Applied GIS
a free, international, refereed e-journal
(ISSN: 1832-5505)

URL:
http://www.appliedgis.net

Volume 4, Number 4
October, 2008

CONTENTS:
All papers published during 2008 are part of *Volume* 4.
Each paper constitutes one *Number.*

Hence this paper should be cited as:

Stefanakis, E. & Prastacos, P. (2008) – Development of an open source-based spatial data infrastructure, *Applied GIS*, 4(4): 1-26

Development of an open source-based spatial data infrastructure

Emmanuel Stefanakis
Department of Geography,
Harokopio University of Athens,
Athens, Greece
estef@hua.gr

Poulicos Prastacos
Foundation for Research and Technology (FORTH),
InfoCharta Ltd, STEP-C,
Heraklion, Greece
poulicos@iacm.forth.gr

Abstract: A pilot regional *Spatial Data Infrastructure (SDI)* has recently been developed for the Heraklion Prefecture in Crete, Greece, using *Geographic Free and Open Source Software (GeoFOSS).* This SDI is compatible with the geospatial standards and specifications introduced by the Open Geospatial Consortium (OGC) and it distributes the geospatial content on the web through widely accepted services (e.g., WMS, WFS, WCS and CSW). This paper presents the architecture, the components and the functionality of the Heraklion SDI. Specifically, it focuses on the map server, along with the services that provide accessibility to the data repositories, the spatial database server, the metadata and data catalog, the visualization tools and the web-client interface.

Keywords: Web services, geovisualization, metadata, spatial data infrastructures.

1.0 Introduction

Special attention has recently been given on the development of effective *spatial data infrastructures (SDIs)* at regional, national or international level (Williamson et al. 2004). SDIs are frameworks of policies, institutional arrangements, data, services, technologies, and people with a common role - promoting the accessibility and usability of geospatial content (data and services). The construction of an SDI presumes that the participating organizations have agreed on the adoption of common vocabularies, practices, standards, technical specifications and operational components (Nebert, 2004). An SDI is not, therefore, a simple data set or repository. Instead, an SDI hosts:

a) geographic content (data and services);
b) sufficient description of this content (metadata);
c) effective methods to discover and evaluate this content (data catalogs);
d) tools to visualize the data (e.g., web mapping); and
e) services and software tools to support specific application domains.

An SDI architecture is usually conceptualized as a *three-tier architecture* (Evans, 2003). At the *top layer* (the client) reside the users and applications; at the *bottom layer* (the server) reside the geospatial data; and at the *middle layer* (the middleware) reside all services that assist the accessibility to the data repositories (Figure 1a). Obviously, the middle layer is the one that facilitates the access to the online geospatial data and their eventual use.

Currently, the SDI paradigm focuses on the exchange of geospatial data. That is, the middle layer supports the discovery and retrieval of data available at the bottom layer, and this is consistent to the term spatial *data* infrastructure. On the other hand, it is widely recognized (e.g., Bernard et al., 2005; Stefanakis and Prastacos, 2007), that future SDI research must focus more on the users' needs and the design of appropriate geo-information services and service architectures to satisfy these needs. In other words, the development of service-driven infrastructures; or spatial *information* infrastructures will enhance functionality. This may be accomplished by enriching the SDI layers (Nebert, 2004) (Figure 1b).

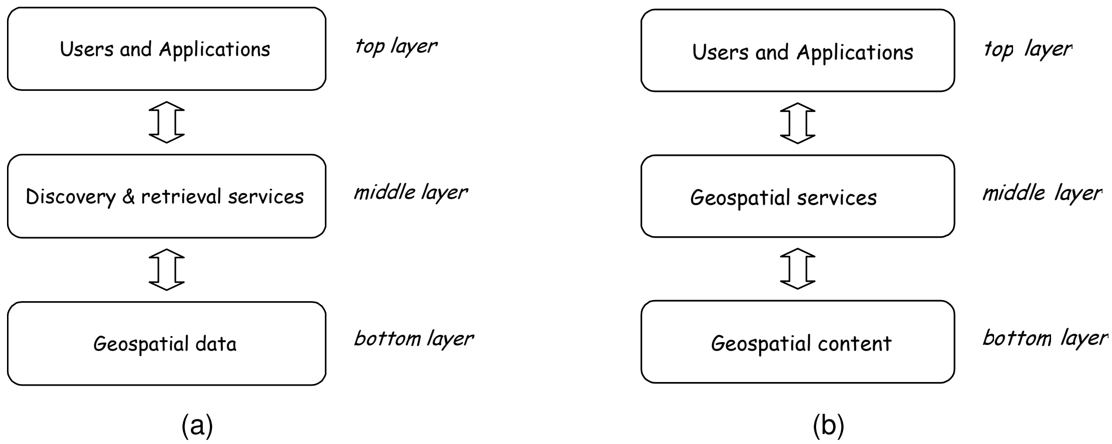


Figure 1 - The three-tier architecture of current (a) and future (b) SDIs.

The *bottom layer* will accommodate analytical software tools as well as data. Moreover, at the *middle layer*, existing discovery and retrieval services will be extended to apply to both the data and services available at the bottom layer. Note that appropriate mechanisms must ensure both that sufficient description (metadata) of the geospatial content (both data and services) is available and the discovery of distributed collections of geospatial content, through their metadata descriptions, is supported - through the catalogue gateway. Finally, the middle layer will incorporate software tools and services that may be used by the top layer (users and applications), and so the whole SDI functionality will be enhanced. As for the *top layer*, it must provide metaphors to discover and access the geospatial content; it must consist of well designed and user friendly interfaces.

The development of an SDI should be supported by a set of sophisticated *software systems and tools* and it must conform to a series of *standards and specifications* in order to ensure interoperability between repositories and geospatial content (Stefanakis and Prastacos, 2007). With respect the software systems, commercial GIS packages are already available to support the development of high quality SDIs and the diffusion of geospatial content. Yet on the other hand, Open Source Software (OSS) has been maturing, over the last several years, into robust and well-supported tools. *Open Source Geospatial software* (e.g., GIS, Map Servers, etc.) is no exception to this trend, and it is now able to accommodate the needs of geoscientists and professionals (OSGeo). As related to the *geospatial standards and specifications*, the Open Geospatial Consortium (OGC), in parallel with the World Wide Web Consortium (W3C) and the International Organization for Standardization (ISO), have already developed rich standards and specifications to support interoperability between repositories that contain geospatial content.

At the Foundation for Research and Technology (FORTH), we are currently running a project entitled "Development of an Integrated GIS for the Management of the Coastal Zone at Heraklion – Crete", and it is funded by the General Secretariat of Research and Technology in Greece and the E.U. (PEP Crete, 2006-8). The main objective of this project is the development of an *SDI for the northern coast of Heraklion prefecture*, but the research team has expanded its scope to cover the whole prefecture. The SDI will host the geographic content collected for the prefecture, accompanied with rich metadata. SDI users will be provided with effective methods to discover and visualize this content, and sophisticated software and tools to support application domains related to regional management. Because of the contemporary trend towards adoption of free and open source software, it was decided to generate, under this project, two parallel SDIs for the Heraklion Prefecture:

a) one SDI using commercial software systems, and
b) an alternative SDI using only Geographic Free and Open Source Software (GeoFOSS).

This paper considers the latter SDI, i.e., the regional SDI based on *GeoFOSS.* Specifically, it presents the overall architecture, the software packages adopted for implementation, the servers and the SDI's functionality for the Heraklion Prefecture. Comparing the open source SDI against the commercial SDI, with respect to their functionality and eventual use, is one of our future plans (see Section 8).

The discussion is organized as follows. Section 2 presents the *Heraklion SDI* architecture, the geospatial content, the software systems and the tools used. Section 3 briefly presents the development phases of the Heraklion SDI. Section 4 describes access to content through map servers and services, i.e. the WMS, WFS, WCS and KML servers. Section 5 focuses on the user interfaces, i.e. the client applications for visualizing SDI geospatial content through the web. Section 6 discusses the spatial database server and it highlights the Heraklion SDI's querying and analysis capabilities. Section 7 describes the geospatial content metadata items and the SDI catalog server, which supports the diffusion of the SDI content (data and applications) onto the web using widely accepted standards and protocols. Finally, Section 8 concludes the discussion by summarizing the contribution of this paper as well as plans for future development.

The Heraklion SDI content is currently available at the following URL:

http://heraklion-sdi.dynalias.net/coastatlas/index-en.html

In the discussion that follows, all URL requests are pointing to the *http://localhost*. For instance, in Figure 3 the introductory page of the SDI is accessed at:

http://localhost/coastatlas/index-en.html

Note also that the reader may gain access to actual page, as well as to all the public datasets and applications presented in this paper, by replacing the *http://localhost* with the SDI server DNS, i.e.:

http://heraklion-sdi.dynalias.net

2.0 The SDI architecture

2.1 Geospatial Content

The SDI of the Heraklion Prefecture can accommodate a series of *geospatial data layers* based on user and application needs for regional planning. The existing public layers of the SDI are listed in Table 1. In what follows we limit our discussion to the layers in Table 1, although numerous other layers could be accommodated within the SDI in the future. Also, the SDI imports and combines, with the layers above, the *geospatial mashups* provided by other servers on the web. Currently, these include Google Maps, the NASA JPL Portal, Nasa's WMS and the Integrated CEOS European Data Server.

2.2 Architecture

The architecture of the *Heraklion SDI* is shown in Figure 2. It is compatible with the proposal for GeoFOSS SDI suggested by Ticheler (2007) and it follows the conceptual *three-tier architecture* for SDIs as presented in Figure 1 above (Evans, 2003). At the *bottom layer*, reside the SDI repositories and the spatial database server. As shown in Table 1, the geospatial layers are available in shapefile and geotiff formats. The content of these layers has been imported into Postgresql/PostGIS in order to achieve more efficient data management and analysis (see Section 6 below).

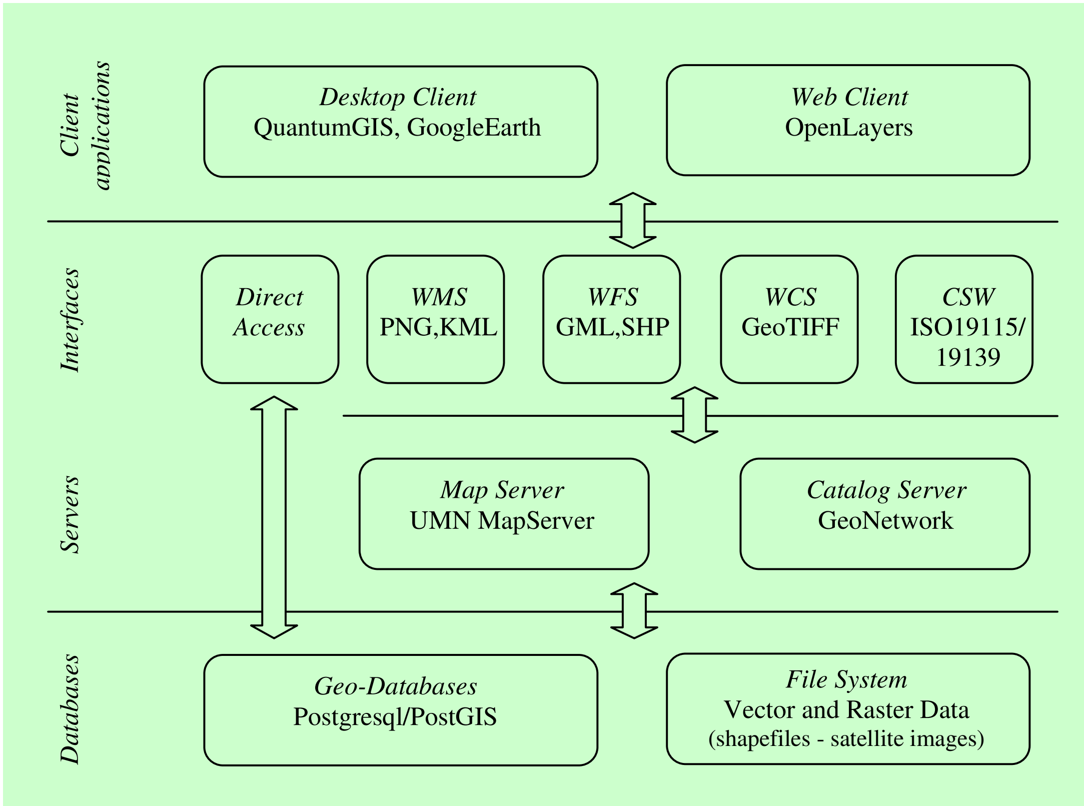


Figure 2 - The architecture of the Heraklion SDI using GeoFOSS.

