트 뷰어

XML 전자해도에서 클라이언트 뷰어는 SVG 문서를 보여주고, 질의하며, 설정하는 사용자 인터페이스 부분으로서 그림 3.7과 같이 실제 변환된 SVG 문서가 보여지는 '뷰', 질의를 입력 받는 '질의입력기', 상하좌우로 전자해도를 재배열하고 확대 및 축소하는 기능 부분인 '툴바'로 나눌 수 있다. SVG의 뷰어 자체는 일반적으로 마이크로소프트사의 인터넷 익스플로러 브라우저를 사용하기 때문에 일반 웹 언어인 HTML과

Java 서버 페이지 언어인 JSP 등을 사용할 수 있고 SVG 문서가 보여지는 '뷰' 부분은 Adobe SVG Viewer* 플러그인을 통해 보여진다[21].

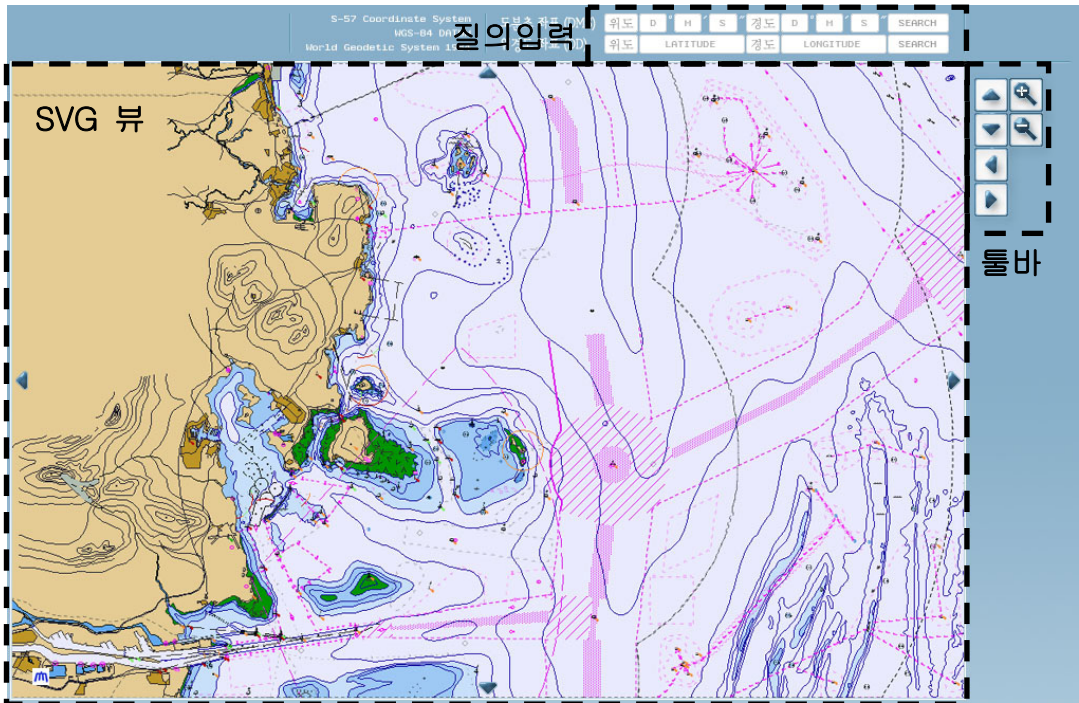


그림 3.7 클라이언트 뷰어의 형태
 Fig. 3.7 The shape of a client viewer

질의 입력기는 질의를 입력 받는 역할을 수행하고 입력 받은 위경도 좌표는 질의 처리기에 의해 중점 좌표와 셀 외곽 위경도 좌표를 계산하여 XML 데이터베이스를 검색 및 추출하는 과정을 거친다. 좌표 입력은 도분초 좌표 DMS와 위경도 좌표 DD 두 가지를 지원할 수 있고 실제 질의로 검색되는 좌표는 위경도 좌표로서 DMS 좌표가 입

* <http://www.adobe.com/svg/viewer/install/>

력되었을 경우, 위경도 좌표로 재계산하여 검색되는 형태가 일반적이다. 뷰어의 툴 박스는 이동 버튼과 줌 버튼으로 구성되고 이 기능들은 SVG가 가지고 있는 기본 기능으로서 수행된다.

표 3.4는 본 논문의 클라이언트 뷰어에서 전송되는 질의 요소들을 나타낸다. 사용자는 위경도 좌표 값을 입력하고 기본 축척 외에 다른 축척을 선택 툴을 통해 질의한다. 사용자 인터페이스는 사용자가 좌표 값을 입력하였거나, 축척을 선택하거나, 이동 버튼을 눌렀을 때나, 인터페이스의 SVG 뷰 크기가 변할 때마다 SVG 뷰의 가로와 세로 길이를 포함한 현재 질의 설정 값을 모두 전송한다. SVG 뷰의 가로와 세로 길이 값은 현재 설정된 축척 값과 함께 뷰에서 보여질 위경도 최소, 최대 좌표 값을 산출하여 변환된 SVG 문서가 데이터베이스 캐시에 저장될 때 SVG의 최상위 viewBox 값으로 삽입되어 최종 SVG 문서가 생성되도록 하였다.

표 3.4 사용자 질의 요소들

Table 3.4 The elements for the user queries

질의 요소들	내용	질의 개체	형식
qy	위도 좌표 값	사용자	Input
qx	경도 좌표 값	사용자	Input
qs	축척 값	사용자	Select
vw	SVG 뷰의 가로 길이	인터페이스	Hidden
vh	SVG 뷰의 세로 길이	인터페이스	Hidden

제 4 장 전자해도의 GML 및 SVG 변환 기법

전자해도 시스템 구조에서 데이터가 변환되는 부분은 S-57 이진 데이터의 GML로의 변환과 GML에서 SVG로 변환되는 두 가지가 존재한다. S-57에서 GML로 변환될 때 핵심 기술로서 GML 응용스키마를 요구하고, GML에서 SVG로 변환 시 XSLT 변환기를 필요로 한다. 그러나 변환 과정 전반에서 몇 가지 이상 현상이 일어났으며, 이는 S-57 전자해도 파일이 가진 포맷과 SVG 포맷과의 좌표 충돌에서 그 해결점을 찾았다.

본 논문에서는 전자해도 시스템 구조에서 위경도 질의가 처리되는 과정을 제안하고 S-57 전자해도를 변환 시 유의할 점과 그 해결 방안을 제시함으로써 변환 기법에 필요한 요소들을 다룬다.

4.1 변환 시 유의할 점

S-57 데이터는 위경도 좌표를 기본으로 하는 전자해도로써 스크린 좌표와 전혀 다른 좌표체계를 가진다. SVG 명세서에서는 지원하는 좌표 시스템 중 위경도 좌표를 포함하나 실제 위경도 시스템을 가진 데이터를 변환하면 SVG는 위경도 좌표를 스크린 좌표로 표현한다. 이 때문에 전자해도를 표현하는데 심각한 이상 현상을 보이며 본 논문에서는 몇 가지 문제점들 중 가장 심각한 세 가지 현상에 대한 원인과 해결 방안을 제시한다.

