GML 지도 가시화 서비스 및 태깅을 이용한 POI 관리 도구 개발☆

Development of GML Map Visualization Service and POI Management Tool using Tagging

박 용 진* 송 은 하** 정 영 식***
Yong-Jin Park Eun-Ha Song Young-Sik Jeong

요 약

본 논문에서는 공통 지도 양식에 의한 교환, 지리정보의 상호 운영성을 위한 국제 표준 GML 기반 지도를 가시화하는 GML 지도 서버를 개발한다. 또한, GML 지도 서버는 동적 지도 분할 및 캐싱을 사용하여 효율적으로 GML지도를 모바일 디바이스에 전송한다. 모바일 디바이스에 실시간으로 지도를 제공하고 가시화하기 위해 모바일 디바이스의 가시영역을 기준으로 분할 관리하며, 분할 영역은 전송상의 이점을 위해 바이트화하여 전송한다. 그리고 모바일 디바이스에서는 수신된 분할영역을 조합한 후 모바일 디바이스의 디스플레이를 기준으로 4개의 가시영역으로 재분할하여 가시화를 하고, 자원의 효율적 운영을 위해 이전에 전송받은 지도의 중복성을 고려한 캐싱 알고리즘을 적용하여 관리한다. 지도의 밀집 지역에 대해서는 전송시간의 지연을 방지하기 위해 적응적 지도 분할 메커니즘을 제안하여 전송시간을 일정함을 유지한다. 본 논문에서는 WIPI 환경에서 모바일 디바이스들의 위치 추적 모니터링 기능을 제공한다. 즉 필드 에뮬레이터를 만들어 모바일 디바이스들을 생성, 위치 이동을 시키고 위치들을 추적한다. 비쥬얼 인터페이스를 사용하여 GML 기반 POI 정보들의 계층적 관리와 개인화 태깅 기술에 의해 효과적인 POI 정보 생성, 수정, 저장 및 검색 기능을 제공하는 POIM(POI Management)를 구현한다.

Abstract

In this paper, we developed the GML Map Server which visualized the map based on GML as international standard for exchanging the common format map and for interoperability of GIS information. And also, it should transmit effectively GML map into the mobile device by using dynamic map partition and caching. It manages a partition based on the visualization area of a mobile device in order to visualize the map to a mobile device in real time, and transmits the partition area by serializing it for the benefit of transmission. Also, the received partition area is compounded in a mobile device and is visualized by being partitioned again as four visible areas based on the display of a mobile device. Then, the area is managed by applying a caching algorithm in consideration of repetitiveness for a received map for the efficient operation of resources. Also, in order to prevent the delay in transmission time as regards the instance density area of the map, an adaptive map partition mechanism is proposed for maintaining the regularity of transmission time. GML Map Server can trace the position of mobile device with WIPI environment in this paper. The field emulator can be created mobile devices and mobile devices be moved and traced it's position instead of real-world. And we developed POIM(POI Management) for management hierarchically POI information and for the efficiency POI search by using the individual tagging technology with visual interface.

☞ keyword : GML, 지도 가시화(map visualization), 계층적 POI 정보 관리(hierarchical POI information management), 태깅(tagging), 동적 지도 분할 및 캐싱(dynamic map partition and caching)

* 준 회 원 : 원광대학교 대학원 컴퓨터공학과 졸업(석사)
yjpark1@wku.ac.kr
** 정 회 원 : 원광대학교 전기전자 및 정보공학부 전임강사
ehsong@wku.ac.kr
*** 종신회원 : 원광대학교 전기전자 및 정보공학부 교수
ysjeong@wku.ac.kr
[2008/01/29 투고 - 2008/02/11 심사 - 2008/02/25 심사완료]
☆ 이 논문은 2007년도 원광대학교의 교비 지원에 의해서 수행됨

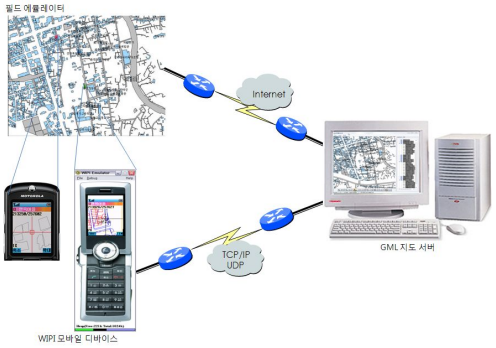
1. 서 론

최근 정보통신기술의 발전과 유비쿼터스 환경은 현대 생활에 많은 편리함을 제공한다. 특히, 모바일 기술은 핸드폰, 네비게이션, 홈 네트워크 등의 현 시대를 대표하는 하나의 아이콘으로 부각되고 있다. 하지만, 현재 LBS 기반 지도 서비스는 지도 데이터에 대해 각 개발 업체마다 서로 다른 포맷을 사용하여 서비스하고 있어 지도 데이터들 간에 자유로운 활용, 통합 및 응용에 있어서 상당히 불편하다. 따라서 공통 교환 포맷 표준과 서로 상이한 하드웨어와 소프트웨어 환경에서 지리정보의 상호 운영성을 제공하기 위하여 지리정보 산업체들이 주축이 되어 만든 민간 GIS 표준 기관인 OGC(Open GIS Consortium)에서 공간적 또는 비공간적 속성을 지닌 지리 사상(feature)을 포함하는 지리정보 전달, 저장을 위한 지리정보 인코딩과 지리 정보나 위치 기반 서비스 정보를 플랫폼과 클라이언트 종류에 상관없이 제공할 수 있도록 해주는 XML 기반 표준 인코딩 언어 GML 명세를 제시하였다[1,2,3].

GML은 확장성과 유연성이 뛰어나 좀 더 효율적인 데이터 처리 및 관리가 가능하다. 그러나 표준명세인 GML은 ASCII 포맷으로써 기존 바이너리 형식의 포맷에 비해 파일이 무겁다는 단점이 존재한다. 이는 데이터 처리 및 관리에 있어서 시스템의 고사양이 필수가 되고 특히 모바일 서비스는 네트워크상에 존재하는 트래픽에 의해 지도 전송이 용이하지 못하며 전송된 지도에 대해서도 처리를 위한 모바일 디바이스의 성능 한계를 극복하기 어렵다. 또한 기존 LBS는 지도 데이터 전체를 모바일 기기에 탑재하는 방식을 사용하는데 이는 도로 및 건물들이 시시각각 변화하는 최근 상황에 비추어 볼 때 잦은 업그레이드를 요구하므로 사용자로부터 불편함을 야기한다[3,4].

또한, GIS는 웹과 연동하여 지도 정보와 POI(Point Of Interest) 서비스를 제공하고 있다. POI는 건물, 도로, 상점, 관공서, 병원 등의 지도 내에서 사용자가 관심 있어 하는 개체를 의미한다. 이를 통해 사용자는 관심을 가지는 곳이나 필요한 정보를 쉽게 얻을 수 있다. 이러한 POI 정보를 지도 서비스에 추가함으로써 단순한 지도 가시화가 아닌 지도를 구성하는 각 개체들의 정보를 확인함으로써 효율성과 활용도를 향상한 GIS 서비스를 제공하게 된다. 그러나 현재 GIS 서비스는 기반이 되는 지리공간데이터의 표현과 POI 정보를 제공하는데 있어서 표준화가 이루어지지 않고 있는 실정이다. 따라서 서비스 개발업체들은 서비스 제공방법, 서비스 구축, 지도 표현 형식에 있어 특정 시스템이나 환경에 의존적이며 그에 따른 제약이 많은 형편이다. 특히 POI 서비스는 해당 개체의 정보가 불충분하여 단순한 지명 서비스 정도이다. 따라서 POI 서비스 사용자는 요구하는 정보에 알맞은 체계적인 검색 기능을 제공하지 못하고 있다[5,6].

본 논문에서는 GML 지도 서버 구축, 모바일 디바이스에 GML 지도를 효율적으로 공급하고 가시화하는 시스템을 구축한다. 먼저, GML 지도 서버는 GML 기반의 지도를 파싱하여 가시화하고 지도 서비스를 위한 기본적인 기능, 확대, 축소, 이동 및 모바일 디바이스 추적 모니터링 기능을 포함한다. 또한, POI 정보처리를 위해 태깅 기술을 활용하여 POI 정보들을 계층적으로 표현하고 실제 GML로 통합되는 POIM(POI Management)을 제안한다. 다음 그림 1은 본 논문에서 제공하는 GML 기반 지도 서비스를 위한 전체적인 운영 개념도이다.