Layer Content/Description	*Format*	*Name*
Prefecture (outline)	Shapefile	*nomos_irakliou*
Municipalities (outlines)	Shapefile	*dhmoi_irakliou*
Municipalities Subdivisions (outlines)	Shapefile	*dhm_diamer_irakliou*
Urban Areas (outlines)	Shapefile	*bua_irakliou*
Towns and Villages (points)	Shapefile	*oikismoi_irakliou*
Road Network (lines)	Shapefile	*odiko_irakliou*
Heraklion City buildings (polygons)	Shapefile	*build_egsa*
Heraklion city Airport and Seaport (points)	Shapefile	*poi_irakliou*
Lakes (polygons)	Shapefile	*limnes_irakliou*
Geology cover (polygons)	Shapefile	*geo_N_Herakleio*
Archaeological spots (points)	Shapefile	*archaiologia_N_Herakleio*
Digital Elevation Model (raster)	GeoTiff	*dem*
Land Cover (raster)	GeoTiff	*lc*
Orthophoto Map (raster)	GeoTiff	*hr_ortho*

Table 1 - The *Public* Data Layers of the Heraklion SDI.

In the *middle layer* (the middleware) reside all the services that assist the accessibility to the data repositories. The Heraklion SDI has two main servers, i.e., the *Map Server* (see Section 4 below) and the *Catalog Server* (see Section 7 below). These two servers disseminate (serve) the geospatial content onto the web, based on standard interfaces (e.g., WMS, WFS, WCS, CSW–ISO19115/139), to facilitate access and use of online geospatial data. The middle layer also provides a *Direct Access Service* to the geographic database for advanced querying and for analysis of the geospatial content (see Section 6 below).

At the *top layer* (the client) reside the users and applications. Access to the SDI geospatial content is possible through either a desktop or a web client. The desktop client can be any software package with geovisualization capabilities and, possibly, GIS functionality, such as a GIS software package (e.g., ESRI ArcGIS, Quantum GIS) or a Map Viewer (e.g., Google Earth). The web client is a web browser, such as MS Internet Explorer, Mozilla Firefox, or Google Chrome. Desktop and web clients, as well as the visualization capabilities available at the Heraklion SDI, are discussed in Section 5.

2.3 Software Systems and Tools

The regional SDI using GeoFOSS has adopted the following *software systems and tools* (OSGeo) to manage, discover, analyze and disseminate geospatial content:

- *Postgresql/PostGIS* (http://www.postgis.org) in the role of the Spatial Database Server.
- *UMN MapServer* (http://mapserver.gis.umn.edu) in the role of the Map Server.
- *GDAL/OGR* (http://www.gdal.org) in the role of the Geospatial Library.
- *QuantumGIS* (http://www.qgis.org) in the role of the Desktop Client.
- *GoogleEarth* (http://earth.google.com) in the role of the Desktop Client (for the KML files).
- *OpenLayers JavaScript Library* (http://openlayers.org) in the role of the Web Client (Interface).
- *GeoNetwork Opensource* (http://geonetwork-opensource.org) in the role of the Catalog Server.
- *Apache Web Server* (http://www.apache.org) in the role of the Web Server. As a web server, Apache is the software that establishes and ensures communication between the Heraklion Server and the World Wide Web.

The following Subsections briefly present most of the software packages that assist implementation of the Open Source-based SDI for the Heraklion Prefecture.

2.3.1 PostgreSQL/PostGIS

PostgreSQL is a powerful, enterprise-level, object-relational database server product, and it is the backbone of data repositories for numerous applications and web services. *PostGIS* is a product developed by Refractions Research, as a project in open source spatial database technology (GNU General Public License), which extends PostgreSQL (also free and open).

PostGIS supports a superset of geographic objects defined by the “Simple Features” proposed by the Open Geospatial Consortium (OGC). From version 0.9 onwards, PostGIS supports all of the objects and functions specified in the OGC “Simple Features for SQL” specification. PostGIS (current version 1.3.3) extends this standard by means of support for 3DZ, 3DM and 4D coordinates.

Hence PostGIS is more than a geographic, data-storage extension of PostgreSQL. It allows, through SQL statements, both management and analysis (e.g., geographic overlays, spatial proximity queries) of geographic data, like any GIS system, in a server-side database solution. Section 6 highlights the capabilities of PostgreSQL/PostGIS in querying and analyzing the geospatial content of the Heraklion SDI by outlining some representative examples.

2.3.2 UMN MapServer

UMN MapServer is an open source, development environment for building spatially-enabled web mapping applications and services. It was originally developed by the University of Minnesota (UMN) in cooperation with NASA and the Minnesota Department of Natural Resources (MNDNR). Presently, the software is maintained by a growing number of developers from around the world, and it is supported by a diverse group of organizations that fund enhancements and maintenance (OSGeo Web Mapping Project). UMN MapServer excels at rendering spatial data (maps, images, and vector data) for the web. Current versions, support common Open Geospatial Consortium (OGC) standards, including WMS, WFS and WCS, and it may quite efficiently play the role of the map server in a spatially-enabled web mapping application.

2.3.3 Geospatial Data Abstraction Library (GDAL/OGR)

Geospatial Data Abstraction Library (GDAL/OGR) is a cross platform, C++ translator library for raster and vector geospatial data formats. GDAL/OGR has an X/MIT style Open Source license and it was developed for, and by the Open Source software community (OSGeo Geospatial Libraries Project). The current version (1.5.1) comes with a variety of useful command-line utilities for data translation (i.e., conversion between various formats and/or transformation between various geographic reference systems) and for processing. GDAL supports over 80 raster formats (e.g., GeoTIFF, JPEG2000, etc), and OGR supports over 30 vector formats (e.g., ESRI Shapefile, PostGIS, GML, KML, Oracle Spatial, MapInfo, DGN, etc.). On fact, GDAL/OGR is probably the most widely used geospatial data access library. It constitutes the primary data access engine for many applications, including MapServer, and QGIS. It is also utilized by packages such as Google Earth, MapGuide and ArcGIS.

2.3.4 Quantum GIS

Quantum GIS (QGIS) is a user friendly Geographic Information System (GIS) that runs on GNU/Linux, Unix, Mac OSX, and Windows. It is has a GNU General Public License and again, was developed for, and by the Open Source software community (OSGeo Desktop Applications Project). It supports the creation and editing of a variety of vector and raster formats, such as ESRI shapefiles, PostgreSQL/PostGIS spatial data and GeoTiff images. Both customized plugins and GIS-enabled applications can also be generated using Python or C++ programming languages, and QGIS itself provides plugins to import delimited text data, or to download tracks, routes, and waypoints from GPS receivers (handling them in a layer such as a shapefile), or to visualize geographic data offered from OGC WMS and WFS servers.

2.3.5 OpenLayers JavaScript Library

OpenLayers is a pure JavaScript library for displaying map data. It is compatible with most modern web browsers, and independent on the server side. OpenLayers is Free Software, developed for, and by the Open Source software community (OSGeo Web Mapping Project) and it provides a JavaScript API (similar to the Google Maps or MSN Virtual Earth APIs) for building rich, web-based geographic applications. Additionally, OpenLayers implements industry-standard methods for geographic data access, such as the Open Geospatial Consortium's Web Mapping Service (WMS) and the Web Feature Service (WFS) protocols.

2.3.6 GeoNetwork Opensource

GeoNetwork opensource is a catalog application for managing spatially referenced resources. It provides powerful metadata editing and search functions, as well as an interactive web map viewer (provided by InterMap; another OSGeo Project). It is an OSGeo Web Mapping Project and implements both the Portal component and the Catalog database of a Spatial Data Infrastructure (SDI) if it is defined in the OGC Reference Architecture. GeoNetwork adopts International and Open Standards for services and protocols (ISO/TC211 and OGC) and it provides native support for ISO19115/ISO19139, FGDC and Dublin Core formatted metadata. Additionally, it allows a distributed search, which provides access to a huge volume of metadata which comes from different Clearinghouses, and it also provides a web-based, interactive map viewer, thereby allowing people to compose maps by taking layers from distributed servers on the internet.

3.0 Phases of development

The development of the Heraklion SDI using GeoFOSS went through four main phases and these will now be described in turn.

Phase 1 - Preparation of the geospatial layers

This phase focused on the preparation of the geospatial layers that had been collected for the Heraklion Prefecture. Specifically, appropriate transformations were carried out in order to insert those layers listed in Table 1 above, at the bottom layer of the SDI (see Figure 1 above). These include map projection transformations, the assignment of map symbols, the loading of geospatial data into the Geographic Database Management System and the mapping of geospatial data into KML format in order that they can be visualized on top of Google Maps (using Google Earth or by applying the Google Maps API). The software packages and tools adopted to support this phase were the Quantum GIS, the GDAL/OGR libraries and the Postgresql/PostGIS systems.

Phase 2 - Design of the WMS, WFS and WCS Servers

In this phase the Map Server was implemented, and the software system used was the UMN MapServer. Using the Quantum GIS (ver. 0.8.1; tool: Export to MapServer Map…) the map symbols assigned in the previous Phase were mapped into appropriate mapfiles, which were then used by the MapServer. The mapfiles were then edited, and appropriate parameters were inserted manually in order to disseminate that geospatial data which resides at the bottom SDI layer into the application layer - using the OGC specifications of WMS, WFS, and WCS. Geospatial data could then be retrieved and visualized by end users in various formats (e.g., PGN, JPG and GML). The access and usability of the WMS, WFS, WCS and KML servers has been enhanced by us building a web page, which is presented below in Section 4. The software packages and tools that were adopted to support this phase were the Quantum GIS, and the UMN MapServer.

Phase 3 - Development of the Web Client Application

This phase focused on the generation of a user-friendly interface in order to access the SDI geospatial content through the web. The web client application has been developed in Javascripts and connected with the geospatial layers provided by other servers (e.g., Google Maps, NASA JPL Portal Nasa's WMS, Integrated CEOS European Data Server – Mashups). This the web client application enables browsing/visualization of the SDI geospatial layers and the retrieval/discovery of information which is based on relatively simple geospatial analysis operations. More details are given below in Section 5.2. Its implementation was based on the Openlayers Javascripts library.

Phase 4 - Development of the Catalog Server

This phase set up the data catalog for the geospatial content (i.e., data layers, satellite images and applications). Appropriate metadata items were generated and coded, based on international standard (ISO 19115/139) and these items were diffused on the web through the catalog server. Implementation of the catalog server, as well as the dissemination of the SDI metadata items, are supported by the GeoNetwork Opensource software system.

4.0 The WMS, WFS, WCS and KML servers

It should now be obvious that a major task during the development of the Heraklion SDI was the generation of appropriate map servers/services in order to provide (serve) the SDI content onto the web and so make it easily accessible by a wide community of users. These servers are the WMS, WFS, and WCS. Since we had already mapped the geospatial layers into the KML format (in Phase 1) a fourth server, the KML server, was also implemented.

It is widely recognized, that the WMS, WFS, WCS and KML servers constitute very powerful tools in contemporary web mapping and web GIS technology, provided that the SDI geospatial layers can be viewed or shared on the web using widely adopted standards and specifications (Mitchell, 2005; Stefanakis and Prastacos, 2007).