(1) 회전 현상

그림 4.1은 실제 S-57 전자해도를 나타내며 그림 4.2는 SVG문서로 변환했을 때 -90도로 회전되어 왜곡된 전자해도를 나타낸다. 이런 이상 현상은 S-57 전자해도의 좌표 시스템이 위경도를 기준으로 하기 때문으로 본 논문에서 사용하는 테스트 데이터가 적도(0°)와 남반구(-90°) 사이에 있어, 남쪽으로 갈수록 y좌표 값이 높아지는 스크린 좌표와는 반대로 남쪽으로 갈수록 위도의 값은 낮아진다.

SVG에서도 위경도 좌표를 지원하지 않는 것은 아니지만 이런 회전 현상을 보이는 것은 위경도 좌표가 적도를 기준으로 북반구로 갈수록 (+) 값을 가지고 남반구로 갈수록 (-) 값을 가지는 WGS-98 좌표 시스템의 DD 형식의 특성을 가졌기 때문이다. 즉, SVG Specification에 포함된 좌표 지원 포맷으로 위경도 좌표가 있으나 SVG에서 위경도 좌표는 실제 스크린 좌표 형태로 표현된다. 이로 인해 원본 전자해도 데이터가 -90도 회전하는 이상 현상을 보이게 된다.

이 현상을 제거하기 위해서는 먼저 (YCOO, XCOO) 형태를 띄고 있는 S-57 위경도 좌표를 SVG 스크린 좌표에서 (XCOO, YCOO)로 단순히 변경하면 된다. 또한 뒤의 다른 현상들을 함께 수정하기 위해서는 0°를 기준점으로 하는 적도의 기준을 바꿔 주는 방법을 제안한다. 그러나 S-57은 국제 전자해도 표준이기 때문에 XML 전자해도 시스템 내에서 데이터베이스에 저장될 GML 문서들에서 이런 현상을 제거하고 표현하기엔 S-57 개발의 취지에 맞지 않다. 따라서 본 논문에서는 이 현상을 해결하기 위해 마지막 SVG 문서로 변환될 때 마다 SVG의 `viewbox` 기준점을 위도 최소값과 경도 최소값으로 하지 않고 두 좌표 모두 0으로 그 기준점을 정의하였다.

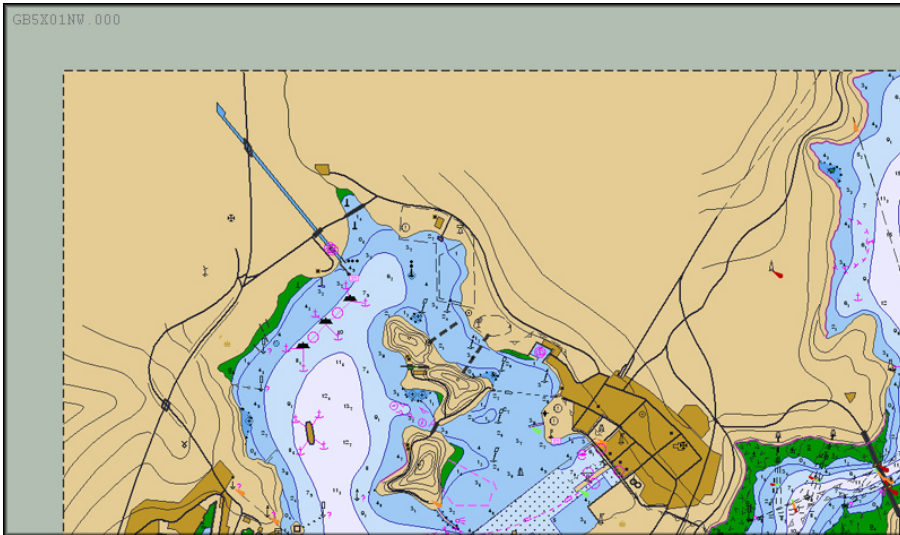


그림 4.1 GB5X01NW.000 실제 전자해도의 표현
 Fig. 4.1 The representation of S-57 ENC, GB5X01NW.000



그림 4.2 -90도 회전된 SVG 문서
 GB5X01NW.svg 표현
 Fig. 4.2 The representation of SVG document,
 rotated -90 degree, GB5X01NW.svg

다음 결과 문서 전체에 포함된 좌표들을 해당 (0, 0) 기준점에 대해 재계산하여 SVG 문서를 생성한다. 이 부분은 (3) SVG viewBox에서 설명한다.

(2) 셀 외곽 빈 공간

그림 4.3에서 알 수 있듯이 S-57 전자해도 데이터는 셀 외곽에 빈 공간을 표시한다. 이런 S-57 전자해도의 특성 때문에 좌표 단위의 질의로 문서를 생성하여 표현할 때 빈 공간이 생기는 현상이 발생한다. 이런 현상은 SVG 문서를 뷰어에서 이동, 확대, 축소를 하거나 회전 기능을 사용할 경우 심각한 이상 현상을 나타낸다.