(그림 1) GML 기반 지도 서비스 운영 개념도

GML 지도 서버는 GML 지도를 가시화하고 처리하는 기능, 모바일 디바이스 위치 추적 모니터링 및 가시화 기능, POI 정보처리를 위한 POIM 관리 도구를 포함한다. WIPI 모바일 디바이스는 GML 지도 서버로부터 지도를 실시간으로 전송받아 가시화하는 기능을 내포한다. 필드 에뮬레이터는 GML 기반으로 가시화된 지도에서 모바일 디바이스들을 생성하여 GPS 좌표 중심으로 이동할 수 있는 기능을 포함한다. 필드 에뮬레이터에서 생성된 모바일 디바이스의 좌표 정보는 GML 지도 서버와 동기화 되어 그 이동경로에 대한 추적 모니터링이 가능하다.

2. 관련연구

2.1 GML 기반 지도 서비스

GML 기반 지도 가시화 시스템들은 eSpatial Inc.의 iSMART Explorer 4.4, Snowflake Software Ltd.의 OS Master Map Viewer 2.0, TatukGIS Inc.의 TatukGIS Viewer 1.4[7], SafeSoftware Inc.의 FME Universal Viewer 2003[8], ETRI의 GML Viewer[9] 등이 있다.

iSMART Explorer 4.4는 어플리케이션 자체가 가벼우며, 쉽게 사용할 수 있고 OCI(Overseas Consultants Incorporated) DB에 연결되어 분석에 유리하다. 또한, 자동적으로 스키마를 발견하여 비공간 속성 데이터를 재검토 할 수 있으며, 확대, 축소, 이동 모두가 가능하다. JAVA로 개발된 OS Master Map Viewer 2.0은 WinZip, GZip, GML 파일을 초당 1900점을 읽을 수 있으며, 하나 혹은 다수의 점을 선택하여 속성별로 비교하며 가시화한다. 또한, 테마별로 통계 및 가시화할 수 있고, 마우스 또는 키보드 조작으로 가시화 영역을 바꿀 경우 변경된 이미지 표현은 점선으로 라운드 형성 후 다시 그리는 방법을 사용한다. TatukGIS Viewer 1.4는 지원하지 않은 포맷이 없을 정도로 래스터(raster) 이미지 파일 포맷, 벡터 그래픽 파일 포맷 등 거의 모든 파일을 지원한다. 가시화된 지리 정보를 PDF 파일로 변경할 수 있도록 했으며, 점의 구성이 이루는 크기, 지점간의 거리, 해당지역을 측정할 수 있으며, 속성 처리는 SQL 빌더로도 할 수 있다. 벡터 속성 정보를 기반으로 지도의 서로 똑같은 속성은 같은 색으로 표현한다. FME Universal Viewer 2003는 뷰어에서 복수의 점에 대한 측정거리, 또 기하학 정보를 보여주며 해당 파일의 데이터 셋을 보여준다. 각각의 속성은 다른 의미로서 GUI로 구별할 수 있으며, *.fmv로 저장 후 다른 포맷 전환이 가능하다. 그 밖에 여러 GML 기반 지도 서비스가 있으나, 이들 가시화 시스템은 단순히 지도 가시화만 해줄 뿐 실질적인 POI 정보 서비스는 제공하지 않는다. 또한 이 시스템들은 GML 파일 내의 속성들만을 보여주기 때문에 전문 개발자들의 지도 편집에 중점을 두어 실제 GIS를 필요로 하는 일반 사용자들이 사용하기에 편리한 GUI를 제공하지 않는다. 이러한 가시화 시스템들은 단순히 지도 포맷을 보여줄 뿐 이동객체(모바일 디바이스를 가진 사용자, 모바일 디바이스 자체 등을 의미)들의 이동을 추적하지 못한다. 따라서 본 논문에서는 가시화한 시스템에서 이동객체들의 이동을 통합 추적 관리하는 컴포넌트를 구축한다. 또한, WIPI 환경에서 모바일 디바이스에 GML 기반 지도를 가시화 기능도 구현한다. 이때, 모바일 디바이스로 효율적인 GML 지도 전송을 위한 지도 분할 및 캐싱 방법도 제시한다.

2.2 기존 POI 지도 서비스

POI 구축과 POI 서비스 순으로 기존의 지도 서비스에 관한 내용을 소개한다. 먼저, POI 구축은 소정의 카테고리별로 분류하여 데이터베이스에 저장하고 분류된 카테고리별로 단순히 지도 위에 TIP 형태로 가시화하든지, DB 검색을 통한 테이블 형태로만 가시화하고 있다. 또한 POI 파일 문서별로 모듈화하고 이를 바탕으로 자동차

네비게이션시 명칭검색, 지역검색 등을 통하여 경로 안내 목적지 표시에 활용하는 가시화 기술이 있다. 이들은 POI 자체에 대한 가시화에 치중하고 있어 단순한 가시화 기술만 제공한다[10].

기존의 POI 서비스들을 주로 모바일과 PDA 환경을 위주로 개발되었다. 모바일과 PDA의 서비스들은 파일 경량화로 인해 상세한 정보를 나타내기에는 역부족이며 사용자의 편의성을 고려하는데 한계가 있다. 지도 가시화 측면에서는 도로와 건물의 색이 구분되지 않아 사용자가 알아보는데 매우 불편하며, 상하좌우 이동과 같은 대부분의 제어가 키패드와 메뉴를 선택한 이동만 가능하게 설계되었다. 특히 POI 정보는 상호명만을 보여주는 단순한 지명 서비스에 한계를 보이며, 실제 사용자가 필요로 하는 전화번호, 주소, 홈페이지, 특징 등의 정보는 제공하지 않는다.

웹 상에서 POI 서비스를 해주는 대표적인 시스템으로는 콩나물닷컴이 있다. 이 시스템에서는 건물명이 주요 건물에만 표시가 되고 확대 축소를 하여도 상세한 건물명은 나오지 않아 사용자가 위치를 찾아 가거나 주변 정보를 살펴보는데 있어 많은 불편하다. 또한 검색을 위해 건물명을 선택할 경우, 상세 POI 정보를 제공하지 않는다.

본 논문에서는 POI 분류 정보 및 POI 정보 가시화를 GML 기반 지도와 조합된 형태로 제공하여 보다 효율적이고 사용자 위주의 인터페이스 기능을 강조한다. 즉, 도로와 건물은 각기 다른 색을 적용하여 사용자에게 식별성을 높여 준다. POI 정보는 건물을 선택할 경우 상호명, 전화번호, 주소 그 건물의 특징, 홈페이지 등의 상세 정보를 제공한다. 또한 사용자의 빠른 정보 검색을 위해 계층적 POI 정보 구축과 함께 사용하기 편리한 검색창을 제공한다. 특히, 기존의 POI 지도 서비스들은 지리데이터의 표준화가 이루어져 있지 않아 서로간의 상호운용이 불가능하여 데이터의 중복 구축의 단점을 GML 기반으로 보완한다.

2.3 태깅 기술과 POI 서비스

태깅은 현재 웹 2.0을 적용한 사이트들에 의해 지원되고 있다. 태깅 기술의 대표적인 사이트로는 Flickr[11], del.icio.us[10], Technorati[12] 등이 있다.

Flickr는 현재 전 세계적으로 큰 인기를 얻고 있는 이미지 공유 사이트이다. 사용자는 자신의 이미지를 업로드하면서 이를 설명하는 태그를 부착 가능하다. 업로드 할 사진의 지리적인 위치 정보를 담고 있는 위치 정보 태그를 추가할 수도 있다. 그리고 자신이 찾고자 하는 이미지를 태그를 활용하여 검색할 수 있다. 태그 검색으로는 이진 표현 검색인 and, or, not등의 연산자를 지원하며 태그 이외의 텍스트 검색도 지원한다. http://del.icio.us는 자신의 즐겨찾기를 저장하고 타인들과 공유하기 위한 소셜 북마킹(social bookmarking) 사이트이다. 사용자는 즐겨찾기를 추가하며 태그를 입력한다. 개개의 사용자들은 같은 URL을 북마킹하면서 서로 다른 태그를 자유롭게 사용할 수 있다. 태그 검색으로는 모든 and, or, not과 xor의 이진 표현 검색을 지원한다. Technorati는 태그 기반의 블로그 검색엔진이다. 사용자는 Technorati에서 자신의 블로그가 검색되도록 하기 위해서 자신의 블로그를 등록하고 인증을 받는다. 그리고 사용자가 블로그에 글을 쓸 때 클레임링크를 글에 삽입하면 Technorati는 크로울링을 통해서 태그와 블로그를 수집한다. 클레임링크는 Technorati에서 태그를 쉽게 수집하기 위해 사용하는 메타데이터이며 HTML의 형태로 되어 있다. 이러한 태깅 서비스들은 태그를 활용한 정보의 분류에 의해 사용자 기준으로 자유로운 연산 작용을 통해 다수의 키워드를 이용하여 데이터를 분류함으로써 기존의 정해진 분류 체계들을 따를 필요없이 각 사용자들의 개성을 반영하는 분류체계를 지원한다. 또한, 정보검색에 있어서도 각 사용자의 기준에 맞는 분류체계를 유지하기 때문에 동의어에 대한 검색에 오류를 범하지 않으며, 기존의 트리 구조 검색 방법이 아닌 태그를 통한 수평적 검색 등의 신속하고 정확한 검색 기능이 가능하다. 본 논문에서는 이러한 태