On the other hand, it is also widely accepted that the use and exploitation of content provided in similar servers is cumbersome, especially for non experienced users (Mitchell, 2005). Accordingly, in order to assist end users and applications to retrieve and visualize SDI geospatial content through the servers above, a detailed web page has been generated, as shown in Figure 3 (Stefanakis and Prastacos, 2008). Users are able to browse the SDI content and to easily access and exploit it by using an open source WMS/WFS Viewer (e.g., Quantum GIS WMS/WFS Plugins, Integraph OGC WMS Viewer - see Section 5.1).

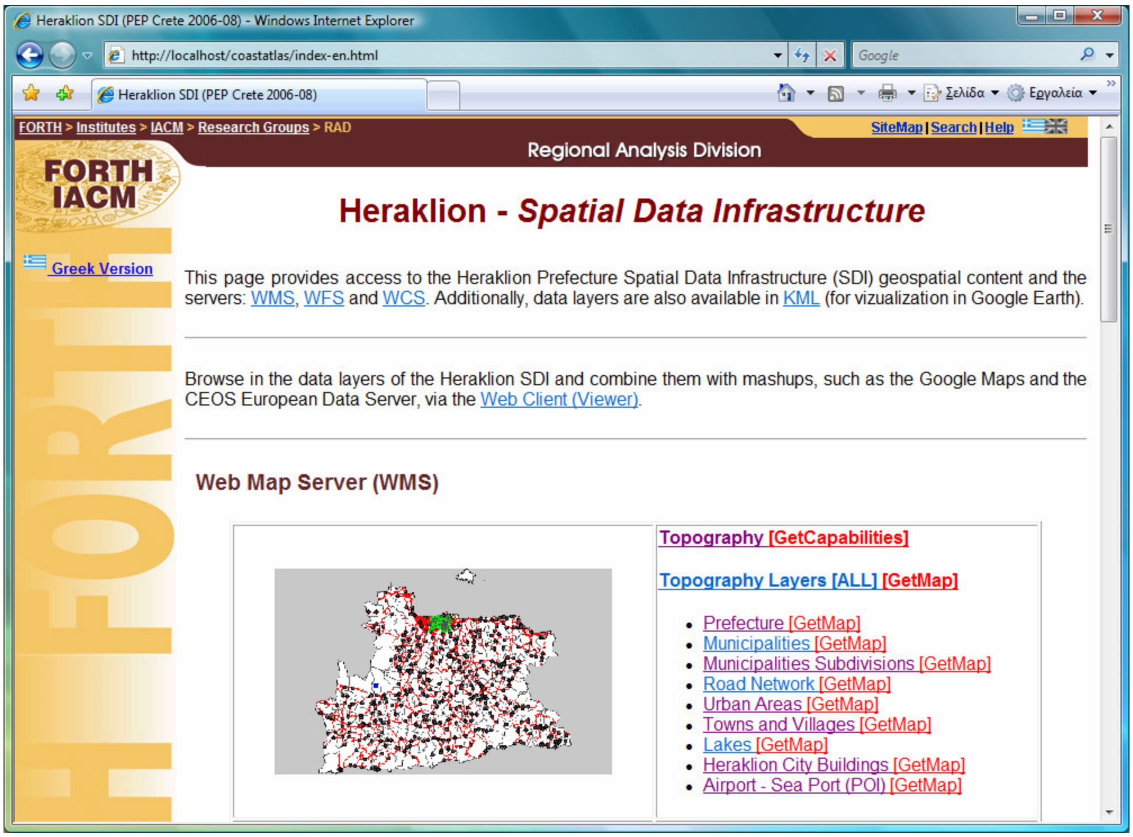


Figure 3 - The introductory web page to the WMS, WFS, WCS and KML servers (part of).

This web page consists of four main sections, each section being devoted to one server, and Figure 4 shows the *Web Map Server (WMS)* section by way of example. This WMS organizes the SDI layers into two groups, with the top group in Figure 4 (i.e., Topography) providing all of the vector layers (Shapefiles) listed in Table 1 above except for those of geology and points of archaeological interest. The latter two layers are part of the bottom group (i.e., Archaeological Spots and Geology). Each group has a link to the capabilities document (GetCapabilities) and a set of links to the actual maps (as images), as served by the WMS (GetMap).

The capabilities document is actually a reference to the corresponding WMS, and capabilities documents play an integral role in effective web services. They contain all the metadata of the WMS and they describe in XML what information is available and how to access it (Mitchell, 2005; OGC). Note that in the SDI web page any link to a capability document hides the URL for a WMS. For instance, the 'Topography [GetCapabilities]' link is attached to the following URL (where */WMS/infocharta.map* is the mapfile that defines the WMS service):

http://localhost/cgi-bin/mapserv.exe?
map=/ms4w/apps/ITE/htdocs/WMS/infocharta.map&
SERVICE=**WMS**&
REQUEST=**GetCapabilities**

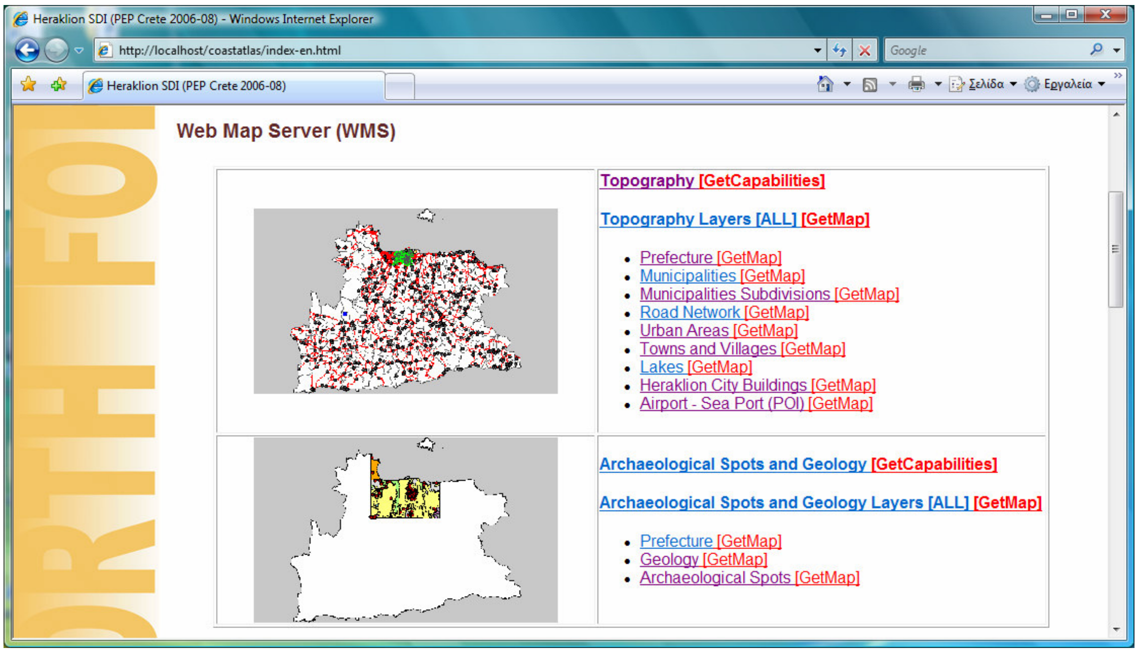


Figure 4 - The WMS section at the Heraklion SDI web page.

Note also that the links to the SDI layers (GetMap) hide the WMS requests (in URL) for each individual layer or for all of them as a group. For example, the following URL requests all layers ('Topography Layers [ALL]') comprising the Topography group:

http://localhost/cgi-bin/mapserv.exe?
map=/ms4w/apps/ITE/htdocs/WMS/infocharta.map&
SERVICE=**WMS**&
VERSION=1.1.1&
REQUEST=**GetMap**&
LAYERS=nomos_irakliou,bua_irakliou,dhmoi_irakliou,dhm_diamer_irakliou,
limnes_irakliou,odiko_irakliou,oikismoi_irakliou,build_egsa,
poi_irakliou&
BBOX=553530,3864020,653540,3925230&
STYLES=&
SRS=EPSG:2100&   // Hellenic Geodetic Reference System EGSA'87
WIDTH=500&HEIGHT=306&
FORMAT=image/png

Figure 5 presents the *Web Feature Service (WFS)* section of the SDI web page, which provides the same SDI layers as in WMS section, with a similar organization into two groups. Each group has a link to the WFS capabilities document (GetCapabilities) and a set of links to the actual layers (as features in GML) of the WFS (GetFeature - GML).

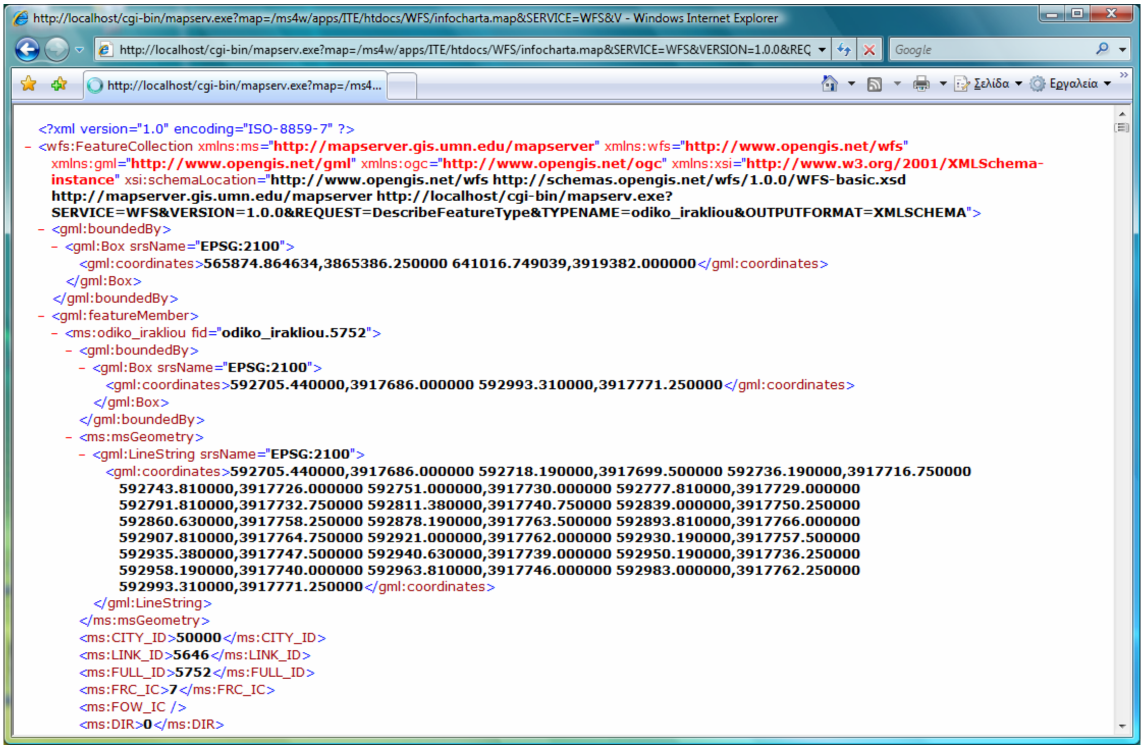
Figure 5 - The WFS section at the Heraklion SDI web page.