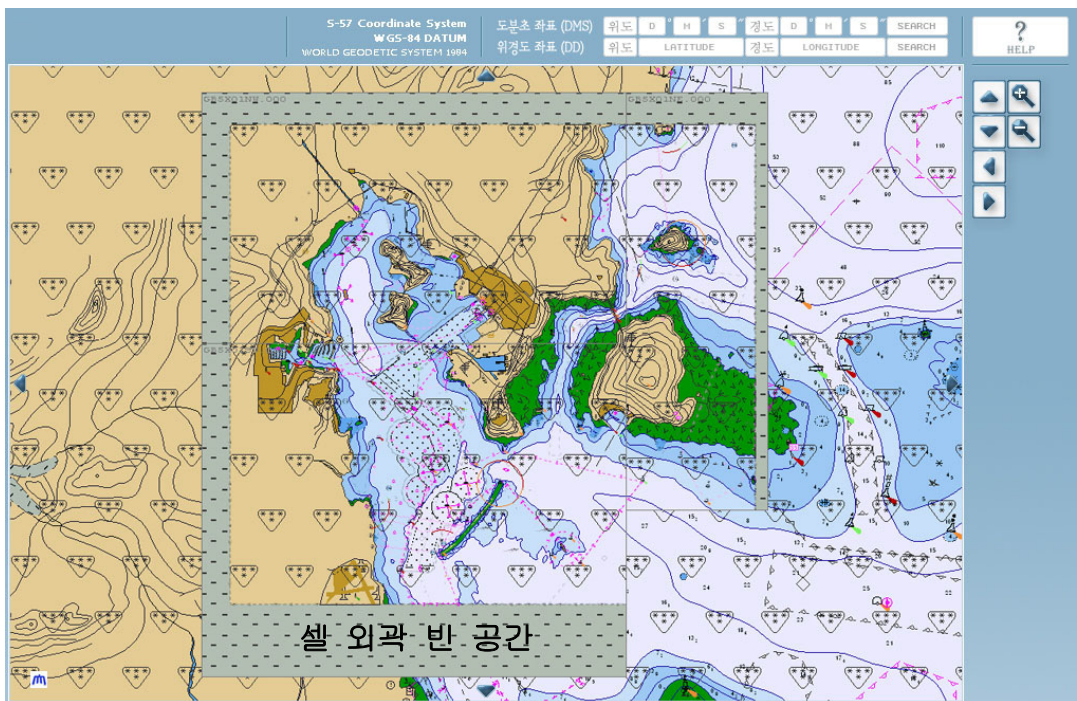


그림 4.3 셀 외곽의 빈 공간
Fig. 4.3 The empty area around cells

S-57 전자해도는 그림 4.4와 4.5에서처럼 두 가지 형태의 커버리지 객체를 포함하고 있다. 두 객체는 'no coverage available'과 'coverage available'이라는 옵션을 가진 하나의 객체로서 인식되며, 이런 옵션은 전자해도 브라우저 설정 방식으로 지정할 수 있다. 커버리지 객체 외에도 'available' 형 옵션을 가진 객체들이 존재하나 셀의 외곽 빈 공간과 같은 현상을 일으키는 객체는 커버리지 객체뿐이다.

```

no coverage available 타입의 커버리지 객체
1 <Coverage gml:id="_540_2135152982_687">
2   <gml:metaDataProperty>
3     <FeatureRecordIdentifier>
4       <recordName>100</recordName>
5       <recordIdentificationNumber>440</recordIdentificationNumber>
6       <group>2</group>
7       <objectLevel>302</objectLevel>
8       <recordVersion>1</recordVersion>
9       <recordUpdateInstruction>1</recordUpdateInstruction>
10    </FeatureRecordIdentifier>
11  </gml:metaDataProperty>
12  <gml:metaDataProperty>
13    <FeatureObjectIdentifier>
14      <agency>540</agency>
15      <featureIdentificationNumber>2135152982</featureIdentificationNumber>
16      <featureIdentificationSubdivision>687</featureIdentificationSubdivision>
17    </FeatureObjectIdentifier>
18  </gml:metaDataProperty>
19  <acronym>M_COVR</acronym>
20  <catcov>
21    <CategoryOfCoverage>
22      <code>18</code>
23      <id>2</id>
24      <value>no coverage available</value>
25    </CategoryOfCoverage>
26  </catcov>
27  <extent xlink:href="#F100_440"/>
28 </Coverage>

```

그림 4.4 No coverage available 타입의 커버리지 객체
 Fig. 4.4 The coverage object on type 'no coverage available'

```

coverage available 타입의 커버리지 객체
1 <Coverage gml:id="_540_2135153003_687">
2   <gml:metaDataProperty>
3     <FeatureRecordIdentifier>
4       <recordName>100</recordName>
5       <recordIdentificationNumber>441</recordIdentificationNumber>
6     </FeatureRecordIdentifier>
7     <group>2</group>
8     <objectLevel>302</objectLevel>
9     <recordVersion>1</recordVersion>
10    <recordUpdateInstruction>1</recordUpdateInstruction>
11  </gml:metaDataProperty>
12  <gml:metaDataProperty>
13    <FeatureObjectIdentifier>
14      <agency>540</agency>
15      <featureIdentificationNumber>2135153003</featureIdentificationNumber>
16      <featureIdentificationSubdivision>687</featureIdentificationSubdivision>
17    </FeatureObjectIdentifier>
18  </gml:metaDataProperty>
19  <acronym>M_COVR</acronym>
20  <catcov>
21    <CategoryOfCoverage>
22      <code>18</code>
23      <id>1</id>
24      <value>coverage available</value>
25    </CategoryOfCoverage>
26  </catcov>
27  <extent xlink:href="#F100_441"/>
28 </Coverage>

```

그림 4.5 Coverage available 타입의 커버리지 객체
 Fig. 4.5 The coverage object on type 'coverage available'

셀 외곽의 빈 공간이 차지하는 부분은 커버리지가 보여질 때 그림 4.3의 회색 부분처럼 배경색을 가지게 되어 만일 더 큰 축척의 S-57 셀 위에 작은 축척의 S-57 셀이 상위 레이어로 놓여진다면 그림 4.3의 빈 공간 영역만큼 하위 레이어 셀을 가리게 되거나 이동, 확대, 축소, 회전 등에 심각한 이상 현상을 일으킨다. 특히 레이어 형태로 여러 셀이 합쳐지는 경우에는 이 공간 때문에 약간의 속도 저하와 객체가 존재하는 영

역과 겹쳐지는 현상을 일으켜 정확한 지도를 표시하기 힘들다.

이것을 해결하기 위해서는 커버리지를 제외한 객체들의 위경도 좌표 중 최소값 좌표와 최대값 좌표를 추출하여 커버리지 좌표 중 해당 영역에 존재하지 않는 좌표들을 제거해야만 한다. 그러나 회전 현상과 마찬가지로 GML 문서 자체에서 이런 문제를 해결하는 것은 표준 데이터로서의 의미가 없다. 따라서 본 논문에서는 커버리지 객체를 제외한 나머지 객체들의 좌표에서 최대, 최소값 좌표를 추출 후, GML 카달로그 파일을 작성하는 방법을 제안한다.

참고로 카달로그 파일은 상용 S-57 데이터 시디에도 포함되어 있으나 SVG의 viewBox 옵션 좌표 등 여러 가지 부분을 추가함으로써 위경도 질의에 따른 빠른 검색을 가능하게 하는 전자해도 시스템 내 중요한 요소 중 하나이다.

(3) SVG viewBox

S-57 전자해도가 최종 문서인 SVG 문서로 변환될 때 가장 중요한 요소 중 하나가 SVG의 viewBox 옵션이다. viewBox는 검색된 전체 이미지에서 필요한 범위만을 잘라 내거나(Crop 기능) 뷰포트의 역할을 하는 등 전자해도를 표현하는데 있어 중요한 부분 중 하나다. 그러나 위경도를 표현하는데 있어, 실제와 다른 도형을 표현하거나 SVG의 기본 기능인 이동, 회전, 확대, 축소에서 정확히 표현하지 못하는 현상, 기준점 (0,0)을 제대로 표현하지 못하는 현상 등 여러 문제점을 일으키는 부분이기도 하다.

이런 현상들의 원인은 대부분 회전현상에서 설명한 것처럼 위경도좌표와 스크린좌표의 충돌 때문이지만 viewBox 기준점이 (0,0)으로 잡히지 않았을 때도 발생한다. 따라서 viewBox의 기준점을 (0,0)으로 변경하여 기준점에 맞춰 모든 좌표를 변경함으로

써 문제를 해결되며 이 과정은 SVG 문서로 변환하여 뷰어에 보여질 때마다 수행함으로써 하나의 GML 데이터를 사용하여 다양한 플랫폼에 적용할 수 있는 장점을 제공한다.

기준점을 (0, 0)으로 바꾸는 것은 카탈로그 파일에 포함된 경도 최소값, 경도 최대값, 위도 최소값, 위도 최대값 네 가지 값으로 계산된다. x좌표의 기준점은 (-)값으로 되어 있으므로 위도 최대값을 빼주면 되고 y좌표 기준점은 (+)값이기 때문에 경도 최소값으로 빼주면 된다. 문서에 포함된 모든 좌표들을 위도 최대값과 경도 최소값을 이용해 재계산한다.

viewbox 기준점의 변경 외에도 경위도를 SVG로 정확하게 표현하기 위해서는 x좌표와 y좌표를 서로 바꿔줘야 한다. 이것은 S-57 전자해도가 가진 (위도값, 경도값)이라는 좌표 시스템 때문이다. 실제로 위도는 적도를 기준으로 북반구와 남반구에 따른 값을 가지므로 이것은 Y좌표에 속하고 경도는 X좌표에 속한다. 즉, viewbox의 기준점부터 모든 좌표 체계를 바꾸어주면 이런 왜곡 현상이 해결된다.