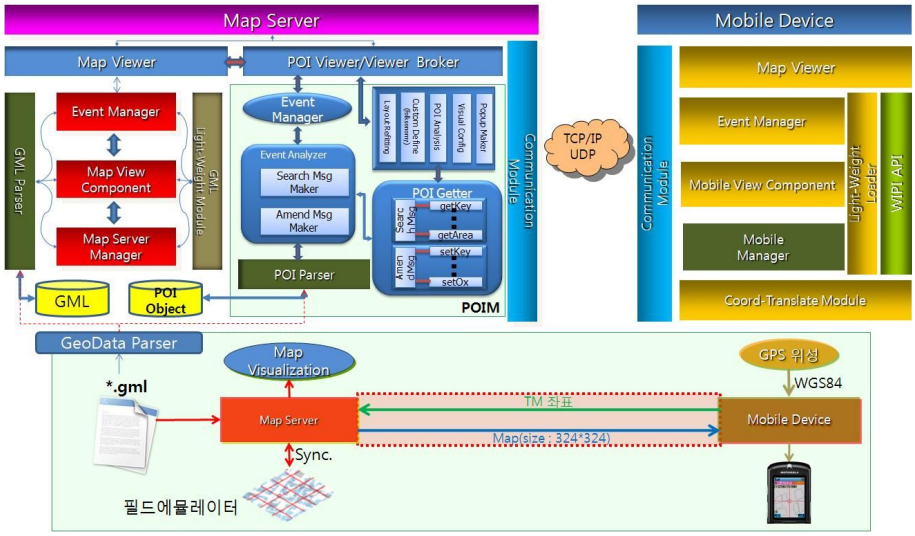
깅 기술을 POI 정보 서비스에 적용하여 POI 정보 분류 및 검색에 활용한다. 이는 POI 태깅 정보를 XML 파일로 생성하여 사용자들 간에 정보 공유를 지원하게 되는 폭소노미(Folksonomy)가 실현된다. 기존의 태깅 기술은 그 자체로써 발전되어 오고 있고 POI 서비스에 접목한 경우는 거의 없다.

2.4 이동객체 모니터링 가시화

기존의 모니터링 시스템은 크게 가시화를 하면서 추적하는 모니터링과 가시화 없이 단순한 표형태의 데이터로만 모니터링하는 시스템으로 구분된다. 먼저 가시화를 통해 모니터링하는 시스템의 대표인 (주)사이버맵월드 위치추적 솔루션은 차량, 물류 및 고객 등 위치추적 관리가 필요한 기업체, 관공서를 대상으로, 차량 및 고객의 위치정보, 이동 경로 등의 상세정보를 GIS 시스템을 이용하여 실시간으로 추적하기 위한 맞춤형 솔루션이다[13,14]. 대용량의 이동객체 관리를 위한 LBS 엔진 및 메인 메모리 DB 기술이 적용된 GIS 엔진, 모든 축척을 포함하는 상세 DXF 파일을 포함한 LBS 서버를 기반으로 솔루션이 공급되는 시스템이다. 대규모의 이동객체 동시 추적 및 메인 메모리 DB 기술로 월등한 빠른 검색 속도를 제공하며, 전체 차량 및 사용자 검색, 추적이 가능하다. NEO 관제 시스템은 접수 창에 콜을 접수 받으면 접수리스트에 배차 전까지의 콜 목록을 표시하며, 배차 후에는 배차리스트에 승객이 승차 혹은 취소할 때까지의 상태를 표시하여 관리한다. 위치를 확인하고 하는 호차를 입력하면 호차의 최근 위치 정보가 DXF 지리 정보 데이터를 기반으로 나타나며 데이터의 전송 주기와 전송 횟수는 상황에 따라 관리자가 관리할 수 있는 시스템이다. 하지만 이러한 모니터링 시스템들은 지리정보 없이 혹은 표준화되지 않은 지리정보 데이터를 가지고 가시화를 실시하여 추적 관리를 한다. 따라서 본 논문에서는 국제적으로 권고되는 표준안인 GML 포맷 형태를 기반으로 모바일 디바이스들의 이동 궤적을 추적, 가시화하는 컴포넌트를 개발한다.

3. GML 지도 가시화 서비스

본 절에서 먼저, GML 기반 지도 가시화, 태깅



(그림 2) GML 기반 지도 서비스 시스템 전체 구조도

기술을 이용한 POI 정보관리도구, 이동객체 위치 추적 모니터링 및 모바일 디바이스로의 지도 서비스에 대한 전체적인 시스템 구조도는 그림 2와 같다. 3절에서는 GML 지도 서버에서 제공하는 지도 처리에 대한 기능과서 모바일 디바이스에서 제공되는 기능을 제시한다. 4절에서는 GML 지도 서버 내 POI 정보표현, 처리를 위한 POIM에 대한 기능들을 소개한다.

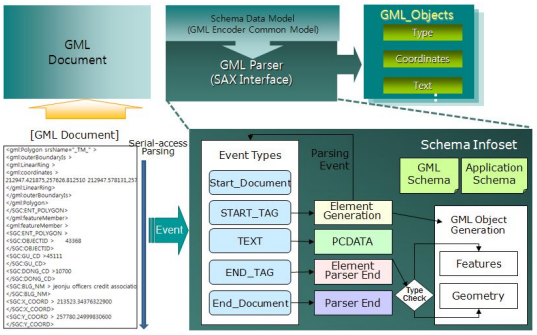
3.1 GML 지도 서버에서의 가시화 방법

GML 문서를 분석하여 속성 추출을 담당하는 GML 파서와 추출된 데이터를 통해 지도를 가시화하는 지도 가시화 컴포넌트, 지도의 이벤트 처리를 위한 Event Manager, 지도를 분할 관리 및 모바일 디바이스로의 전송을 담당하는 Map Server Manager, 지도 전송시 GML 경량화를 위한 GML Light-weight 모듈로 구성된다.

3.1.1 GML 파서

GML 파서는 XML 파서를 통해 GML 파싱을 유도한다. GML 파서는 SAX에서 제공하는 표준 라이브러리 모듈 중 XMLParser 클래스를 상속받아 GMLParser 클래스를 정의한다. GMLParser 클래스는 GML Schema 기반으로 각 엘리먼트들을 정의하고, 각 엘리먼트들을 처리하기 위한 이벤트 형태로 정의한 핸들러를 호출한다.

GMLParser 클래스는 엘리먼트를 키/값 형식으로 태그를 처리한다. 키는 태그 이름[<,>, 생략]이고, 값은 시작 태그와 종료 태그를 실행할 함수 이름이며 튜플로 구성된다. 예를 들어, <gml id="2" type="simple"> 태그일 경우, GMLParser 클래스는 gml_start_tag[{'id':'2', 'type'='simple'}]을 호출한다. 함수 호출 방식은 시작 태그의 경우에는 handle_starttag 함수를 호출하며 종료태그의 경우 handle_endtag 함수를 재정의 하면서 GML의 루트 엘리먼트에서 파일의 끝까지 반복한다. 파싱이 종료되면 지도 가시화 및 모바일 전송을 위해 객체 형태로 저장하게 된다. GML_Object는 객체의 Geometry Type, 객체의 중심좌표, 그리기 위한 좌표들로 구성되어 저장된다. 그림 3은 GML 파싱 과정을 나타내고 있다.

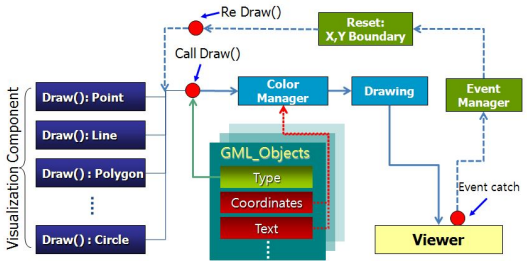


(그림 3) GML 파서 구조

3.1.2 지도 가시화 모듈

지도 가시화 모듈은 GML 파서에서 생성된 GML_Object의 지도 속성들을 가지고 Viewer에 가시화하며, 유저 이벤트에 대한 처리도 담당한다. 먼저 지도 가시화를 위해 GML이 표현 가능한 Geometry Type들에 대해 각 Type별 Draw()라는 이미지 프로세싱 메소드들을 미리 구현하고 Visualization Component라는 그룹으로 패키지화한다. 지도 가시화는 GML_Object의 Type 속성을 가지고 Visualization Component에 접근해 해당 Type에 매칭되는 Draw() 메소드를 호출한다. 호출된 Draw() 메소드는 가시화 과정에서 가시화시 사용자로부터 건물과 도로의 식별성을 두기 위해 각 Color 설정값을 갖고 있는 Color Manager에서 Color를 설정한다. Color 설정이 끝나면 실제 Viewer에 가시화를 위해 GML_Object의 Coordinates와 Text를 가져와 가시화 작업을 수행한다. 또한, 가시화 모듈은 사용자 이벤트 처리를 위해 Event Manager를 두고 이벤트가 발생되면 해당 이벤트를 분석하여 원하는 이벤트에 맞는 X, Y 경계값을 설정하고 reDraw() 메소드를 호출해 가시화 루틴을 다시 한 번 수행한다. 그림 4는 GML 기반

지도를 가시화하는 과정이다.