The capabilities of each WFS are accessed through the corresponding link in the web page, which hides a URL request. For example, the following URL refers to the capabilities document of the Topography layers, i.e., 'Topography [GetCapabilities]' (where */WFS/infocharta.map* is the mapfile that defines the WFS service):

http://localhost/cgi-bin/mapserv.exe?
map=/ms4w/apps/ITE/htdocs/WFS/infocharta.map&
SERVICE=**WFS**&
VERSION=1.0.0&
REQUEST=**GetCapabilities**

Also, the actual features (objects) of each WFS layer may be retrieved in GML format (see Figures 6 and 7) through the corresponding links (URL requests) on the SDI web page. For instance, the following URL requests the features of the road network (named as "odiko_irakliou") within the Topography group, i.e., 'Road Network [GetFeature–GML]':

http://localhost/cgi-bin/mapserv.exe?
map=/ms4w/apps/ITE/htdocs/WFS/infocharta.map&
SERVICE=**WFS**&
VERSION=1.0.0&
REQUEST=**GetFeature**&
typename=odiko_irakliou




```
<?xml version="1.0" encoding="ISO-8859-7" ?>
- <wfs:FeatureCollection xmlns:ms="http://mapserver.gis.umn.edu/mapserver" xmlns:wfs="http://www.opengis.net/wfs"
  xmlns:gml="http://www.opengis.net/gml" xmlns:ogc="http://www.opengis.net/ogc" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
  instance" xsi:schemaLocation="http://www.opengis.net/wfs http://schemas.opengis.net/wfs/1.0.0/WFS-basic.xsd
  http://mapserver.gis.umn.edu/mapserver http://localhost/cgi-bin/mapserv.exe?
  SERVICE=WFS&VERSION=1.0.0&REQUEST=DescribeFeatureType&TYPENAME=odiko_irakliou&OUTPUTFORMAT=XMLSCHEMA">
 - <gml:boundedBy>
  - <gml:Box srsName="EPSG:2100">
     <gml:coordinates>565874.864634,3865386.250000 641016.749039,3919382.000000</gml:coordinates>
    </gml:Box>
   </gml:boundedBy>
 - <gml:featureMember>
  - <ms:odiko_irakliou fid="odiko_irakliou.5752">
   - <gml:boundedBy>
    - <gml:Box srsName="EPSG:2100">
       <gml:coordinates>592705.440000,3917686.000000 592993.310000,3917771.250000</gml:coordinates>
      </gml:Box>
     </gml:boundedBy>
   - <ms:msGeometry>
    - <gml:LineString srsName="EPSG:2100">
       <gml:coordinates>592705.440000,3917686.000000 592718.190000,3917699.500000 592736.190000,3917716.750000
         592743.810000,3917726.000000 592751.000000,3917730.000000 592777.810000,3917729.000000
         592791.810000,3917732.750000 592811.380000,3917740.750000 592839.000000,3917750.250000
         592860.630000,3917758.250000 592878.190000,3917763.500000 592893.810000,3917766.000000
         592907.810000,3917764.750000 592921.000000,3917762.000000 592930.190000,3917757.500000
         592935.380000,3917747.500000 592940.630000,3917739.000000 592950.190000,3917736.250000
         592958.190000,3917740.000000 592963.810000,3917746.000000 592983.000000,3917762.250000
         592993.310000,3917771.250000</gml:coordinates>
      </gml:LineString>
     </ms:msGeometry>
     <ms:CITY_ID>50000</ms:CITY_ID>
     <ms:LINK_ID>5646</ms:LINK_ID>
     <ms:FULL_ID>5752</ms:FULL_ID>
     <ms:FRC_IC>7</ms:FRC_IC>
     <ms:FOW_IC />
     <ms:DIR>0</ms:DIR>
```


Figure 6 - The header and one feature from the road network in GML as retrieved from the WFS (‘Road Network [GetFeature–GML]’ link in Figure 5)

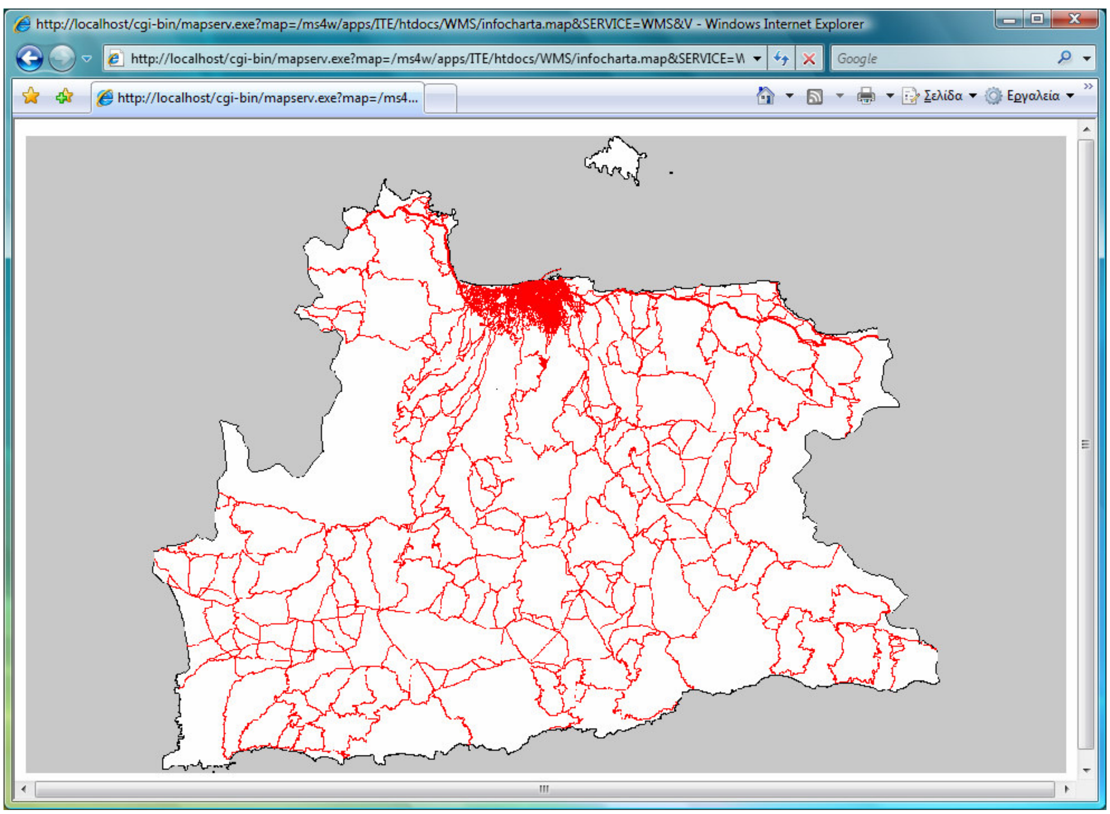


Figure 7 - The road network (“odiko_irakliou”) as retrieved from the WMS (‘Road Network [GetMap]’ link in Figure 4).

Notice that all features retrieved in GML format are the actual objects (vectors) that comprise the road network. This is the main difference between WMS and WFS services. The former returns images, whereas the latter returns geographical features in GML format - the most popular format for describing geographical entities. Moreover, GML output can be imported into any GIS software package as a data layer (e.g., shapefile) for further processing.

Figure 8 shows the *Web Coverage Server (WCS)* section of the SDI web page. It provides those raster (GeoTiff) SDI layers that are listed in Table 1 above. As with the WMS and WFS sections, the end user is provided with a link (URL request) to the WCS capabilities document (GetCapabilities), as well as a set of links to the actual images of the WCS (GetMap).

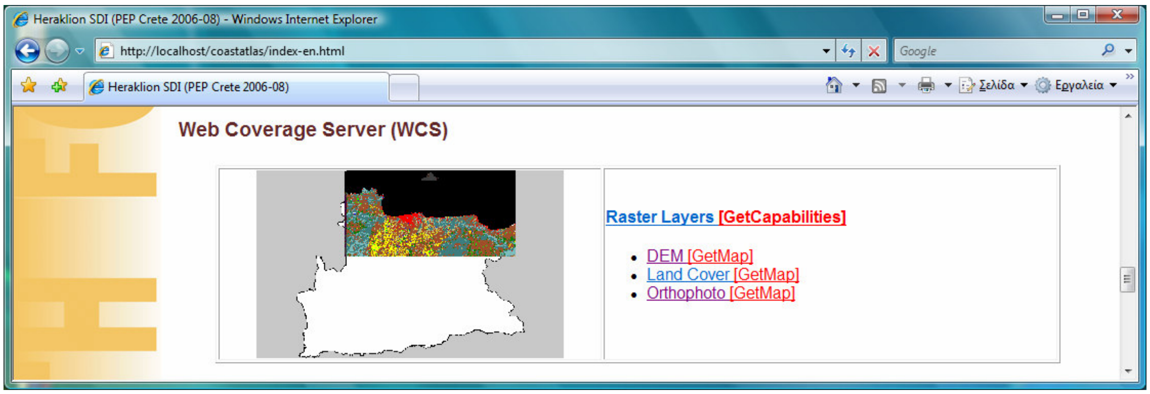


Figure 8 - The WCS section of the Heraklion SDI web page.

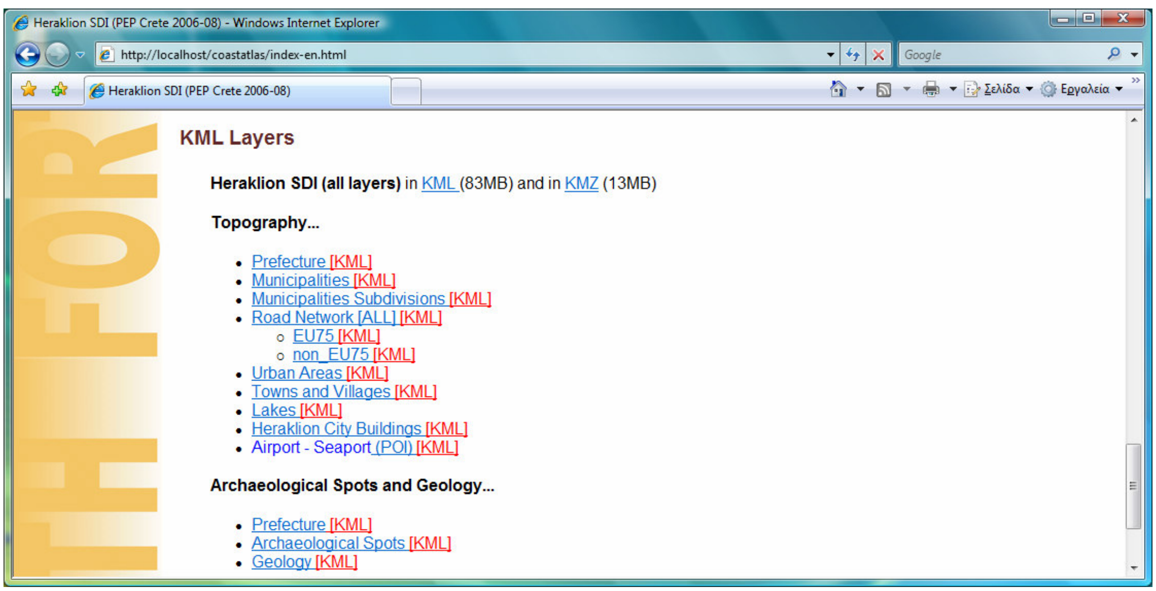


Figure 9 - The KML section at the Heraklion SDI web page.

Figure 9 shows the *Keyhole Markup Language (KML)* section of the SDI web page. Here the end user is able to download KML and zipped KML (KMZ) files of the SDI geospatial layers. The latter have been generated using the *ogr2ogr* utility provided in GDAL/OGR Geospatial Library (see Phase 1). The output of this utility is loaded into Google Earth, and appropriate symbols are assigned to the features, i.e., place marks, paths and polygons. The edited layers are then saved into KML and uploaded to the KML server for diffusion via this web page. End-users may visualize such layers either in Google Earth or on top of Google Maps using the Google Maps API. For example, Figure 10 shows the Heraklion SDI geospatial layers draped over satellite Google Maps.