표 4.1과 4.2는 SVG의 viewBox에 작성되는 기준점을 비교한 것으로 표 4.1에서 S-57 전자해도로부터 추출한 위경도 최소·최대값을 이용해 작성한 viewBox 기준점과 위경도 단위의 가로, 세로 길이를 표현하고, 표 4.2는 표 4.1의 기준점을 (0,0)으로 하는 viewBox를 표현한다. viewBox의 기준점은 위도의 경우 모든 위도값에 최소값을 뺀 후 절대값으로 만들고 경도의 경우엔 단지 최소값을 뺀 값으로 기준점을 설정한다.

표 4.1 S-57 전자해도의 viewBox 좌표

Table 4.1 The coordinates in viewBox for S-57 ENC

위경도 좌표	
파일	viewBox
GB4X0000	(-32.633333 60.766667 0.316666 0.566666)
GB5X01NE	(-32.533333 60.966667 0.083333 0.033333)
GB5X01NW	(-32.500000 60.866667 0.050000 0.100000)
GB5X01SE	(-32.566666 60.966666 0.033332 0.033334)
GB5X01SW	(-32.566667 60.866667 0.066667 0.100000)
GB5X02SE	(-32.566667 60.983333 0.028334 0.025000)

표 4.2 기준점 (0,0)으로 변경된 viewBox

Table 4.2 The coordinates in the viewBox based (0,0)

스크린 좌표	
파일	viewBox
GB4X0000	(0.000000 0.000000 0.316666 0.566666)
GB5X01NE	(0.000000 0.000000 0.083333 0.033333)
GB5X01NW	(0.000000 0.000000 0.050000 0.100000)
GB5X01SE	(0.000000 0.000000 0.033332 0.033334)
GB5X01SW	(0.000000 0.000000 0.066667 0.100000)
GB5X02SE	(0.000000 0.000000 0.028334 0.025000)

4.2 카탈로그 파일의 작성

XML 문서는 많은 객체와 좌표 정보들을 담고 있고 모두 텍스트로 되어 있어 파일 크기만으로도 대용량의 무게를 가진다[11]. 이런 대용량 XML 문서들을 데이터베이스를 통해 검색할 경우 심각한 트래픽을 야기시키고 많은 시간적 손실이 발생한다. 따라서 본 논문에서는 위경도 좌표 질의가 해당하는 셀들을 검색하는 방법으로 데이터베이스에 저장된 GML 전자해도들을 검색하지 않고 GML 전자해도 셀들의 정보를 담고

있는 카달로그를 작성하여 검색함으로써 빠른 탐색과 쉬운 접근이 용이하도록 하였다.

(1) S-57 카달로그

그림 4.6과 같은 S-57 카달로그는 제공 CD마다 확장자 .031을 가진 파일로 제공되어 해당 CD에 포함된 모든 S-57 전자해도 셀들의 요약 정보들과 위경도 좌표로 표현하는 셀의 크기를 포함한다. 또한 객체의 설명이나 현상에 관한 텍스트 파일들 (.TXT)에 대한 정보, 카달로그의 메타정보들을 표현한다.

```
002413LE1 0900058 ! 34040000019000000010440019CATD1200063-0000;& -
0001CATD-0100;& ISO 8211 Record Identifier(I(5))-1600;& Catalog Direc-
tory fieldRCNM!RCID!FILE!LFIL!VOLM!IMPL!SLAT!WLN!NLAT!ELON!CRCS!COMT-
(A(2),I(10),3A,A(3),4R,2A)-00115 D 00039
21040001060CATD706-00000-CD0000000001CATALOG.031V01X01ASCEXchange Set
Catalog file...-00142 D 00039
21040001060CATD976-00001-CD0000000002GB4X0000.000V01X01BIN-32.63333330-
60.76666670-32.3166667061.33333330815CD84C-00142 D 00039
21040001060CATD976-00002-CD0000000003GB5X01NE.000V01X01BIN-32.53333332-
60.96666668-32.4500000061.000000005162A9ED-00142 D 00039
21040001060CATD976-00003-CD0000000004GB5X01NW.000V01X01BIN-32.50000000-
60.86666670-32.4500000060.966666705DAA4F06-00142 D 00039
21040001060CATD976-00004-CD0000000005GB5X01SE.000V01X01BIN-32.56666600-
60.96666600-32.5333340061.000000000B9D9E68-00142 D 00039
21040001060CATD976-00005-CD0000000006GB5X01SW.000V01X01BIN-32.56666668-
60.86666668-32.5000000060.96666668EE18D91B-00142 D 00039
21040001060CATD976-00006-CD0000000007GB5X02SE.000V01X01BIN-32.56666668-
60.98333332-32.5383333261.00833332085FB317-00142
```

그림 4.6 S-57 카달로그 파일
Fig. 4.6 The S-57 catalogue file

표 4.3은 CATALOG.031 카달로그 파일의 요소들을 보여준다.

표 4.3 S-57 카달로그 파일의 요소들

Table 4.3 The elements of the S-57 catalogue files

CATD	97600001	97600002	97600003	97600004	97600005	97600006
RCID	CD2	CD3	CD4	CD5	CD6	CD7
FILE	GB4X0000	GB5X01NE	GB5X01NW	GB5X01SE	GB5X01SW	GB5X02SE
VOLM	V01X01	V01X01	V01X01	V01X01	V01X01	V01X01
IMPL	BIN	BIN	BIN	BIN	BIN	BIN
SLAT	-32.6333333	-32.5333333	-32.5000000	-32.5666666	-32.5666666	-32.5666666
WLON	60.7666667	60.9666668	60.8666667	60.9666660	60.8666668	60.9833333
NLAT	-32.3166667	-32.4500000	-32.4500000	-32.5333340	-32.5000000	-32.5383333
ELON	61.3333333	61.0000000	60.9666667	61.0000000	60.9666668	61.0083333
CRC	815CD84C	5162A9ED	5DAA4F06	0B9D9E68	EE18D91B	085FB317

S-57 카달로그는 숫자와 줄임말로만 표현되어 복잡한 구조를 가지고 있어 이를 이용하여 검색을 할 경우 매번 파싱 과정을 거쳐야만 하며 4.1 절에서 언급하였듯이 커버리지를 포함한 최소, 최대 위경도 좌표 값을 통해 셀 크기를 표현한다. 또한 S-57 카달로그가 가진 셀 정보만으로는 검색에 어려움이 있고 XML 전용 데이터베이스를 사용하는 XML 전자해도 시스템에서는 더 복잡한 과정을 요구하기 때문에 사용에 불편함이 있다. 따라서 본 논문은 XML 기반으로 작성된 GML 카달로그를 작성하여 XML 전용 데이터베이스에 저장하는 방법을 사용한다.