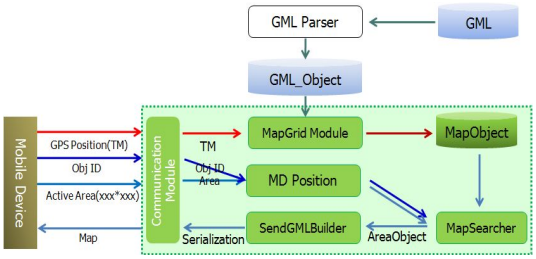


(그림 4) GML 지도 서버에서의 가시화 과정

3.1.3 지도 서버 관리자

지도 서버 관리자는 GML을 분할 관리하며 지도 요청에 대해 특정 분할영역을 전송한다. 지도 분할 및 캐싱을 위한 이론은 다음 3.1.4절에서 설명한다.

모바일 디바이스에서 좌표, 액티브 영역과 오브젝트 ID를 전송해오면 이를 커뮤니케이션 모듈에서 수신하고 좌표와 오브젝트 ID는 MD Position으로, 액티브 영역은 GridGML 모듈에 전송한다. 액티브 영역을 전송받은 MapGird 모듈은 GML 가시화에 사용된 GML_Object에 접근하여 액티브 영역 크기별로 지도를 분할하고 MapObject로 저장한다. 모바일 디바이스의 좌표와 보유 오브젝트 ID를 전송받은 MD Position은 MapSearcher에 보내 MapObject로부터 재전송 영역 추출 및 중복성 제거를 수행하고 이를 SendGMLBuilder에서 바이트화하여 모바일 디바이스로 전송한다. 그림 5는 지도 서버 관리자의 수행 과정이다.

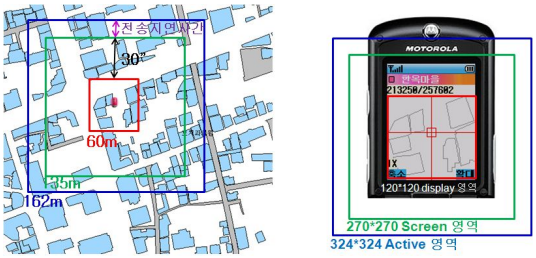


(그림 5) 지도 서버 관리자의 수행 과정

3.1.4 동적 지도 분할 및 캐싱

본 논문에서 동적으로 지도를 분할하고 캐싱을 위해 기본적으로 디스플레이 크기가 120*160의 Aroma WIPI Emulator 환경을 기반으로 지도 서버와 모바일 디바이스간 지도 전송 및 가시화를 위해 분할 및 캐싱 기법을 설계한다. 디스플레이 화면에서 지도 표현영역은 줌 레벨 1일 경우 가로 세로 각각 60m를 표현 영역으로 한다. 또한 이동 속도의 경우 성인 남자가 4km/h의 속도로 걷는다는 가정하에 동적 지도 분할 및 캐싱 기법을 제안한다.

(a) 가시 영역 분할



(b) 스크린 영역과 액티브 영역 추출

(그림 6) 지도 영역 분할 및 추출

㉮ 지도 분할 영역 추출

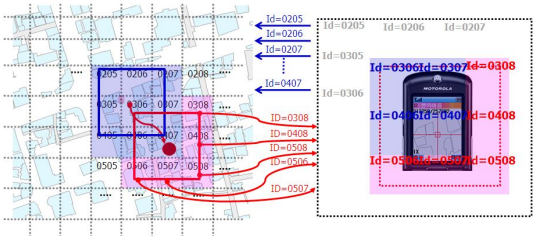
본 논문은 모바일 디바이스에 가시화를 목적으로 하고 있기 때문에 모바일 디바이스의 디스플레이 크기를 기준으로 4개의 영역으로 분할한다. 4개의 영역은 디스플레이 영역, 스크린 영역, 액티브 영역, 캐쉬 영역으로 구성된다. 디스플레이 영역은 모바일 디바이스의 화면 크기에 해당하는

영역이며, 스크린 영역은 지도를 나타내는 영역이고, 지도 재전송의 기준이 되는 영역, 액티브 영역은 지도 재전송에 따르는 지연시간을 위한 예비 영역, 캐쉬 영역은 과거에 받아놓은 지도의 재활용을 위해 일정기간 저장해 놓는 영역이다.

스크린 영역과 액티브 영역은 모바일 디바이스의 디스플레이 영역을 기준으로 추출한다. 디스플레이 영역은 120*160 크기이나 인터페이스 부분을 뺀 120*120 크기를 디스플레이 영역이라 정의한다. 지도의 재전송 타이밍을 30초로 잡고 30초간 재전송 없이 이동할 수 있는 영역 270*270(실제 지도 크기 : 135*135)을 스크린으로 추출한다. 스크린 영역을 벗어나 재전송이 이루어지면, 지도를 다운로드 하는 동안에 실시간 가시화를 위해 액티브 영역을 둔다. 액티브 영역은 스크린 영역 대비 120%의 영역으로 두고 있으며, 재전송에 대한 15초 정도의 지연시간을 보완시킨다. 서버로부터 지도를 전송받는 크기는 액티브 영역에 해당하는 크기를 전송받는다.

모바일 디바이스의 액티브 영역이 결정되면 서버의 지도 분할 기준이 결정된다. 이는 본 논문의 기준이 모바일 디바이스이므로 분할 방법이 결정된 후에 이에 맞는 최적의 크기를 지도 서버에서 전송해야 한다. 지도 분할은 모바일 디바이스로의 전송 시 최상(best case)와 최악(worst case)을 고려하여 분할한다. 지도 서버는 액티브 영역 대비 1/4의 경계값을 기준으로 분리한다. 최상시 4개의 분할 영역이 전송되고, 최악시는 9개 영역이 전송되도록 한다. 좀 더 세분화하여 지도를 분할할 경우 최상과 최악의 경우 차를 더 줄일 수는 있으나, 그럴 경우 지도의 줌 레벨 1에 포함되는 객체들의 수가 너무 적어지는 경우가 생길 수 있고 큰 도로의 경우에는 객체가 포함되지 않는 경우가 생길 수 있다. 이에 액티브 영역 대비 1/4의 경계값이 객체들이 고르게 분포하는 최적의 크기이며, 전송 시간을 고르게 분포하도록 한다. 지도 분할 방법은 하나의 영역 경계 값이 결정되면 지도 전체의 경계 값에서 하나의 영역 경계 값으로 나누고 객체의 중심 좌표가 영역 경계 값의 포함 여부를 판단하여 분할한다. 그림 7은 모바일 디바이스를 고려한 서버의 지도 분할 방법에 대한 것을 나타내고 있다.

(그림 7) 서버의 기본 지도 분할



(그림 8) 중복성을 제거한 지도 캐싱 기법

㉯ 중복성 제거를 위한 지도 캐쉬

중복성을 제거하기 위해 서버에서 지도 분할 시 각 영역별로 ID를 두고 관리한다. 그림 8과 같이 모바일 디바이스에서 스크린 영역을 벗어나 지도 재전송을 요청하면, 현재 모바일 디바이스가 보유하고 있는 영역의 ID와 중복성을 체크 후 중복되지 않는 부분들에 대해서만 전송한다. 모바일 디바이스에서는 새롭게 받아온 영역들과 기존에 존재하던 영역들을 조합하고, 재영역 분할을 통해 액티브 영역에 포함되지 않는 영역에 대해서는 캐쉬에 저장해 놓는다. 캐쉬의 크기는 상하좌우 각각 한 번씩의 재전송을 보완할 수 있는 크기로 설정한다.