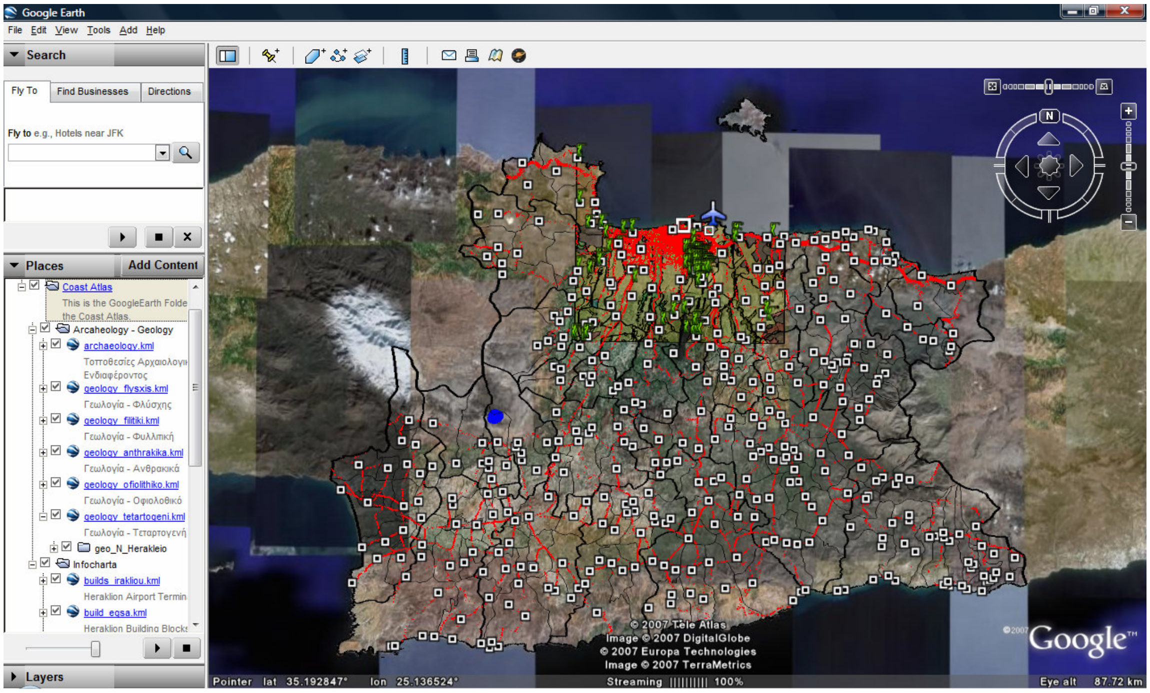


Figure 10 - Visualizing the Heraklion SDI geospatial layers in Google Earth.

5.0 Visualizing the geospatial content

As shown in the SDI Architecture (Figure 2, top layer), the geospatial content may be visualized from either a *desktop client* or a *web client*. In this project the *Quantum GIS (QGIS)* was adopted in the role of the desktop client. Additionally, Google Earth was used for visualizing the KML files provided by the KML Server (Section 4). A web client interface was also implemented in *OpenLayers JavanScripts*. In the following subsections, therefore, the functionality of both the QGIS desktop client and the OpenLayers web client application will be presented with respect to the visualization of the geospatial content of the SDI. Additionally, some alternative tools that may also be useful to SDI clients (end-users) are briefly presented.

5.1 Visualizing the SDI Content using the QGIS Desktop Client

Because *Quantum GIS (QGIS)* has enhanced capabilities, it has been enthusiastically adopted as the Desktop Client of the SDI for the Heraklion Prefecture. QGIS can be connected to the Map Servers, i.e. WMS, WFS and WCS, as developed in the middle layer (Section 4), and it can provide advanced visualization of their content. The process of connecting and retrieving content from Map Servers is assisted by appropriate wizards, which relieve the user of the burden of composing the URL parameters in text mode, as presented in the Section 4. Additionally, QGIS users may subsequently analyze the geospatial content by exploiting its rich analytical functionality.

As regards the SDI's WMS server, QGIS can currently act as a *WMS client* which understands WMS 1.1, 1.1.1 and 1.3 servers (see Figure 11). Specifically, QGIS can be easily connected to the WMS Server of the Heraklion SDI [1] by performing a GetCapabilities Request through the appropriate WMS GetCapabilities document URL [2] (see Section 4 above). Then, the QGIS user selects from the list of the available layers [3] and the GetMap

Request is formed and executed to the WMS server. The selected layers are retrieved and shown on the QGIS Map View frame as a PGN or JPEG *image* [4] (see Section 4 above).

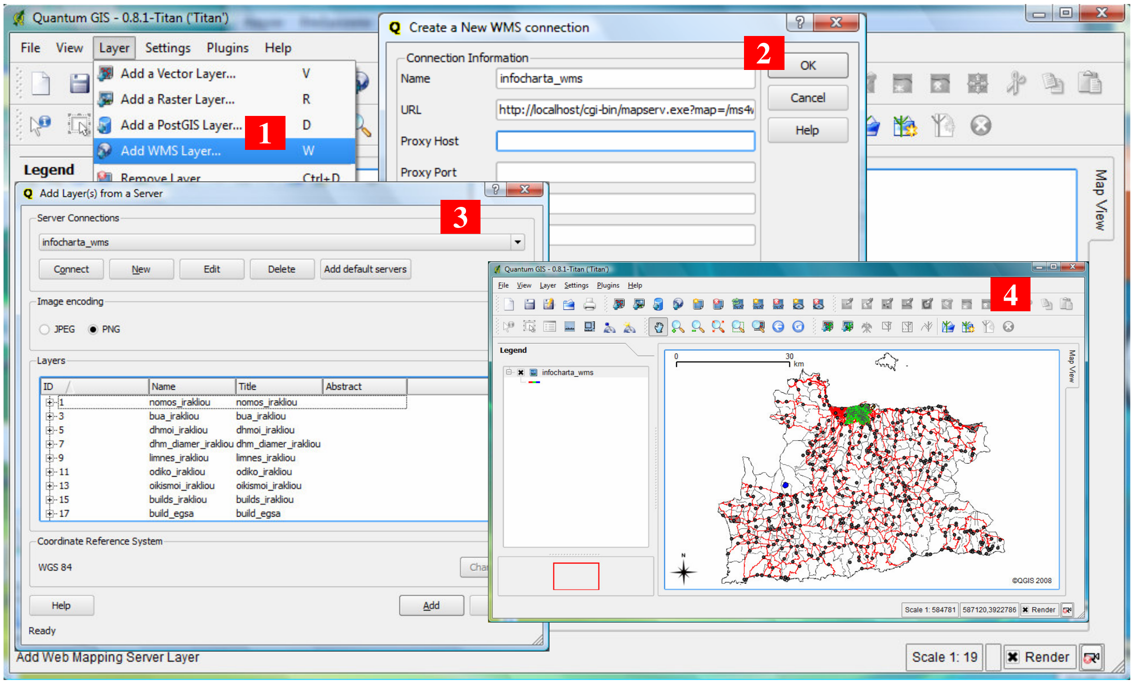


Figure 11 - Visualizing the Heraklion SDI WMS server layers into the QGIS (the numbered squares correspond to the sequential actions described in the text).

Also, QGIS currently offers a plugin (under optimization) to *add WFS layers for processing,* as shown in Figure 12. Obviously, all feature components are added in the QGIS, i.e., both the spatial and thematic (non-spatial) attributes. In other words, the QGIS can be easily connected to the WFS Server of the Heraklion SDI [1] by performing a GetCapabilities Request through the appropriate WFS GetCapabilities document URL [2] (see Section 4 above). Then the QGIS user selects from the list of the available layers [3] and the GetFeature Request is formed and executed to the WFS server. The selected layers are retrieved and shown on the QGIS Map View screen as *vectors*, while their non-spatial attributes are also loaded into the attribute table [4] (see Section 4 above). In a similar procedure with WMS, the QGIS may be connected to, and used to visualize the thematic layers available in the WCS server of the Heraklion SDI (see Section 4 above).

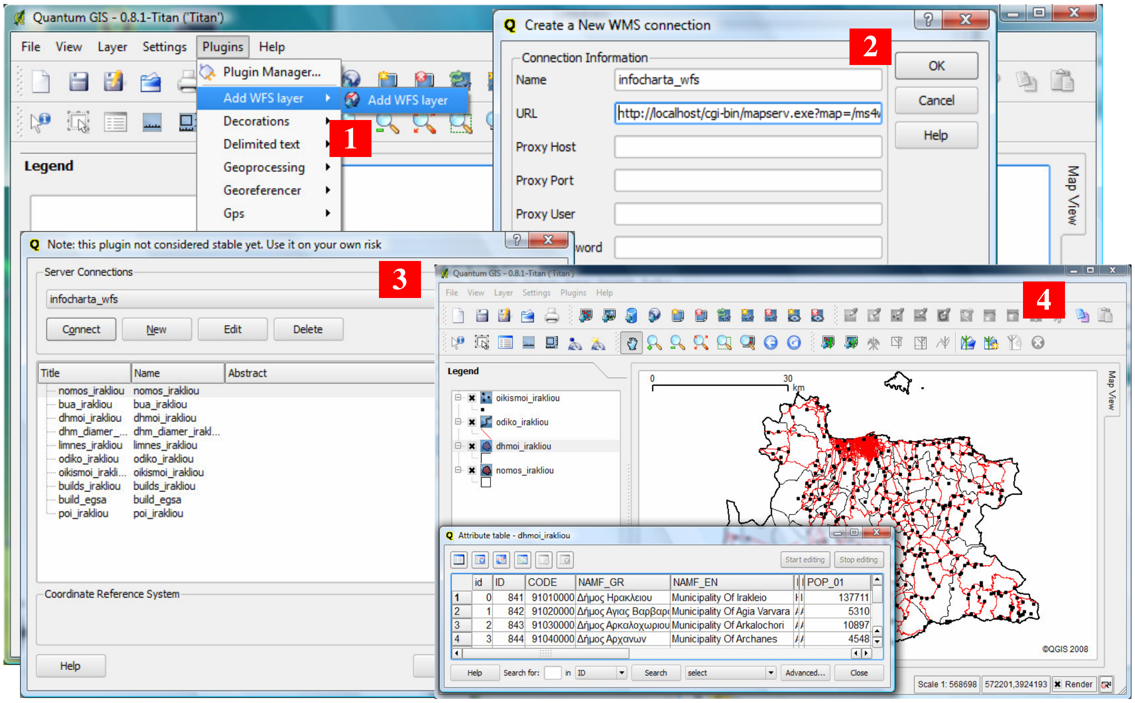


Figure 12 - Adding the Heraklion SDI WFS server layers into QGIS (the numbered squares correspond to the sequential actions described in the text).

5.2 Visualizing the SDI Content using the OpenLayers Web Client Application

An alternative way to visualize the layers of the Heraklion SDI is offered by the *Web Client Application* that has been implemented using the *OpenLayers JavaScripts API* (see Section 2.3.5 above). The interface of this application is shown in Figure 13 and it consists of a graphical window, where *one base map* and *a set of layers* are overlaid. The base map and the layers may be selected by clicking the radio and the multiple selection buttons, respectively, on the right side of the window.

The interface provides some basic operations for browsing the graphical window (e.g., zoom in/out, pan), while the scale and the mouse coordinates are continuously reported. At the bottom of the interface, a few lines are devoted to providing some instructions for initial use (i.e., when the application is loaded) and then report attribute values for the clicked features at the thematic layers - Municipalities (polygonal features, the name and the population are reported); Cities/villages (point features, the name and the population are reported) and Road Network (linear segments, the road category, the speed limit and the segment length are reported).