(2) GML 카달로그

S-57 카달로그는 복잡한 구조로 되어있기 때문에 카달로그를 참조할 때마다 구조를 분석하거나 파싱 과정을 거쳐야 하며 원활한 검색을 위해 관계형 데이터베이스를 요구하는 등의 복잡한 단계를 필요로 한다. 또한 S-57 카달로그 파일은 파일의 정보 외에

좌표에 관한 요소는 최대, 최소 위경도 좌표값뿐이다. 따라서 이 정보만으로는 위경도 좌표 질의에 대한 원활한 검색이 어렵기 때문에 본 논문에서는 XML 기반의 GML 카달로그를 작성하였다. 표 4.4는 GML 전자해도를 위한 카달로그 파일 요소들을 나타낸다.

표 4.4 GML 카달로그 파일 요소들

Table 4.4 The elements of the GML catalogue files

S-57 ENC 파일	GB4X0000	GB5X01NE	GB5X01NW	GB5X01SE	GB5X01SW	GB5X02SE
Latitude North 위도 최대값	-32.316667	-32.450000	-32.450000	-32.533334	-32.500000	-32.538333
Longitude East 경도 최대값	61.333333	61.000000	60.966667	61.000000	60.966667	61.008333
Latitude South 위도 최소값	-32.633333	-32.533333	-32.500000	-32.566666	-32.566667	-32.566667
Longitude West 경도 최소값	60.766667	60.966667	60.866667	60.966666	60.866667	60.983333
viewbox y 기준점	-32.633333	-32.533333	-32.500000	-32.566666	-32.566667	-32.566667
viewbox x 기준점	60.766667	60.966667	60.866667	60.966666	60.866667	60.983333
viewbox 가로	0.316666	0.083333	0.050000	0.033332	0.066667	0.028334
viewbox 세로	0.566666	0.033333	0.100000	0.033334	0.100000	0.025000
기본 축척	52,000	25,000	25,000	25,000	25,000	25,000
최소 축척	80,000	40,000	40,000	40,000	40,000	40,000

GML 카달로그 파일은 S-57의 최대, 최소 위경도 좌표값을 가지면서 SVG의 viewBox 기준점과 가로 세로 요소들을 포함하며 기본 축척과 최소 축척 정보를 가진다. GML 카달로그는 그림 4.7에서처럼 하나의 카달로그 파일에 모든 GML 전자해도 셀들의 정보를 가진다. GML 카달로그는 XML 문법을 따르고 있어 XML 전용 데이터

베이스에서 검색하기가 용이하도록 만들어졌으며 요소마다 <cellinfo>, <scaleinfo>, <coorinfo>, <viewboxinfo> 네 가지의 그룹들을 가지는 쉬운 구조로 되어 있어 XQuery와 XPath 질의 언어를 쉽게 사용한다. 따라서 GML 카달로그는 질의에 의한 빠른 검색과 다중 레이어에 필요한 셀들을 더 쉽고 빠르게 탐색한다.

```

GML 카달로그 내용과 구조
1 <catalog>
2   <cell>
3     <cellinfo>
4       <filename>GB4X0000.000</filename>
5     </cellinfo>
6     <scaleinfo>
7       <scaleminimum>80000</scaleminimum>
8       <scalebasic>52000</scalebasic>
9     </scaleinfo>
10    <coorinfo>
11      <latitudesouth>-32.63333330</latitudesouth>
12      <longitudewest>60.76666670</longitudewest>
13      <latitudenorth>-32.31666670</latitudenorth>
14      <longitudeeast>61.33333330</longitudeeast>
15    </coorinfo>
16    <viewboxinfo>
17      <vby>-32.6333333</vbx>
18      <vbx>60.7666667</vbx>
19      <vbw>0.3166666</vbw>
20      <vbh>0.5666666</vbh>
21    </viewboxinfo>
22  </cell>
23  <cell>
24    ...
25  </cell>
26  ...
27 </catalog>

```

그림 4.7 GML 카달로그
Fig. 4.7 The GML catalogue

4.3 질의 처리

본 논문에서 전자해도 시스템 기본 질의는 위경도를 기본으로 한다. 뷰어에서 받은 사용자 입력 위경도 질의는 질의처리기로 전달되고 질의처리기는 데이터베이스에서 검색될 새로운 질의들을 산출하여 뷰어에 보여질 전자해도 셀들을 추출한다. 그림 4.8은 질의가 처리되는 전체 구조를 나타낸다.

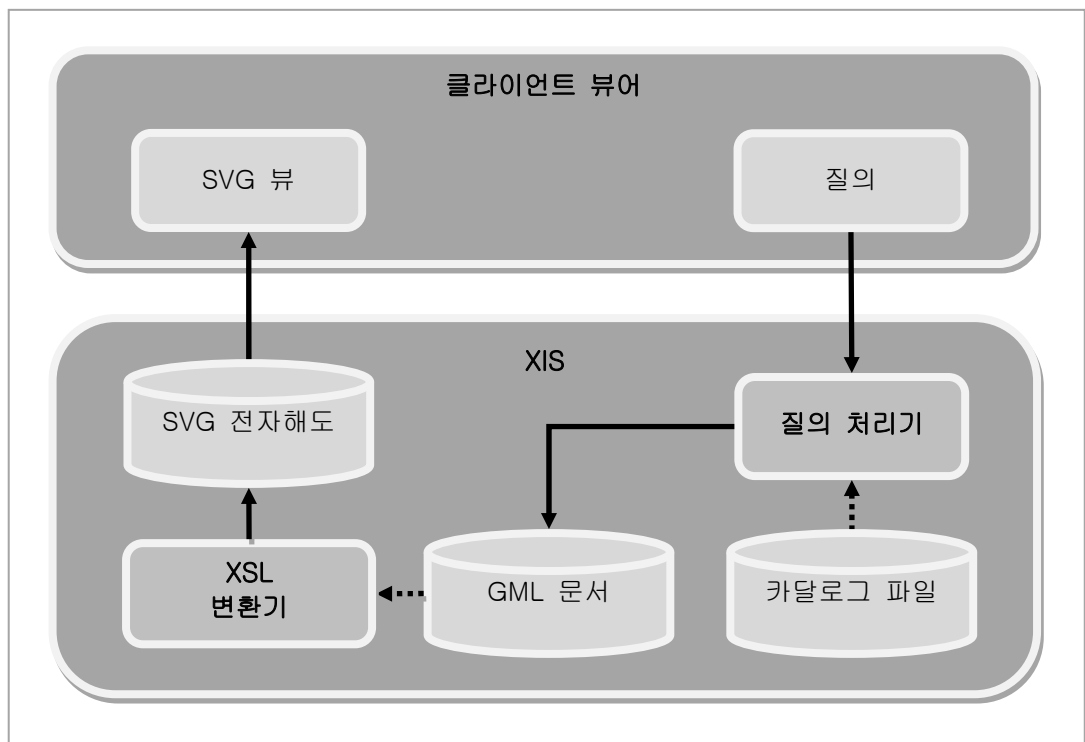


그림 4.8 질의 처리 과정
Fig. 4.8 Query processing

사용자로부터 입력 받은 위경도 질의는 데이터베이스 서버의 질의처리기로 보내지고, 위경도 질의는 서버에 미리 생성된 전자해도 셀 정보들을 담고 있는 카탈로그를 통해

결과 셀들을 찾는다. 데이터베이스로부터 해당 셀들에 대응하는 미리 변환된 GML 문서들을 추출하여 XSLT를 실행하고 SVG 문서로 변환하여 최종 사용자 브라우저에 전달한다.