㉰ 적응적 지도 분할

적응적 지도 분할은 전송시간 지연을 방지하기

위한 기법이다. 서버의 지도 분할시 데이터의 크기별로 분할하지 않고, 특정 영역으로 분할하기 때문에 그림 9에서 보이는 바와 같이 객체들이 밀집된 지역과 그렇지 않은 지역으로 구분되어 질 수 밖에 없다. 이는 데이터 전송에 있어서 전송 시간이 일정치 않을 수 있을 뿐만 아니라, 객체들이 매우 밀집된 번화가에 대해서는 전송 시간이 너무 지연되는 경우가 존재한다. 이를 방지하기 위해 밀집된 지역에 대해서는 재분할을 수행하여 전송한다. 재분할 기준은 전송시간을 기준으로 요청시간과 전송 완료시간을 비교하여 평균치를 내고 이 평균치를 크게 벗어날 경우에 서버에 전송 요청하는 영역에 대해 재분할하여 전송하도록 요청한다. 이에 서버는 포함되는 영역의 분할 기준을 한 단계 높인, 기존 액티브 영역 대비 1/4 크기에서 1/9 크기로 좀 더 세분화하여 분할하고 전송한다.

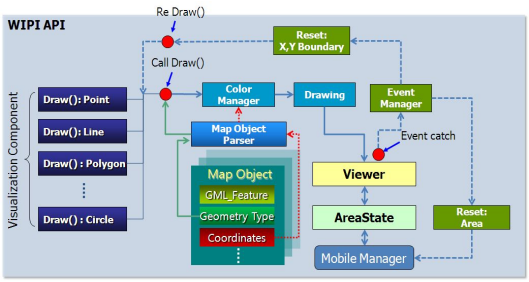
(그림 9) 적응적 지도 분할

3.2 모바일 디바이스에서의 지도 가시화

GML 지도 서버는 모바일 디바이스로부터 지도 요청이 있을 경우 액티브 영역을 포함하는 분할 영역을 전송한다. 모바일 디바이스는 GPS로부터 수신된 WGS84 좌표를 TM 좌표로 변환 후 서버에 지도를 요청하며 서버로부터 전송된 지도에 대해서는 다시 한 번 분할하여 가시화 한다. 또한, 모바일로 전송된 지도 데이터는 ID를 부여하여 관리한다. 본 논문에서는 구현 환경과 테스트를 위해 실제 모바일 환경에서의 UDP 프로토콜 대신 TCP/IP를 사용한다. 좌표변환 모듈은 실시간 시뮬레이터가 가능하도록 모바일 디바이스에 적용한다. 모바일 디바이스는 GPS 좌표를 수신하여 서버에 지도를 요청한다. 또한 서버로부터 수신된 지도는 모바일 디바이스의 디스플레이 영역을 기준으로 재분할하여 가시화 한다.

3.2.1 모바일 가시화 컴포넌트

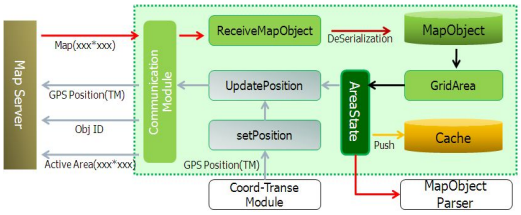
모바일 가시화 컴포넌트는 WIPI 플랫폼을 기반으로 한다. Map Object 파서에 의해 서버로부터 전송받은 Map Object를 분석하고, GML Feature의 객체들을 Viewer에 가시화한다. Map Object 파서에 의해 Map Object의 GML_Feature 내용들을 추출해 오면 Geometry Type에 따라 Visualization Component에 접근하여 해당 객체의 Type에 맞는 Draw() 메소드를 호출하고 다시 GML_Feature의 좌표를 가져와 도로와 건물간의 식별성을 위한 가시화 색깔을 설정하고 Drawing 과정을 수행하여 최종적으로 Viewer에 가시화한다. 모바일 디바이스의 이벤트에 대해서는 Event Manager에 의해 이벤트를 추출하고 해당 이벤트에 따라 X,Y 경계값을 재설정하여 재 가시화를 하면, 재설정된 경계값에 따라 초기 가시화 루틴을 재 수행한다. 또한 확대/축소 이벤트에 대해서는 모바일 디바이스의 4개의 가시영역이 변화하는데, 이에 따라 새로운 4개의 가시영역을 재설정하고, 이를 모바일 관리자(Mobile Manager)에 전달하여 서버의 지도 분할 영역에 대해서도 재설정한다. 그림 10은 모바일 디바이스의 지도 가시화 과정이다.



(그림 10) 모바일 디바이스의 지도 가시화 과정

3.2.2 모바일 관리자

그림 2의 모바일 디바이스 컴포넌트에 모듈로 존재하는 모바일 관리자는 서버에 지도 요청 및 수신을 담당한다. 모바일 관리자는 전송받은 지도를 4개의 가시영역별로 분할하며, 스크린 영역을 벗어날 경우 서버에 지도 재전송을 요청한다. 모바일 관리자의 작업 수행과정은 지도 서버로부터 지도를 전송받으면 바이트 형식으로 되어있는 MapObject를 ReceiveMapObject에서 복원하여 재저장한다. GridArea는 현재 화면이 Active 영역인지 스크린 영역인지를 판단하는 모듈로서, GridArea는 현재 보유중인 MapObject의 영역별로 분할하기 위해 AreaState로 보내고 AreaState는 맵의 현재 영역을 구분지어 각각의 수행 루틴으로 전달한다. 스크린 영역을 벗어나 액티브 영역에 진입했을 때에는 Update Position을 호출하여 현재 좌표를 GPS로부터 수신하여 모바일 디바이스에 지도 재전송 요청을 수행한다. 그림 11은 모바일 관리자의 수행 과정이다.



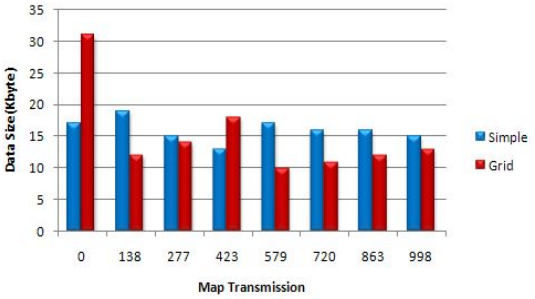
(그림 11) 모바일 관리자의 수행 과정

3.3 성능평가

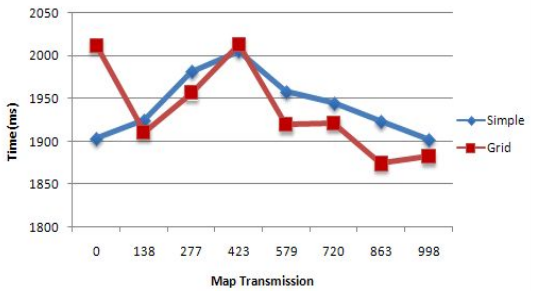
동일한 이동경로에서 1Km의 범위를 벗어나는 데에 대해 본 논문에서 제안한 기법들의 성능을 평가한다. 먼저 동적 지도 분할 및 전송 기법의 성능을 분석하고, 적응적 지도 분할 기법의 성능을 평가한다.

3.3.1 동적 지도 분할 및 전송 기법에 대한 성능 평가

그림 12는 동적 지도 분할 및 전송 기법(Grid로 표현됨)과 단순 지도 전송 기법을 비교한 그래프이다. 단순 지도 전송 기법은 지도를 분할 관리하지 않고 모바일 디바이스의 요청시 매번 일정한 영역을 추출하여 전송하는 기법으로써, 동적 지도 분할 및 전송 기법에 비해 초기에 보내는 데이터 크기는 작으나 그 후 이동시에는 데이터 크기가 크다. 따라서 그림 13과 같이 동적 지도 분할 및 전송 기법은 데이터를 전송하고 처리하는데 걸리는 시간이 감소한다.

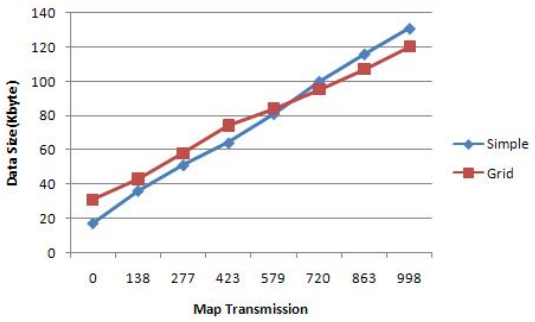


(그림 12) 지도 재전송에 따르는 데이터 크기



(그림 13) 지도 재전송에 따르는 데이터 전송 및 처리 시간

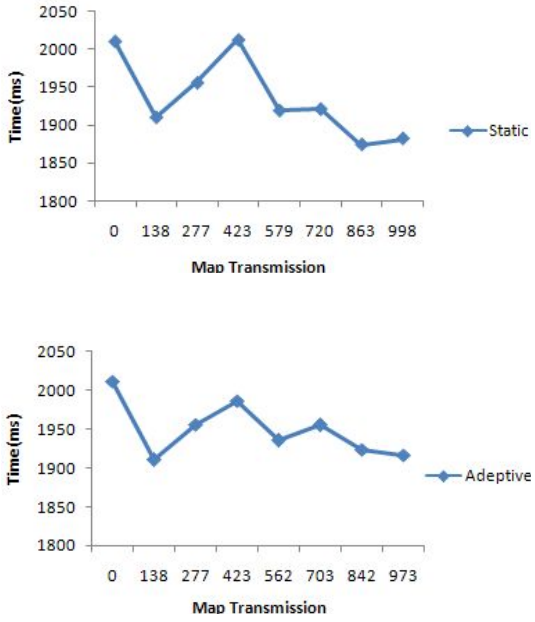
모바일 디바이스에 누적되는 데이터 크기는 6번째 전송이 이루어지는 720m 지점에서 그림 14와 같이 단순 지도 전송 기법이 동적 지도 분할 및 전송 기법에 비해 증가하는 것으로 결과를 보인다.