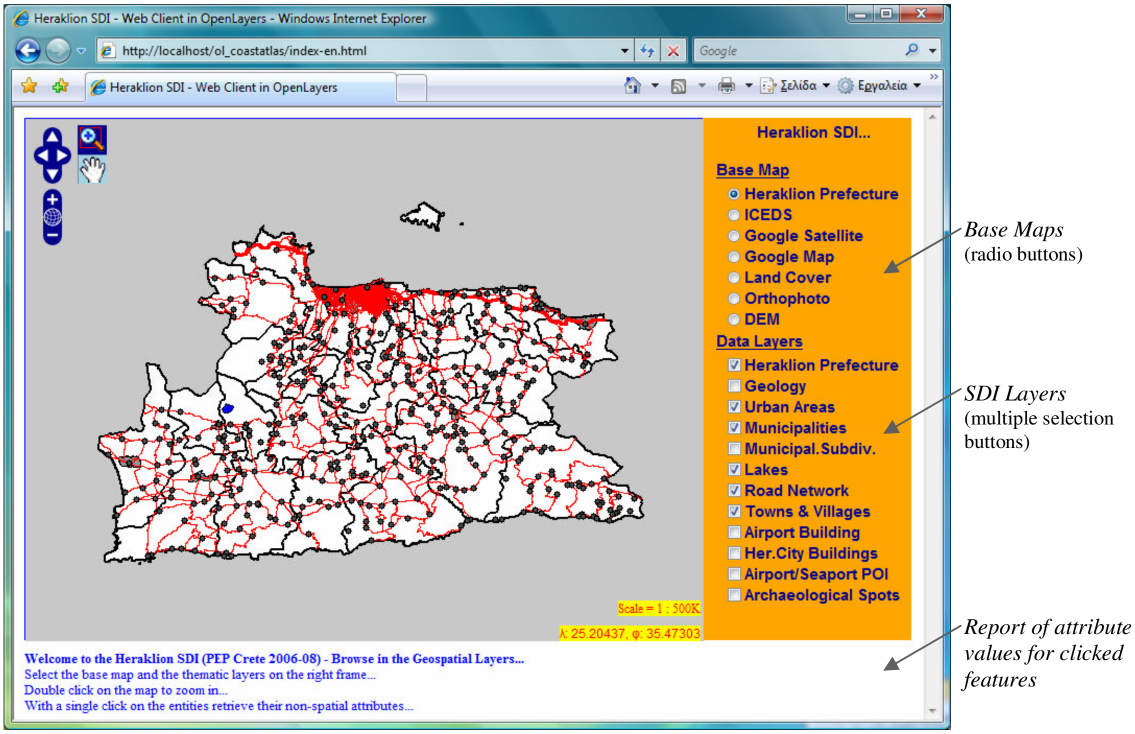
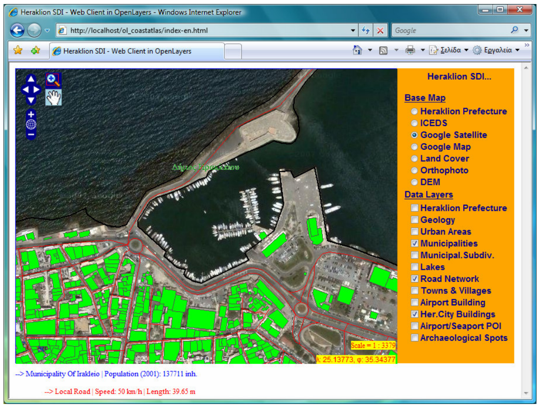


Figure 13 - The Web Client Application implemented in OpenLayers API.

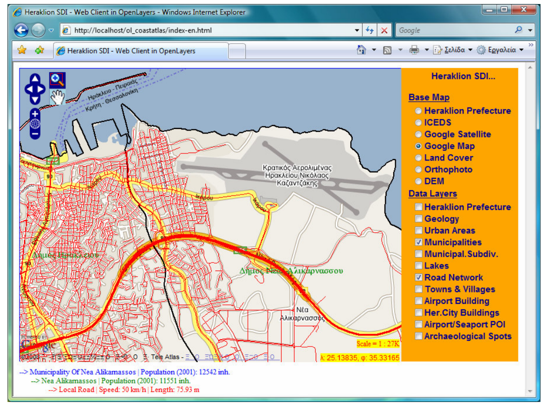
It must be pointed out that the Web Client Application permits end-users to generate *mashups* of the SDI layers with Google Maps (normal map and satellite image), the Integrated CEOS European Data Server (ICEDS WMS) and the raster images offered by the WCS server of the Heraklion SDI (see Table 1, last three lines). Figure 14 presents some examples of these mashups.

Access to the *base maps* is achieved through the use of appropriate classes offered by the OpenLayers JavaScripts API. More exactly, access to the base maps of google satellite, google map, ICDES WMS and the Heraklion SDI's orthophoto map is achieved through the following four commands, respectively (the */WMS/necdata_ol.map* represents the mapfile that defines the WMS service):

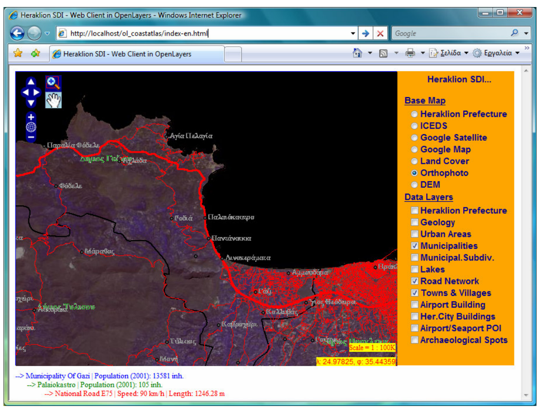
```
google_sat = new OpenLayers.Layer.Google( "Google Satellite" ,
        {type: G_SATELLITE_MAP} );
google_map = new OpenLayers.Layer.Google( "Google Map" , {type: G_NORMAL_MAP });
icdes = new OpenLayers.Layer.WMS.Untiled( "ICEDS",
        "http://iceds.ge.ucl.ac.uk/cgi-bin/icedswms?", {layers: "srtm"});
ortho_map = new OpenLayers.Layer.WMS.Untiled( "Orthophoto",
        "http://localhost/cgibin/mapserv.exe?
        map=/ms4w/apps/ITE/htdocs/WMS/necdata_ol.map",{layers: "hr_ortho"} );
```



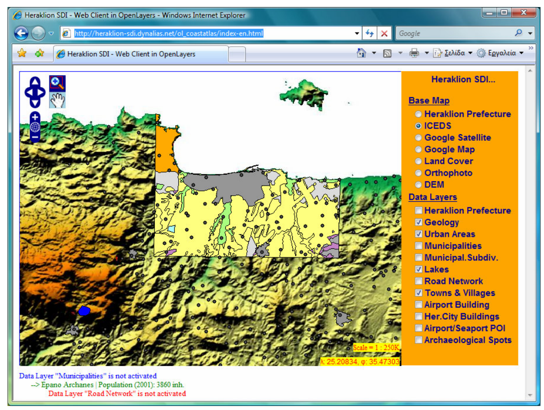
Base Map: Google Satellite
Layers: Municipalities, Road Network, Buildings



Base Map: Google Map
Layers: Municipalities, Road Network



Base Map: Orthophoto Map
Layers: Municipalities, Towns, Road Network



Base Map: ICEDS (WMS)
Layers: Geology, Urban areas, Towns, Lakes

Figure 14 - Example mashups from the Web Client Application of the Heraklion SDI.

Alternatively, access to the Heraklion *SDI layers* can be accomplished through the OpenLayers.Layer.WMS class available within the OpenLayers JavaScripts Library. This class accesses the WMS server implemented by the UMN MapServer (see Section 4) to retrieve the requested layers. Finally, the following two commands perform the access to, and retrieval of the WMS layers prefecture (nomos_irakliou) and road network (odiko_irakliou) respectively (the */WMS/infocharta_ol.map* is the mapfile that defines the WMS service):

```
nomos_irakliou = new OpenLayers.Layer.WMS.Untiled( "Heraklion Prefecture",
        "http://localhost/cgi-bin/mapserv.exe?
        map=/ms4w/apps/ITE/htdocs/WMS/infocharta_ol.map",
        {layers: "nomos_irakliou", 'transparent':'true'} , {'reproject': true} );
odiko_irakliou = new OpenLayers.Layer.WMS.Untiled( "Road Network",
        "http://localhost/cgi-bin/mapserv.exe?
        map=/ms4w/apps/ITE/htdocs/WMS/infocharta_ol.map",
        {layers: "odiko_irakliou", 'transparent':'true'} , {'reproject': true});
```

The Web Client Application enables one to *access the non-spatial attributes* that are associated with the features drawn in the graphical window and to report them, as mentioned previously on the bottom lines of the user interface. This is accomplished through the GetFeatureInfo request, which is also (like GetCapabilities and GetMap) a WMS request (OGC). The OpenLayer API implements this request through the Layer.getFullRequestString class. Finally, the command used to retrieve the name and the population of the municipalities (dhmoi_irakliou) is the following:

```
dhmoi_irakliou.getFullRequestString({
        REQUEST: "GetFeatureInfo",
        EXCEPTIONS: "application/vnd.ogc.se_xml",
        BBOX: dhmoi_irakliou.map.getExtent().toBBOX(),
        X: e.xy.x, Y: e.xy.y,
        INFO_FORMAT: "text/html",
        FONT: "c:/ms4w/fontlist/times.ttf",
        ENCODING: "ISO-8859-7",
        QUERY_LAYERS: dhmoi_irakliou.params.LAYERS,
        WIDTH: dhmoi_irakliou.map.size.w,
        HEIGHT: dhmoi_irakliou.map.size.h});
```

5.3 Visualizing the SDI Content using Alternative Tools

The Heraklion SDI content may also be visualized using still other configurations and tools. For example, one alternative is to use the *ogr2ogr* function that is available at GDAL/OGR Geospatial Data Abstraction Library (see Section 2.3.3). This function, among others, may convert the format of a GML file (provided by WFS server) or of a PostgreSQL/PostGIS table (or query result; see Section 6 below), to make it into a KML or ESRI Shapefile. The latter two may be then visualized using Google Earth, Quantum GIS or ESRI ArcGIS software packages.

Also, one can make use of the command line tools offered by PostgreSQL/PostGIS, such as the *pgsql2shp*, which converts PostGIS tables (or queries) into ESRI Shapefiles. Moreover, PostGIS provides a function that returns the geometry of a query into SVG format (W3C), and it can be viewed using a web browser (e.g., MS Explorer, Netscape), provided that an SVG Viewer plugin has been loaded (e.g., Adobe SVG Viewer).

Finally, MapServer also provides the *shp2img* command line utility, which converts ESRI Shapefiles, which may have been generated using the actions described in the previous paragraphs, into an image PNG file. The latter may be viewed using a Web Browser or an Image Viewer/Editor.

6.0 The direct access interface to the database server

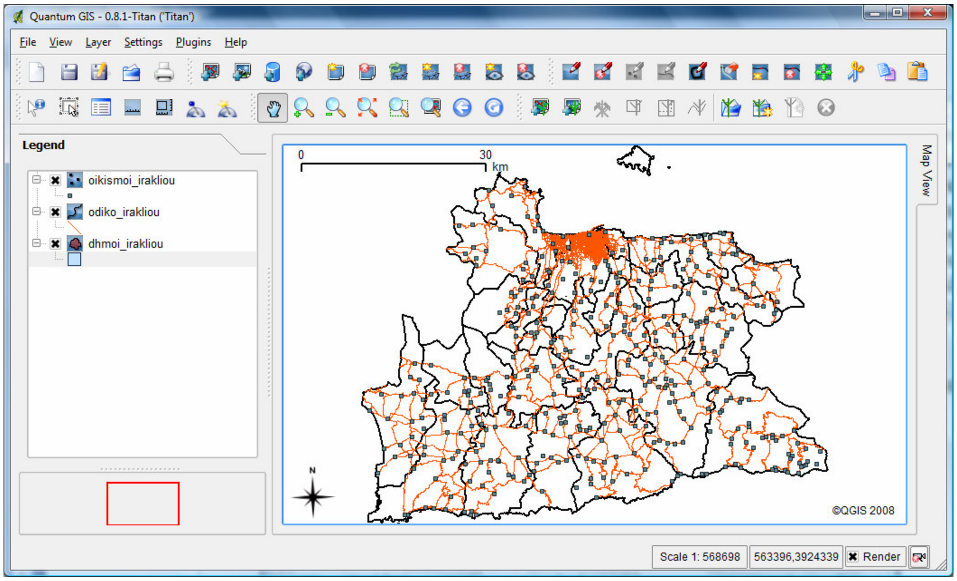
According to the Open SDI architecture as shown in Figure 2 above, the middle layer provides the *Direct Access Interface* to the data stores which reside in the spatial database server. The spatial database server has been implemented in the *PostgrSQL/PostGIS* software package (see Section 2.3.1). Indeed, this product can effectively support both the *querying* and the *analysis* of geospatial content using *SQL statements*.