(1) 질의 처리 과정

클라이언트 브라우저에서 접속한 뷰어를 통해 사용자가 직접 입력한 위경도 좌표 질의를 받으면 질의 처리기는 먼저 뷰어에 설정된 기본 축척과 셀 크기에 맞춰 질의 좌표를 중점으로 하는 새로운 셀 박스의 위경도 좌표를 계산해낸다. 셀 박스의 위경도 좌표는 뷰어의 기본 축척 크기와 비교하여 가장 남쪽인 경도와 가장 북쪽인 경도, 가장 서쪽인 위도와 가장 동쪽인 위도를 계산한다.

질의 처리기는 데이터베이스에서 셀 박스 좌표 범위에 속하는 객체들을 추출하여 새로운 GML 문서를 생성시키고 XSLT Translator에 미리 정의해둔 XSL 문서를 통해 SVG 문서로 변환 및 선행 작업한 객체들의 SVG 벡터 세이프들을 추가하여 클라이언트의 뷰어로 보내면 스타일시트 CSS 문서를 참조하여 전자해도를 표현한다.

그림 4.9는 질의의 처리 과정을 나타낸다. 질의를 입력 받는 뷰어는 사용자로부터 위경도 질의 좌표 (q_y , q_x)와 축척 질의 q_s , 뷰어의 '뷰' 가로 픽셀 길이 vw , 세로 픽셀 길이 vh 를 질의처리기로 전송한다. 전송된 질의들은 '뷰'의 가로, 세로 픽셀 길이와 사용자 축척 질의에 따른 최대, 최소 위경도 좌표값 ($Maxq(y)$, $Maxq(x)$) ($Minq(y)$, $Minq(x)$)를 산출해내고, 위경도 질의를 중점으로 하는 '뷰'의 위경도 단위 가로, 세로 길이 $Q(vw)$, $Q(vh)$ 를 계산한다. 새로 산출한 값들은 SVG 문서의 `viewBox` 값들이므로 데이터베이스의 캐시에 저장되어 SVG 문서가 생성될 때 저장된다.

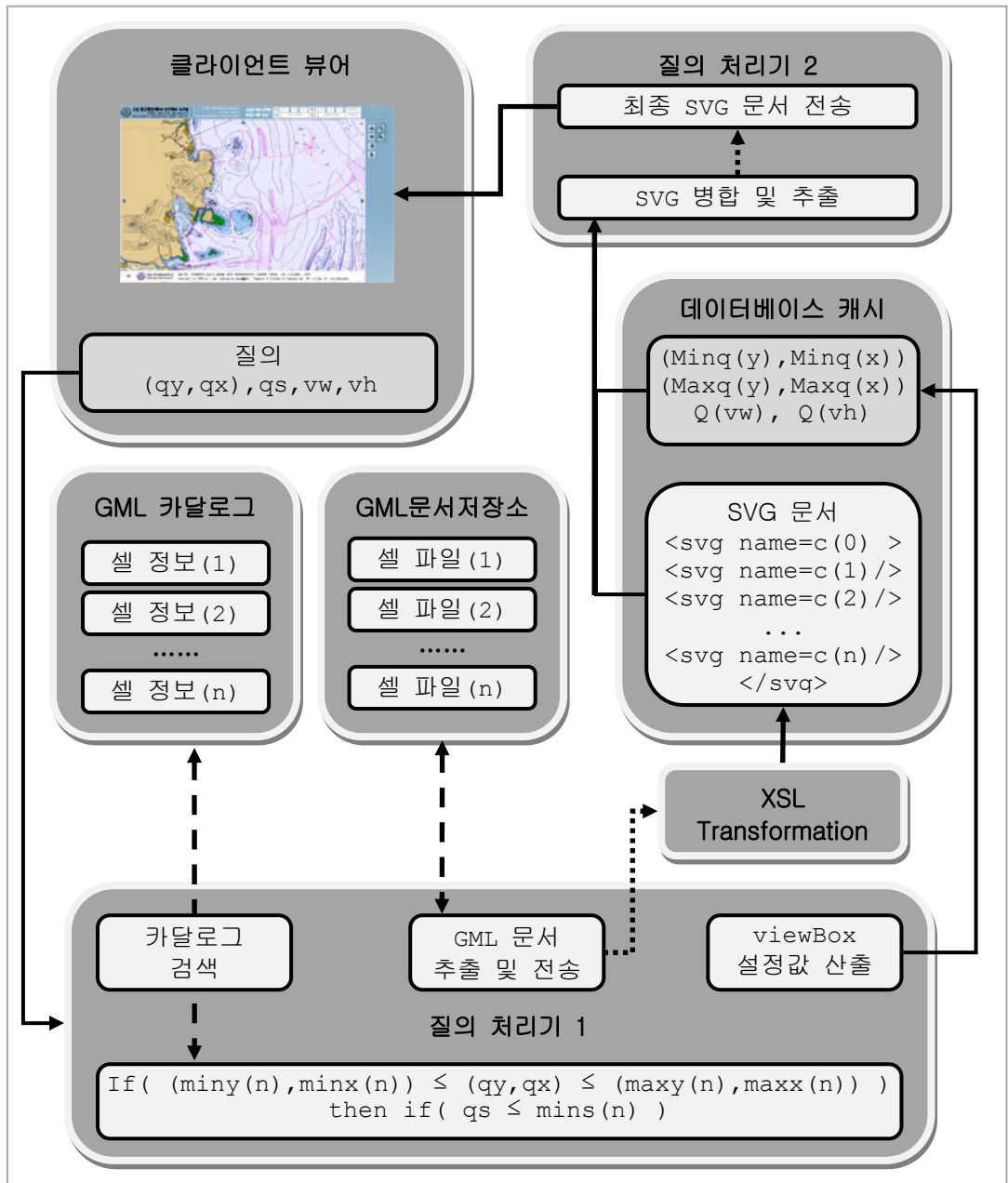


그림 4.9 위경도 질의 처리 과정

Fig. 4.9 Query processing for latitude and longitude

최대, 최소 위경도 좌표값은 카탈로그 파일에 작성된 셀들의 좌표값을 검색하여 해당 셀을 GML 문서가 저장된 데이터베이스로부터 추출한다. 위경도 질의 (Q(y), Q(x))가 카탈로그의 최소, 최대 좌표값 안에 포함된 셀들을 검색하고 질의 축척이 검색된 셀들의 최소, 최대 축척에 포함되는 셀들을 재검색한다. 그림 4.10은 XQuery로 작성된 사용자 질의를 나타낸다.

```

사용자 질의 XQuery 문
1 <result>
2 {
3   for $i in doc("catalog.xml")/catalogue/cell/
4   return
5     if ($i/coorinfo/[latitudesouth <= $qy <= latitudenorth
6         and longitudewest <= $qx <= longitudeeast])
7     then
8       if ($i/scaleinfo/[scalebasic <= $qs <= scaleminimum])
9       then
10        <Searched_Cells>
11          { $i/cellinfo/filename }
12        </Searched_Cells>
13      else
14        <Searched_Cells>
15          There is no cell for Scale you want
16        </Searched_Cells>
17    else
18      <Searched_Cells>
19        There is no cell for Coordinates you want
20      </Searched_Cells>
21  }
22 </result>

```

그림 4.10 사용자 질의 XQuery 문
Fig. 4.10 The XQuery code for a user query

줄 3에서 카탈로그 파일의 cell 엘리먼트까지의 구조를 \$i에 저장하여 사용함을 보여준다. 줄 4의 return 값은 XQuery의 if then 문을 사용하여 사용자의 위경도 질

의가 속하는 셀 파일이 존재할 때 줄 8에서 사용자 축척 질의가 기본 축척과 최소 축척 사이에 존재하는 셀 파일 이름들을 찾아 XML 구조로 반환한다. 셀이 존재하지 않을 때 줄 13과 17처럼 셀이 존재하지 않음을 표시하는 텍스트를 값으로 반환한다.