(그림 14) 모바일 디바이스에 누적되는 데이터 크기

3.3.2 적응적 지도 분할 기법

그림 15는 고정된 지도 분할 영역을 있는 그대로 전송하는 고정 지도 분할 기법과 밀집 지역 등에 대해서는 적응적으로 맵을 재분할하여 전송하는 적응적 지도 분할 기법과의 비교이다. 고정 지도 분할 기법은 전송 시간이 큰 폭으로 변화하고 있으나, 적응적 지도 분할 기법에서는 대체적으로 1900ms와 2000ms사이에서 일정한 전송 시간을 보이고 있다.



(그림 15) 적응적 지도 전송 기법 성능 평가

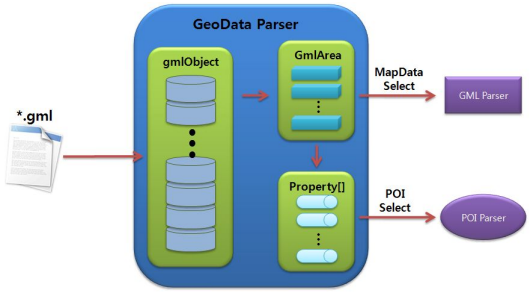
4. 태깅 기술을 활용한 GML 기반 POI 정보 관리 도구

그림 2에서 GML 지도 서버의 POIM는 기존의 POI 단순 분류 체계를 계층적 분류 체계화 시키고 태깅 기술을 활용하여 POI 관리를 자동으로 하는 컴포넌트에 속한다. POIM은 사용자가 POI에 대한 정보 추가, 수정, 삭제 등의 기능을 수행하면 자동으로 XML 기반 코드를 생성하여 GML과 결합하고 다시 GML 파싱을 걸쳐 지도 위에 가시화한다. POI 가시화를 위해 중간계층의 내부 데이터로 분류하는 파서 영역과 파서를 통해 분류된 지도 데이터와 POI 데이터를 기반으로 가시화를 담당하는 가시화 영역으로 분류한다.

4.1 GeoData 파서

그림 2에서 보는 바와 같이 GeoData 파서는 GML 지도와 POI 속성을 분류한다. 파싱된 GML 지도와 POI 객체는 다시 GML 파서와 POI 파서에 입력 데이터로 들어가게 된다.

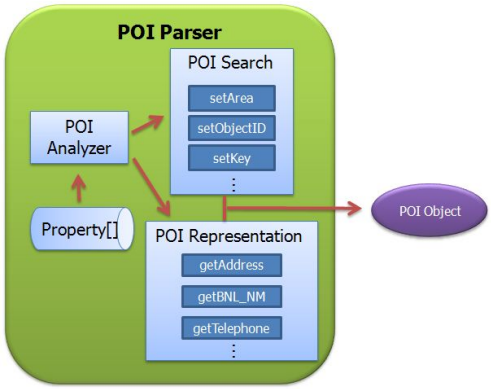
GeoData 파서는 File Importer에서 불러온 FeatureMember를 분석하여 지도를 표현하는 요소와 POI를 표현하는 요소로 분류하여 해당하는 파서에 전달하는 역할을 한다. POI Select는 POI 표현을 위한 주소, 상세주소, 전화번호, 홈페이지, 특징 엘리먼트들과 POI 검색을 위한 Key_cord, S_Cord, O_X 엘리먼트를 추출하여 POI 핸들러로 보낸다. 그림 16은 GeoData 파서 모듈의 세부 구조이다



(그림 16) GeoData 파서 구조

4.2 POI 파서

POI 파서는 GeoData 파서로부터 전달된 POI 엘리먼트들을 POI Analyzer에 의해 POI Search와 POI Representation로 분류된 POI Object를 생성한다. POI Search는 3-Layer 구조를 따르는 계층적 검색을 위한 식별키가 전달되며, POI Representation은 화면에 가시화될 실제 POI 정보 표현 부분이 전달된다. POI Search는 Area, ObjectID, Key 등을 전달받아 사용자가 POI 검색을 요청할 때, setArea, setObjectID, setKey 등의 메소드를 통해 설정된다. POI Representation는 실제 화면에 가시화될 정보들인 주소, 이름, 전화번호, 홈페이지, 특징 등을 전달받으며, getAddress, getBLG_NM, getTelephone, getHomepage, getEx 메소드에 의해 원시 코드인 엘리먼트 형식의 코드를 얻는다. POI Object Maker는 POI Search에 의해 키가 설정되고, POI Representation에 의해 가시화 정보를 트리 구조 형태로 POI Object Extractor를 통해 POI Object를 생성한다. 그림 17은 POI 파서의 세부 구조이다.



(그림 17) POI 파서 구조

4.3 POI 가시화 컴포넌트

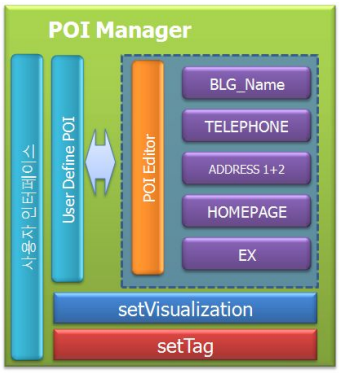
POI 가시화 컴포넌트는 그림 2에서 POIM를 의미하며, POI 파서에서 생성된 POI Object를 이용하여 사용자가 POI 검색을 요청할 경우, Viewer에 팝업창 형태로 가시화해주는 역할을 한다.

Viewer에서 POI 검색을 요청할 경우, 공통 모듈인 Event Manager는 POI 관련 이벤트만을 분류해 POI Parser로 이벤트를 알린다. POI 관련 이벤트는 POI 정보 보기와 계층적 POI 검색을 제공한다. POI 관련 이벤트는 Search Eng.의 Event Analyzer에서 검색 요청된 건물의 ObjectID, Area 등을 메시지화 시켜주는 Search Msg Maker에 의해 메시지를 생성한다. POI 관리자는 검색 메시지가 넘어 왔을 경우 해당 메시지 값을 POI 파서로 보낸다. POI 파서는 POI Object에 접근해 메시지 값을 검색하여 요청된 식별키를 반환한다. POI Getter는 POI 관리자로 전달된 키와 Search Eng.에서 넘어온 검색 메시지를 매칭한 후, POI Object에서 가시화할 POI 정보를 가져온다. 이렇게 얻어진 POI 정보는 POI Analyzer에서 최종 POI 분석을 하고 Visual Config에서 가시화할 부분만을 추출된다. POI 정보는 팝업 형태로 가시화되는데, 이는 PopUp Maker를 통해 팝업 창이 생성된다.

4.4 POI 관리자

POI 관리자는 POI 가시화 컴포넌트 내부에 존재하며, 기존의 고정된 POI 항목에서 내용만을 편집했던 서비스에서 내용뿐만 아니라 항목까지도 수정하여 사용자로 하여금 서비스에 참여하여 서비스 제공에 직접적으로 관여하도록 지원한다. POI 관리자는 사용자로부터 POI 검색이 요청되면 파서로부터 POI 정보들을 불러와 팝업형식의 사용자 인터페이스로 가시화 한다. 또한 사용자 인터페이스의 POI 정보의 편집에 따라 POI Object의 원본 POI 정보들을 수정하는 역할을 한다. 본 논문의 기반으로 사용되는 GML 파일의 POI는 객체의 이름을 표현하는 BLG_Name, 전화번호를 의미하는 TELEPHONE, 주소인 ADDRESS1+2, 객체의 홈페이지인 HOMEPAGE, 기타 해당 개체의