It is widely argued that future GIS desktop applications will be little more than products for visualization with GIS functionality, taking place within the spatial database (Mitchell, 2005), and the Direct Access Interface offered by the middle layer of the Heraklion SDI moves in this direction. In this Section, therefore, a series of examples will show the available SQL statements for querying and analysis. Note that the Quantum GIS (QGIS) has been used as a GIS desktop visualizer.

The SQL statements that follow involve three layers (tables), which are stored in the database server:

a) the municipalities layer, named as “dhmoi_irakliou”,
b) the cities/ villages layer, named as “oikismoi_irakliou”, and
c) the road network layer, named as “odiko_irakliou” (see Table 1 above).

These layers are visualized, using the QGIS, in Figure 15, which also shows a few non-spatial attributes attached to the layers and involved in the SQL queries. Notice that a spatial index has been built within the geometry column (the_geom) of all three tables.



- **Table of Municipalities: "dhmoi_irakliou" (type: multi_polygon)**

SELECT gid, NAMF_EN, POP_01, AsText(the_geom)
FROM dhmoi_irakliou;

gid (identifier)	NAMF_EN (municipality name)	POP_01 (population in 2001)	the_geom (geometry column)
1	Municipality Of Irakleio	137711	MULTIPOLYGON(…)
2	Municipality Of Agia Varvara	5310	MULTIPOLYGON(…)
3	Municipality Of Arkalochori	10897	MULTIPOLYGON(…)
4	Municipality Of Archanes	4548	MULTIPOLYGON(…)
…	…	…	…

26 rows

- **Table of Cities/Villages: "oikismoi_irakliou" (type: point)**

SELECT gid, NAMEENG, POP01, AsText(the_geom)
FROM oikismoi_irakliou;

gid (identifier)	NAMEENG (city/village name)	POP01 (population in 2001)	the_geom (geometry column)
1	Agia Pelagia	553	POINT(592221 3918593)
2	Paralia Fodele	99	POINT(586237 3917962)
3	Achlada	119	POINT(589949 3917093)
4	Fodele	540	POINT(586850 3915575)
…	…	…	…

400 rows

- **Table of Road Network: "odiko_irakliou" (type: multi_linestring)**

SELECT gid, SPEED, EU_CODE, AsText(the_geom)
FROM odiko_irakliou;

gid (identifier)	SPEED (speed limit)	EU_CODE (E75 for national roads)	the_geom (geometry column)
1	50		MULTILINESTRING(…)
2	50		MULTILINESTRING(…)
3	80	E75	MULTILINESTRING(…)
4	70	E75	MULTILINESTRING(…)
…	…	…	…

12228 rows

Figure 15 - The three layers (tables) involved in Figure 16's example SQL statements.

Figure 16 lists a few representative SQL statements for the layers (tables) shown in Figure 15. These demonstrate the advanced querying and analysis capabilities available when using the direct access interface of the Heraklion SDI's middle layer (see Figure 2 above).

Query 1: Find how many cities there are per municipality.

SQL Statement

```
SELECT r.NAMF_EN as Municipality,
  count(m.the_geom) as Number
FROM dhmoi_irakliou AS r,
    oikismoi_irakliou AS m
WHERE intersects(r.the_geom, m.the_geom)
GROUP BY r.NAMF_EN
ORDER BY number_of_cities DESC;
```

Output

```
Municipality                 | Number
-----------------------------+---------
Municipality Of Viannos      | 46
Municipality Of Arkalochori  | 40
Municipality Of Asterousia   | 28
Municipality Of Gortyna      | 26
Municipality Of Kasteelli    | 24
...
```

Query 2: What is the length of roads fully contained within each municipality? Report only the 5 largest.

SQL Statement

```
SELECT m.NAMF_EN as Municipality,
 sum(length(r.the_geom))/1000 as Roads_km
FROM odiko_irakliou AS r,
    dhmoi_irakliou AS m
WHERE r.the_geom && m.the_geom
AND contains(m.the_geom,r.the_geom)
GROUP BY m.NAMF_EN
ORDER BY roads_km DESC
LIMIT 5;
```

Output

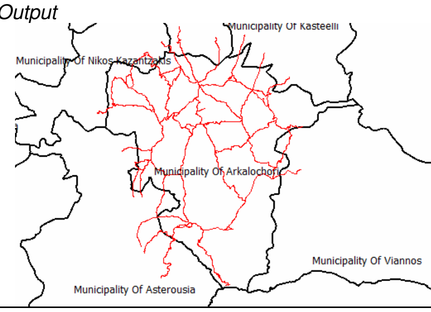
```
Municipality                 | Roads_km
-----------------------------+---------
Municipality Of Irakleio     | 595.440
Municipality Of Gazi         | 200.706
Municipality Of Arkalochori  | 147.853
Municipality Of Asterousia   | 146.120
Municipality Of Viannos      | 137.159
-----------------------------+---------
```

Query 3: Find the road segments intersected by the municipality of Arkalochori

SQL Statement

```
CREATE TABLE "ark_roads"
  (gid serial PRIMARY KEY,"id" int4);
SELECT AddGeometryColumn('','ark_roads',
  'the_geom','2100','MULTILINESTRING',2);
INSERT INTO ark_roads(id, the_geom)
SELECT r.gid, r.the_geom
FROM odiko_irakliou AS r,
    dhmoi_irakliou AS m
WHERE r.the_geom && m.the_geom
AND intersects(m.the_geom,r.the_geom)
AND m.NAMF_EN =
   'Municipality Of Arkalochori';
```

Output

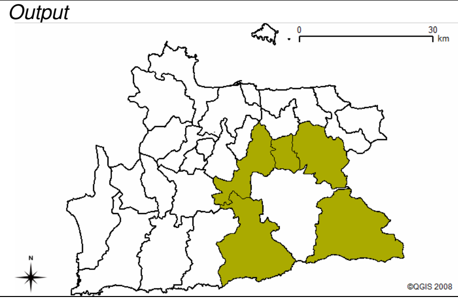


Query 4: Find the neighbors (with a common border) of the municipality of Arkalochori.

SQL Statement

```
CREATE TABLE "ark_neigh"
  (gid serial PRIMARY KEY,"id" int4);
SELECT AddGeometryColumn('','ark_neigh',
  'the_geom','2100','MULTIPOLYGON',2);
INSERT INTO ark_neigh (id, the_geom)
SELECT n.gid, n.the_geom
FROM dhmoi_irakliou as m,
    dhmoi_irakliou as n
WHERE m.NAMF_EN =
     'Municipality Of Arkalochori'
AND Touches(m.the_geom, n.the_geom);
```

Output



Query 5: Create a buffer zone of 10 km around the village of Larani.

SQL Statement

```
CREATE TABLE "buffer"
  (gid serial PRIMARY KEY,"id" int4);
SELECT AddGeometryColumn('','buffer',
  'the_geom','2100','POLYGON',2);
INSERT INTO buffer (id, the_geom)
SELECT id, Buffer(the_geom, 10000, 16)
FROM oikismoi_irakliou
WHERE NAMEENG = 'Larani';
```

Output

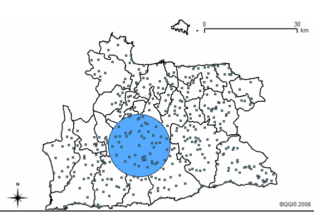


Figure 16 - Example SQL statements for querying and analyzing the layers (tables) shown in Figure 15. The QGIS has been used to visualize the results.

7.0 The Catalog Server

It is obvious from the previous sections, that the Heraklion SDI accommodates many digital types and formats, ranging from geospatial data layers and satellite images to web mapping applications and services. In order to make this content accessible on the web, and so assure its usability, two actions are required (see Section 1 above):

a) appropriate *metadata* items must be generated, and

b) a *data catalog server* is needed to make these metadata items available on the web and support the efficient discovery and evaluation of the SDI content.

These actions have been implemented, based on the specifications and protocols proposed by the ISO/TC211 and the OGC. Specifically, all data layers of the Heraklion SDI shown in Table 1 above and the SDI applications, including the web client application (see Section 5.2), have been assigned appropriate metadata items in XML format and conforming to a customized *ISO19139 template*. Then a catalog server has been implemented using GeoNetwork Opensource Server software version 2.1 (see Section 2.3.6). The configuration adopted utilizes the PostgreSQL in the role of the DBMS server, and the Apache Tomcat v5.5 in the role of the Web Server.

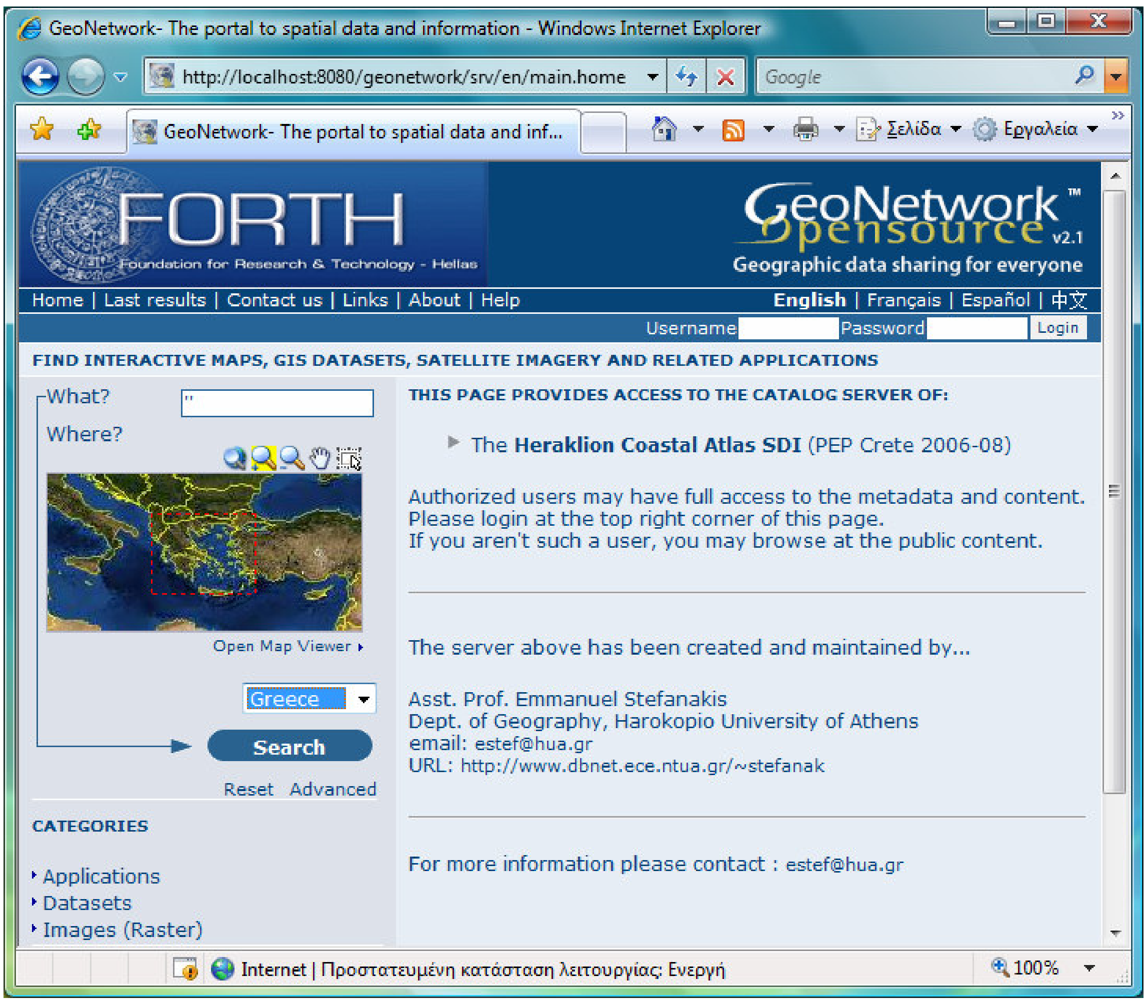
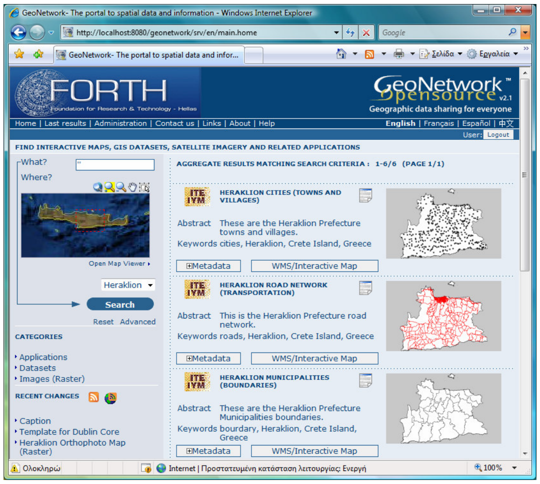