그림 4.11은 사용자 질의를 통해 XML 구조로 나온 결과값이다. XQuery 자체에서 결과값을 XML 구문으로 표현할 수도 있고 그림 4.11처럼 XML 전용 데이터베이스 자체 기능으로 표현된다. XIS는 XQuery와 XPath 질의어에 대해 결과값을 XML 구조로 표현하는 기능을 가진다.

```

사용자 질의 결과
1  <?xml version="1.0" encoding="UTF-8" ?>
2  <xlnxql:result xlnxql:count="6" xlnxql:type="node_list"
3  xmlns:xlnxql="http://www.exceloncorp.com/DXE/namespaces/query"
4    <filename>GB4X0000.000</filename>
5    <filename>BG5X01NE.000</filename>
6    <filename>GB5X01NW.000</filename>
7    <filename>GB5X01SE.000</filename>
8    <filename>GB5X01SW.000</filename>
9    <filename>GB5X02SE.000</filename>
10 </xlnxql:result>

```

그림 4.11 사용자 질의 결과
 Fig. 4.11 The result by a user query

하나 이상의 GML 문서가 추출된 경우, SVG 문서로 변환될 때 다중 레이어로 구성된다. 레이어 구성은 질의 축척이 클수록(축척이 클수록 작은 값을 가진다) 하위 레이어가 되고, 같은 축척의 셀인 경우 크기가 큰 셀부터 하위 레이어가 된다. SVG 문서 내에서 레이어의 레벨이 낮을수록 위쪽으로 배치되며 그림 4.9의 데이터베이스 캐시에

있는 SVG 문서에서 $c(1)$ 이 가장 하위 레벨이며 $c(n)$ 이 최상위 레벨이다.

그림 4.12는 사용자 인터페이스의 SVG 뷰에서 다중 레이어가 어떻게 표현되는지 보여주는 그림이다. 그림에서 C(1) 전자해도가 가장 하위에 있고 C(6) 전자해도는 가장 상위에 있음을 알 수 있다.

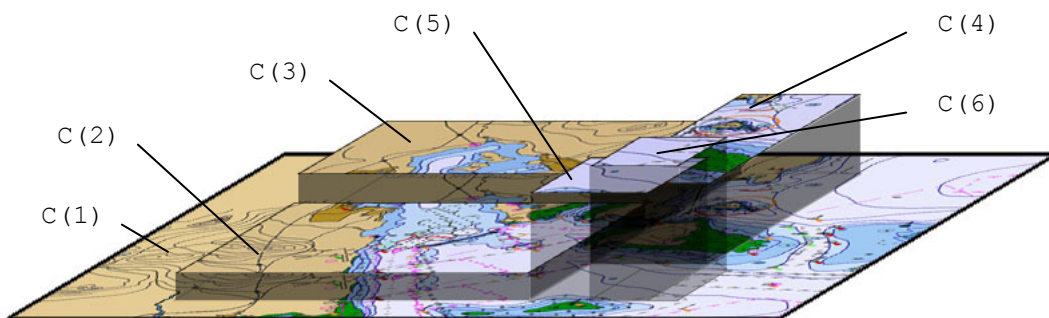


그림 4.12 다중 레이어 표현

Fig. 4.12 The representation for multi-layers

다중 레이어로 표현되는 결과값 전자해도는 SVG 문서 내부에서 쉽게 표현할 수 있는데 그림 4.13은 전자해도 벡터 이미지를 XML 구문으로 표현하고 있는 SVG 문서를 나타낸다.

사용자 질의 XQuery 문

```
1 <?xml version="1.0" standalone="no"?>
2 <!DOCTYPE svg>
3 <?xml-stylesheet type="text/css"
4 href="../../CSS/gml2svg_styles.css"?>
5 <svg name="C0_SUPERBG" width="800" height="600"
6   viewBox="-32.500000 60.866667 0.316666 0.566666">
7   <svg name="C1_GB4X0000" width="800" height="600"
8     viewBox="-32.500000 60.866667 0.316666 0.566666">
9     <g id="DepthArea" class="DepthArea">
10      <path id="DEPARE" d="M -32.493758 60.928285 ...>
11      ...
12    </g>
13    ...
14  </svg>
15  <svg name="C2_GB5X01NE" width="800" height="600"
16    viewBox="-32.533333 60.966667 0.083333 0.033333">
17    ...
18  </svg>
19  <svg name="C3_GB5X01NW" width="800" height="600"
20    viewBox="-32.500000 60.966667 0.050000 0.100000">
21    ...
22  </svg>
23  <svg name="C4_GB5X01SE" width="800" height="600"
24    viewBox="-32.566666 60.966666 0.033332 0.033334">
25    ...
26  </svg>
27  <svg name="C5_GB5X01SW" width="800" height="600"
28    viewBox="-32.566667 60.866667 0.066667 0.100000">
29    ...
30  </svg>
31  <svg name="C6_GB5X02SE" width="800" height="600"
32    viewBox="-32.566667 60.983333 0.028334 0.025000">
33    ...
34  </svg>
35 </svg>
```

그림 4.13 다중 레이어를 표현하고 있는 SVG 문서
Fig. 4.13 The SVG document represented multi-layers

변환된 SVG 문서는 데이터베이스 캐시에 저장되는데, 앞서 저장된 산출 값들을 viewBox 값에 병합하여 최종 SVG 문서가 생성된다. 캐시의 산출 값들은 그림 4.13

의 줄 4에서 이름이 C0_SUPERBG인 SVG의 viewBox 설정값이 된다. 이름이 C0_SUPERBG인 SVG는 전체 레이어들을 포함하여 viewBox의 기준점과 가로, 세로 길이로 뷰의 크기와 맞추는 역할을 한다. 데이터베이스 캐시에 저장된 최종 SVG 문서는 마지막으로 클라이언트 뷰어의 '뷰'에서 SVG 플러그인을 통해 출력된다.

제 5 장 결론 및 향후 연구 과제

쉽게 정보를 공유하고 획득할 수 있는 시대가 도래하면서 일반인들의 전문화가 가속화되고 일반인들의 지적 수준이 높아지면서 대부분의 전문 기술에 전문가가 아닌 사람들의 개입이 잦아지고 있다. 그에 따라 전자해도 기술에서도 일반인들이 쉽게 접근할 수 있도록 다양한 연구들이 진행 중이고 특히 GPS의 보급률이 높아지면서 일반인들이 출입 가능한 근해의 접근 경로들이 일반 지도에서는 표현하기 힘들어 전자해도의 필요성이 더욱 커지고 있다.