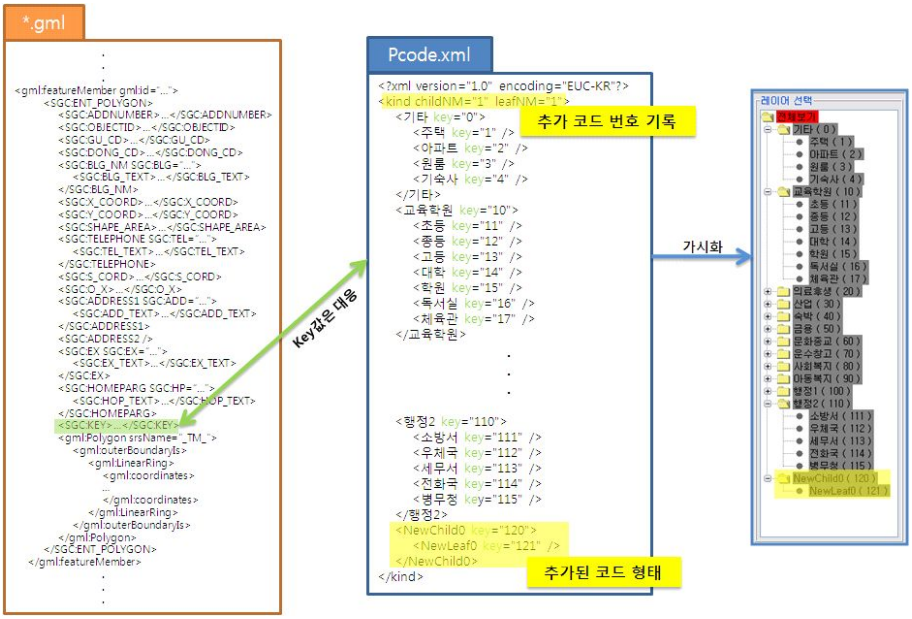
특징을 나타내는 EX로 총 5개의 항목으로 이루어져 있다. 초기 POI 검색은 기존 5개의 POI 항목을 사용자 인터페이스에 기본적으로 표현한다. POI 편집은 기존 5개 항목을 포함한 항목의 추가/삭제와 각 항목에 따른 내용 수정을 제공한다. User Define POI는 사용자 인터페이스에서 넘어온 수정 값들에 대한 데이터를 저장하고, POI Editor에 의해서 각 항목 엘리먼트의 포인터에 접근하여 각 수정 항목들에 매칭하여 User Define POI의 정보로 수정한다. 일정 지도 안의 모든 객체들의 POI표현에 대해서는 사용자가 원하는 POI만 표현되도록 setVisualization에서 설정하도록 지원한다. 그림 18은 POI 관리자의 세부 구조이다. 태깅에 대해서는 다음 절에서 자세히 기술하고 개인화 태그정보의 PCode 태그정보들을 바탕으로 해당 POI의 분류를 설정한다. 태깅에 대해서는 GML 파일 내부의 POI를 구성하는 엘리먼트 중 <key> 엘리먼트에 설정되며 이를 통해서 태그 정보로 검색을 제공한다.



(그림 18) POI Manager 세부 구조

4.5 개인화 정보 태깅

개인화 태그정보는 GML의 POI 정보들에 태그를 달기 위한 기반으로 POI의 분류체계를 사용자가 직접 정의한 후 이를 XML 형식으로 분류정보를 패킹하여 PCode.xml이라는 파일로 저장한다. 저장된 분류정보와 GML의 <key> 태그는 서로



(그림 19) 개인화 정보 태깅 과정

매칭이 되면서 POI 객체에 분류체계를 적용한다. 그림 19는 개인화 정보 태깅 과정이다.

GML은 지도표현을 위한 마크업 언어로써 전체적인 구조는 GML 스키마에 의존적이다. 이런 특정 스키마에 국한된 GML파일은 각 객체의 POI 속성을 담당하는 태그들에 사용자가 임의적으로 수정하거나, 각각의 객체마다 POI 태그들을 다른 이름으로 작성한다는 건 문법상의 오류를 범하게 된다. 이에 본 논문에서는 GML파일의 기본 골격이라 할 수 있는 GML 스키마를 해하지 않으면서 각 객체의 POI마다 분류태그를 달기 위해 POI 정보에 <key> 태그를 추가하였다. <key> 태그의 값은 개인화 태그정보에 의해 결정된다.

개인화 태그정보는 사용자가 정의한 태그정보를 XML 파일로 따로 분류하고 GML 파일 내의 <key> 태그에 개인화 태그정보의 key값으로 설정하여 POI 객체에 분류체계를 적용한다. 사용자가 새로운 분류를 추가하면 NewChild라는 기본이름을 갖는 분류와 그에 따르는 기본 key값이 생성되고 이는 개인화 태그정보에 <NewChild key=""> 라는 엘리먼트로 추가 삽입된다. 기본 이름으로 정의된 NewChild는 Edit를 통해 사용자가 원하는 이름으로 변경이 가능하며 변경된 내용이 적용된다. 개인화 태그 정보는 최초 사용자에 의해 분류가 되면 이는 단순히 태깅 정보에 불과하지만 POIM을 사용하는 사용자들에게 배포함으로써 해당 개인화 태그정보를 공유하는 사용자들 간에 집단이 형성되며 이러한 집단을 통해 수시로 업데이트 되는 새로운 버전의 개인화 태그정보를 사용자들의 POIM에 적용 가능하다. 이는 웹 자원이 아닌 GIS의 POI 데이터에 대해서도 태깅 기술을 적용하고 태그정보들을 배포 및 공유를 통해 집단을 구성하여 폭소노미를 실현한다.

5. GML 지도 서비스 구현 적용 사례

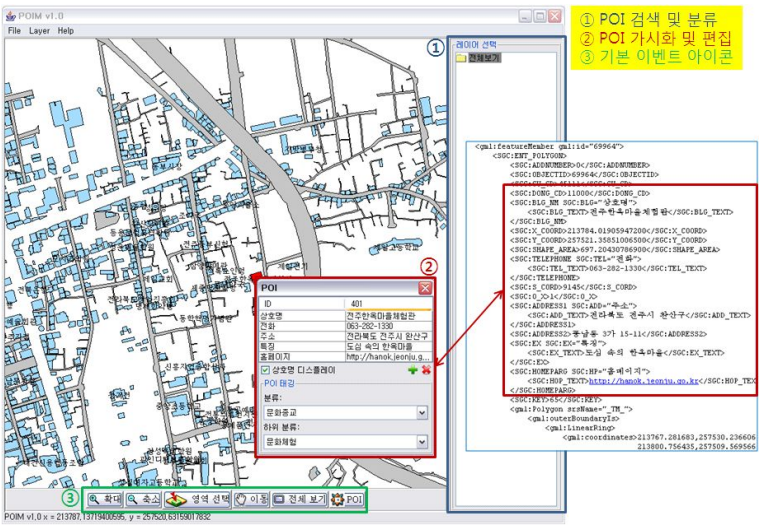
그림 20은 GML 지도 서버와 모바일 디바이스로 지도를 가시화한 화면이다. 먼저 지도 서버는 도로와 건물은 색상으로 구분한다. 가시화 창을 중심으로 위쪽엔 메뉴바(①)가 있고, 지도 가시화 창을 기반으로 모니터링 아이콘(②)이 가시화되며, 하단에는 이벤트 아이콘(③)이 존재한다. 그리고 모바일 디바이스는 GML 지도 서버로부터 지도를 실시간 공급받고 이를 Aroma WIPI Emulator에 가시화하는 화면(④)이다. 이 경로는 “모 치과의원”에서부터 “전주 동부신협” 옆 골목까지 이

(그림 20) GML 지도 가시화 및 모바일 디바이스 가시화

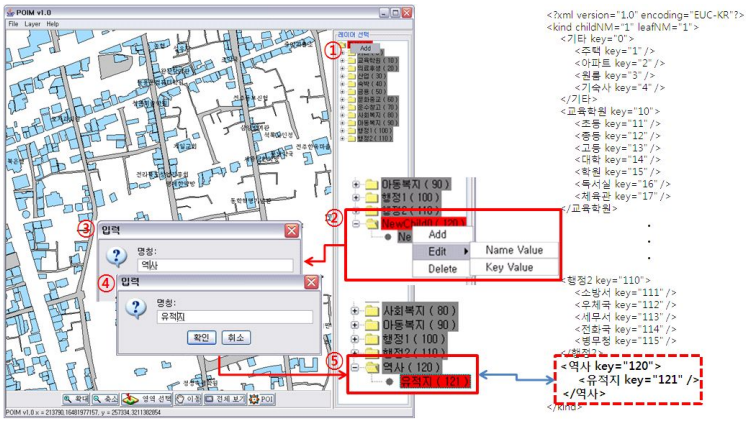
동한 화면이다.