Figure 17 - The initial screen of the Catalog Server for the Heraklion SDI.

Figure 17 shows the initial screen of the Catalog Server, which gives authorized users full access to the metadata items and content of the SDI after logging in on the top right. The catalog lists all public data layers (see Table 1) and the web client application (see Section

5.2) to non-authorized users, as well as mechanisms for discovering and evaluating this content. As depicted at the bottom-left corner, the content has been organized into three categories, i.e., Applications, Datasets (vector data), and Images (raster data).

Figure 18 provides the snapshots of the interface after clicking on the three categories. Each retrieved item has a title, an abstract description, a list of keywords and two buttons - the “metadata” button, which retrieves its whole metadata description, and the “WMS/Interactive Map” button which loads either the corresponding WMS map or the web client application (see Sections 4 and 5).



(a)

(b)

(c)

Figure 18 - Snapshots from the three categories of metadata items within the public content of the Heraklion SDI: (a) the datasets (vector layers), (b) the raster images, and (c) the web client application.

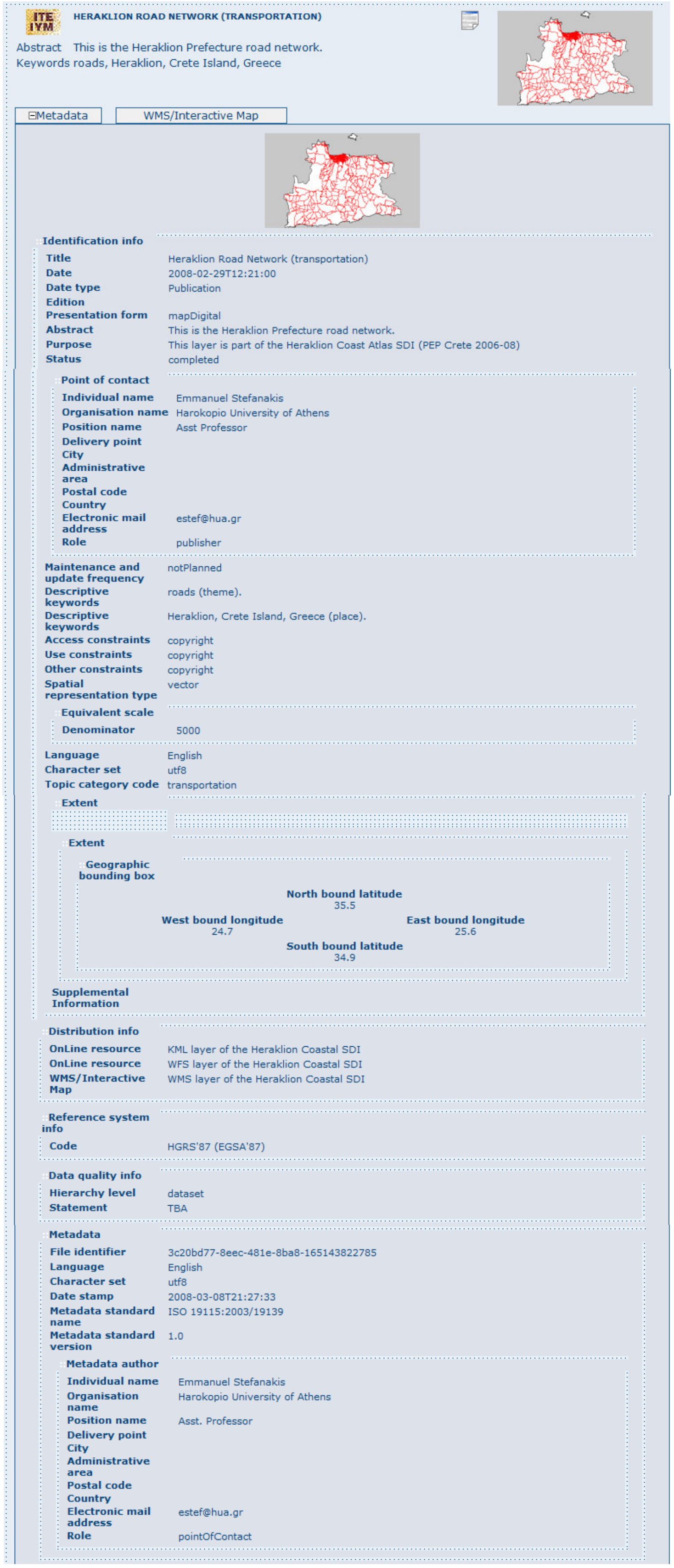


Figure 19 - The metadata items for the road network of the Heraklion SDI.

Figure 19 presents the metadata items for the road network ("odiko_irakliou" in Table 1 and in Figure 7) data layer as they are presented to the end-user. This metadata is encoded using an ISO19139 template (XML format). The metadata are organized into three main sessions. The first session includes the *identification* details, such as the metadata title, abstract, keywords, category, representation type, language, author and distributor. The second session includes the *spatio-temporal description*, such as the spatial reference system, the spatial extent, the temporal stamp and the interval in which the dataset is valid (not included in Figure 19; missed in public data). The third session includes *online resources*, such as links to the WMS, WFS, KML servers or the URL where the digital content/application is available. Note that the GeoNetwork opensourse server provides an efficient interface to access and browse the metadata, and to retrieve the geospatial content and/or applications, based on *thematic*, *spatial* and *temporal* predicates of the metadata items.

Finally, Figure 20 shows the customized query interface of the Heraklion SDI Catalog Server. End-users may enter:

a) the thematic predicates at the left side of the screen (e.g., a Keyword="Transportation") and/or at the right side of the screen (e.g., Category= "Images (Raster)");
b) the spatial predicate on the map frame at the middle of the screen (e.g., Spatial Query Window = [min_long, min_lat, max_long, max_lat], and Spatial Operator = "overlaps"); and/or
c) the temporal predicate at the top-right side of the screen (e.g., Temporal Interval = [2005–2007]).

Figure 20 - The query interface for the Heraklion SDI Catalog Server.

The last two predicates, i.e., the spatial and the temporal, are not important for the public layers, provided that they all overlap with the Prefecture of Heraklion and show up the current situation within the prefecture. However, they are of high significance to the Heraklion SDI. The temporal predicates will be in use when historical data sets are to be loaded in the future. The spatial predicates are also expected to play an important role - when the Heraklion SDI will constitute a node within the National SDI for Greece.

8.0 Conclusion

A regional *Spatial Data Infrastructure* (SDI) for the Heraklion prefecture has been developed using *Geographic Free and Open Source Software* (GeoFOSS), and special attention has been paid to the SDI features by hosting (Nebert, 2004):

a) rich geographic content (data and services);
b) sufficient description of this content (metadata);
c) effective methods to discover and evaluate this content (data catalogs);
d) tools to visualize the data (e.g., web mapping); and
e) services and software tools to support specific application domains.

The discussion above focuses on the SDI architecture, the components, the servers, the services, the interfaces and the functionality of the Heraklion SDI, and the reader may have access to the public content through the following URL:

http://heraklion-sdi.dynalias.net/coastatlas/index-en.html.

Plans for the future include enrichment and improvement of the current implementation, growth of the geospatial content accommodated and served, comparison and combination of this Open SDI with the parallel SDI under development using commercial software packages and the integration/interoperability of the SDI geospatial content with the National SDI of Greece currently under construction due to the INSPIRE initiative.

Acknowledgements

The authors wish to thank the General Secretariat for Research and Technology (GSRT) in Greece for partially funding this research.

References

Bernard, L., Georgiadou, Y., and Wytzisk, A., (Eds) (2005) - *First Research Workshop on Cross-learning on Spatial Data Infrastructures and Information Infrastructures*. ITC, The Netherlands.

CSW – Catalog Service Web, http://www.opengeospatial.org/standards/cat

Evans, J.D. (2003) - Geospatial Interoperability Reference Model. *FGDC, WG GAI*. http://gai.fgdc.gov/girm/v1.0/

ESRI - http://www.esri.com/

GE – Google Earth, http://earth.google.com/

GMaps – Google Maps, http://maps.google.com/

GML – Geography Markup Language, http://www.opengeospatial.org/standards/gml

INSPIRE - INfrastructure for SPatial InfoRmation in Europe, http://www.ec-gis.org/inspire/

ISO – International Organization for Standardization, http://www.iso.org/

ISO/TC211 - http://www.isotc211.org/

KML – Keyhole Markup Language, http://code.google.com/apis/kml/documentation/

Mitchell, T. (2005) - *Web Mapping*, USA,O'Reilly.

Nebert, D.D. (Ed.) (2004) - Developing Spatial Data Infrastructures: the SDI Cookbook. Version 2. GSDI Publications. http://www.gsdi.org/docs2004/Cookbook/cookbookV2.0.pdf

OGC – Open Geospatial Consortium, http://www.opengeospatial.org/

OSGeo – Open Source Geospatial Foundation, http://www.osgeo.org/

Stefanakis, E., and Prastacos, P. (2007) - Semantic-based spatial information infrastructures: integrating data and services into a single collection, in *Proceedings of the 23rd International Cartographic Conference*, Moscow, Russia, August.

Stefanakis, E., and Prastacos, P. (2008) - Development of a coastal SDI using GeoFOSS, in *Proceedings of the 11th AGILE Conference*, Girona Spain, May.

SVG – Scalable Vector Graphics, http://www.w3.org/Graphics/SVG/

Ticheler, J. (2007) - What are SDI, OpenSDI and GeoFOSS? *GeoNetwork OpenSource Community Website*, http://geonetwork-opensource.org/documentation/faq/foss-sdi-and-opensdi [accessed in November]

W3C – World Wide Web Consortium, http://www.w3.org/

WCS – Web Coverage Server, http://www.opengeospatial.org/standards/wcs

WFS – Web Feature Service, http://www.opengeospatial.org/standards/wfs

Williamson, I.P., Rajabifard, A., and Feeney, M.E.F, (Eds) (2004) - *Developing Spatial Data Infrastructures: From Concept to Reality*, Taylor & Francis

WMS – Web Map Server, http://www.opengeospatial.org/standards/wms

---