본 논문에서는 이러한 요구에 맞춰 전자해도 국제 표준인 S-57과 XML 기반 표준 언어만을 사용하는 전자해도 시스템을 설계할 때 변환에 대하여 발생할 수 있는 심각한 문제점들을 다루고 해결 방안을 제안한다. 이런 대부분의 문제점들은 S-57의 위경도 좌표 체계와 SVG의 스크린 좌표 체계 사이에서 발생하며 SVG의 viewBox 설정값의 변경과 XML 기반 GML 카탈로그의 작성 등으로 해결하였다. 또한 위경도 좌표, 축척, 뷰의 가로, 세로 길이 등 사용자 입력 질의를 XPath 형식의 XQuery 질의 언어를 사용하여 XML 전용 데이터베이스에 저장된 GML 카탈로그를 검색한다. GML 카탈로그에서 XML 구조로 이뤄진 결과값 셀 파일 이름들을 통해 XML 전용 데이터베이스에 저장된 GML 문서를 추출한다. GML 문서는 SVG 문서로 변환하는 모듈을 통해 데이터베이스 캐시에 저장되고, 사용자 인터페이스의 SVG 뷰 크기에 맞는 viewBox 설정값으로 초기 사용자 질의에 의해 데이터베이스 캐시에 저장된 값들을 변환된 SVG 문서의 최하위 SVG 태그에 삽입한다. 최종 생성된 SVG 문서는 사용자 인터페이스의 SVG 뷰를 통해 표현된다.

S-57 전자해도를 S-57 표준에 맞춰 GML 문서로 변환하고 XML 전용 데이터베이

스에 저장하여 효과적으로 전자해도를 관리 및 검색할 수 있다. 뷰어의 좌표 질의를 통해 검색된 전자해도는 질의처리를 통해 재계산된 좌표 체계를 가진 객체들로 구성된 SVG 문서로 추출 및 생성되어 다양한 플랫폼이나 다른 지리정보시스템과의 정보 교환이 가능하다. 뷰어에서는 검색으로 생성된 GML 문서를 XSLT와 CSS를 참조하여 SVG 문서로 표현함으로써 XML 기반의 기술만을 사용하기 때문에 타 시스템에서 재가공 및 확장이 가능하다는 이점과 여러 플랫폼에서 사용이 용이하다는 장점을 가진다.

향후 과제로는 다양한 좌표 체계를 연구 및 실험하여 S-57에 맞는 방법을 사용하고 S-57의 좌표들을 모두 미터 단위로 계산하여 픽셀당 축척을 더 정확하게 표현한다. XML의 장점이 다양한 플랫폼에서의 구현인 만큼 이를 실현하기 위해 윈도우즈 뿐만 아니라 리눅스 및 유닉스 운영체제에서부터 모바일, PDA 등 다른 플랫폼에서의 구현으로 GPS와의 연동이나 더 다양한 시점으로의 연구가 필요하다. 또한 좌표 검색과 함께 한글로 작성된 객체들을 질의하여 표현할 수 있게 함으로써 전자 지도 서비스와 일괄적인 사용이 가능하게 한다.

참고문헌

- [1] International Hydrographic Bureau, IHO Transfer Standard for Digital Hydrographic Data, Edition 3.1, 2000.
- [2] OpenGIS, Geography Markup Language (GML) Version 3.0, <http://www.opengeospatial.org>, 2003.
- [3] W3C, Scalable Vector Graphics (SVG) Version 1.1, <http://www.w3.org/TR/SVG11>, 2003.
- [4] 한국전산원, 지리공간 정보 인코딩(Encoding) 표준 개발에 관한 연구, 2002
- [5] W3C, XSL Transformations (XSLT) Version 2.0, <http://www.w3.org/TR/xpath>, 1999.
- [6] Galdos Incorporated, S-57 Schema and Related Tools Manual, S-57/GML, Project, 2004.
- [7] 국립해양조사원, “수로도서지(水路圖書誌)”, 2005.
- [8] 이성대, 강형석, 박휴찬, “전자 해도용 XML 스키마의 정의 및 변환”, 한국해양정보통신학회논문지, 제8권, 제1호, pp. 200-212, 2004.
- [9] 감승철, 이성대, 곽용원, 박휴찬, “GML과 SVG에 기반한 전자해도 시스템의 설계 및 구현”, 한국정보과학회 제32회 추계학술발표회 데이터베이스, 제32권, 제2호, pp. 127-129, 2005.
- [10] 감승철, 이성대, 곽용원, 박휴찬, “GML과 SVG를 사용한 웹 기반전자해도시스템의 개발”, 한국해양대학교 부설 산업기술연구소 연구논문집, 제23집, pp. 83-88, 2006.
- [11] 전성환, 이성대, 곽용원, 박철현, 박휴찬, “S-57 전자해도의 GML 변환 및 데이터베이스 관리”, 한국정보과학회 제33회 추계학술발표회 데이터베이스, 제33권, 제2호, pp. 230-234, 2006.

- [12] 이성대, 곽용원, 박휴찬, “객체관계형 데이터베이스에 기반한 XML 문서 저장 및 검색 시스템의 설계 및 구현”, 한국해양정보통신학회논문지, 제7권, 제2호, pp. 183-193, 2003.
- [13] Daniela Florescu, Donald Kossmann, “Storing and Querying XML Data using an RDBMS”, Bulletin of the Technical Committee on Data Engineering, Vol. 22, No. 3, pp. 27-34, 1999.
- [14] 배해영, “공간 데이터베이스 관리 시스템 보유 기술”, 전자공학회지, 제29권, 제12호, pp. 60-67, 2002.
- [15] Takeyuki Shimura, Masatoshi Yoshikawa, Shunsuke Uemura, “Storage and Retrieval of XML Documents using Object-Relational Databases”, Lecture notes in Computer Science, Vol. 1677, pp. 206-217, 1999.
- [16] eXcelon Incorporated, eXtensible Information Server Developer Guide Documentation, 2003.
- [17] Andreas Neumann, Andreas M. Winter, Time for SVG – Towards High Quality Interactive Web-Maps, Talk at ICA Congress, Beijing, China, 2001.
- [18] Matthew Austin, Creating a GIS from NOAA Electronic Navigational Charts, NOAA Coast Survey Development Laboratory, 2005.
- [19] 이동진, 김동오, 한기준, “GML 3.0 기반의 Web Feature Service 시스템 설계 및 구현”, 한국정보과학회, 제30권, 제2호, pp. 22-24, 2003.
- [20] 이혜진, 이현아, 김동호, 김진석, “GML을 적용한 이동체 관리 시스템의 설계 및 구현”, 한국정보과학회 제30회 추계학술발표회 데이터베이스, pp. 202-204, 2003.
- [21] 박유림, 이민수, “XQuery 생성을 위한 사용자 인터페이스”, 정보과학회 한국컴퓨터종합학술대회 논문집, Vol. 32, No. 1(B), pp. 220-222, 2005.