POIM은 지도 가시화창을 중심으로 오른쪽엔 POI 검색과 사용자정의 분류체계를 적용할 수 있는 레이어 선택창(①)과 하단에는 지도의 기본 이벤트 아이콘(③)들이 있으며 POI 검색이나 객체 선택을 통해 POI를 가시화하는 POI Viewer(②)로 그림 21과 같다. 레이어 선택창은 POI의 계층구조를 사용자가 직접 정의하고, 정의된 계층구조를 바탕으로 POI 검색을 한다. 기본 이벤트 아이콘은 지도의 확대/축소/영역선택/이동/전체보기/POI 보기로 이루어져 있다. POI Viewer는 POI를 가시화 할 뿐만 아니라 POI의 항목 등에 대한 편집이 가능하며, 상호명 디스플레이를 통해 상호명을 지도에 표현 할 것인지에 대한 유/무 체크가 가능하다. 또한 레이어 선택창의 사용자 정의 계층구조를 바탕으로 POI 태깅을 통해 계층구조를 적용한다.

POI 개인화 정보 태깅에 대한 사례는 그림 22



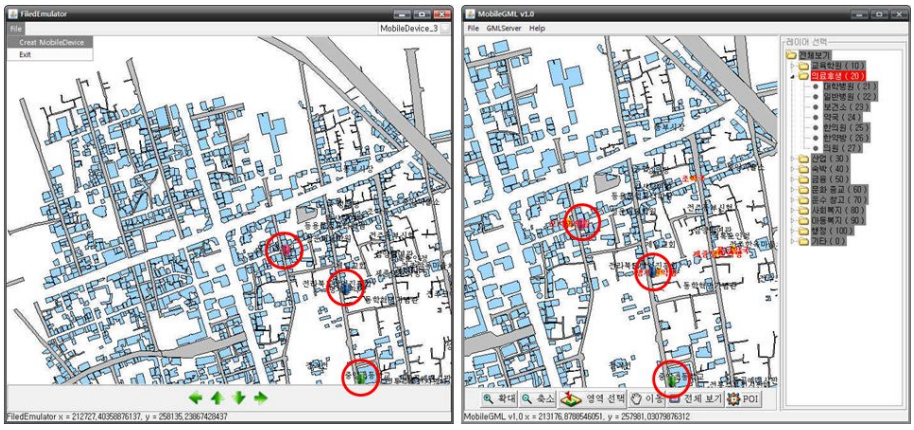
(그림 21) POIM 가시화 실례



(그림 22) 개인화 정보 태깅 과정 사례

와 같고 POI 편집 및 태깅은 아직 등록되지 않은 객체에 대해 POI를 가시화하면 빈 테이블만 보이며, 빈 테이블에 항목과 항목에 따르는 데이터를 삽입한다. 항목은 기본으로 5개 리스트가 제공되나 사용자가 원하는 만큼의 항목 리스트를 추가 가능하여 객체마다 POI 항목을 다르게 하여 좀 더 완전한 사용자 맞춤형 POI 서비스를 지원한다.

필드 에뮬레이터를 통하여 모바일 디바이스를 생성하여 GML 지도 서버와 동기화는 것은 그림 23과 같다. 왼쪽 부분이 필드 에뮬레이터이며, 간단히 마우스를 통하여 모바일 디바이스가 생성되며, 하단 화살표에 의해 위치를 이동하게 되면 동시에 오른쪽 GML 지도 서버와 동기화 되어 위치 추적이 모니터링된다. 즉, (b)에서 필드 에뮬레이



(a) 모바일 디바이스 생성 및 동기화

(b) 모바일 디바이스 이동 및 위치 추적 모니터링

(그림 23) 필드 에뮬레이터와 GML 지도 서버와의 동기화

터의 왼쪽화살표를 클릭하여 모바일 디바이스(빨간색아이콘)가 왼쪽으로 GML 지도 서버의 모바일 디바이스와 동시에 이동하는 상태를 보여준다.

6. 결론

본 논문에서 GML 지도 서버에서는 GML 지도를 가시화하고 처리하는 기능, WIPI 환경에서 필드 에뮬레이터를 활용하여 모바일 디바이스 위치 추적 모니터링 및 가시화 기능, POI 정보처리를 위한 POIM 관리 도구를 개발하였다. WIPI 모바일 디바이스는 GML 지도 서버로부터 지도를 실시간으로 전송받아 가시화하고 동적 지도 분할 및 캐싱 기법을 구현하였다. 필드 에뮬레이터는 GML 기반으로 가시화된 지도에서 모바일 디바이스들을 생성하여 GPS 좌표 중심으로 이동시켰다. 필드 에뮬레이터에서 생성된 모바일 디바이스의 좌표 정보는 GML 지도 서버와 동기화되어 그 이동경로에 대한 추적 모니터링이 됨을 증명하였다. POIM은 비주얼한 인터페이스를 통하여 손쉽게 POI 정보 구조를 계층화하여 생성, 수정, 저장하도록 하였으며, 개인화 정보 태깅 기술을 적용하여 POI 정보 관리 및 검색이 더욱 효율적으로 이루어졌다. 최종적으로 GML은 태깅된 정보들을 포함하는 구조로 향후 웹과 연동하는 API를 구현하게 되면 시맨틱 웹 기술 적용을 위한 토대를 마련하였다.

향후 연구 과제로는 GML 데이터의 좀 더 효율적 전송 및 관리를 위해 ZIP 알고리즘 등 압축 알고리즘을 적용하여 좀 더 경량화 시키고, preCache 기법을 적용하여 방향성을 고려한 예측 지도 전송 알고리즘을 연구해야 한다. EH한, 모바일 디바이스로 GML 지도 정보만이 아니라 분할된 지도정보 내 POI 정보들이 서로 동기화되어 동시에 GML 지도 서버에서 모바일 디바이스로 전송하는 방법을 연구하여야 한다.

참 고 문 헌

[1] OpenGIS Consortium, Inc., Geography Markup Language[GML] Implementation Specification, http://www.opengeospatial.org/docs/02-023r4.pdf

[2] Shashi Shekhar, Ranga Raju Vatsavai, Namita Sahay, Thomas E. Burk Stephen Lime, "WMS and GML based interoperable web mapping system", GIS: Geographic Information Systems, pp.106~111, 2001.

[3] Zhimao Guo, Shuigeng Zhou, Zhengchuan Xu, Aoying Zhou, "G2ST: a novel method to transform GML to SVG", Proceedings of the 11th ACM international symposium on Advances in geographic information systems, pp.161-168, 2003.

[4] S. Takino, "GIS on the fly to realize wireless GIS network by Java mobile phone," Web Information Systems Engineering, Proceedings of the Second International Conference on, Vol. 2, pp.76-81, 2002.

[5] Owen Kaser and Daniel Lemire, "Tag-Cloud Drawing: Algorithms for Cloud Visualization", Tagging and Metadata for Social Information Organization, WWW 2007, 2007.

[6] 김대식, 김형진, 손봉수, 유완, "생활지리정보 검색 및 안내를 위한 POI 구축 및 활용", 한국콘텐츠학회 추계종합학술대회 논문집Vol. 1 No. 2, pp.423-430, 2003.

[7] eSpatial Inc. iSMART Explorer4.4, http://www.espatial.com/productsismartexplorerversion44.htm

[8] Saft Software Inc. FME Universal Viewer 2003, http://www.saft.com/producsts/fme/index.php

[9] TatukGIS Inc. TatukGIS Viewer 1.4, http://www.tatukgis.com/products/Viewer/viewer.aspx

[10] del.icio.us, http://del.icio.us/

[11] Flickr, http://www.flickr.com/

[12] Technorati, http://technorati.com/
[13] 삼성전자 통신연구소, "LBS 측위기술 개발", LBS 산업협의회 창립 총회 및 기념 세미나, 2003.
[14] OVUM, "Mobile Location Services, Market Strategies", 2000.

◐ 저 자 소 개 ◑

박 용 진(Yong-Jin Park)
2006년 원광대학교 전기전자 및 정보공학부 졸업(학사)
2008년 원광대학교 대학원 컴퓨터공학과 졸업(석사)
관심분야 : LBS, GIS
E-mail : yjpark1@wku.ac.kr

송 은 하(Eun-Ha Song)
1997년 원광대학교 통계학과 졸업(학사)
2000년 원광대학교 대학원 컴퓨터공학과 졸업(석사)
2006년 원광대학교 대학원 컴퓨터공학과 졸업(박사)
2007~현재 원광대학교 전기전자 및 정보공학부 전임강사
관심분야 : 그리드컴퓨팅, 시맨틱그리드, 분산병렬시스템, LBS
E-mail : ehsong@wku.ac.kr

정 영 식(Young-Sik Jeong)
1987년 고려대학교 수학과 졸업(학사)
1989년 고려대학교 대학원 전산학과 졸업(석사)
1993년 고려대학교 대학원 전산학과 졸업(박사)
2004년 웨인 주립대학교 객원교수
1993~현재 원광대학교 전기전자 및 정보공학부 교수
관심분야 : 그리드컴퓨팅, 시맨틱그리드, 분산병렬시스템, LBS
E-mail : ysjeong@wku.ac.kr