RAPID WEBGIS DEVELOPMENT FOR EMERGENCY MANAGEMENT

RAPID WEBGIS DEVELOPMENT FOR EMERGENCY MANAGEMENT

E Agosto [a], S. Dalmasso [a], P. Pasquali [a]

[a] ITHACA - Information Technology for Humanitarian Assistance, Cooperation and Action

Via P.C. Boggio 61, 10138 Torino, Italy - (eros.agosto, simone.dalmasso, paolo.pasquali)@ithaca.polito.it

Commission VI, WG VI/4

KEY WORDS: webgis, FOSS, rapid maping, emergency management

ABSTRACT:

The use of spatial data during emergency response and management helps to make faster and better decisions. Moreover spatial data should be as much updated as possible and easy to access.

To face the challenge of rapid and updated data sharing the most efficient solution is largely considered the use of internet where the field of web mapping is constantly evolving.

ITHACA (Information Technology for Humanitarian Assistance, Cooperation and Action) is a non profit association founded by Politecnico di Torino and SITI (Higher Institute for the Environmental Systems) as a joint project with the WFP (World Food Programme). The collaboration with the WFP drives some projects related to Early Warning Systems (i.e. flood and drought monitoring) and Early Impact Systems (e.g. rapid mapping and assessment through remote sensing systems).

The Web GIS team has built and is continuously improving a complex architecture based entirely on Open Source tools. This architecture is composed by three main areas: the database environment, the server side logic and the client side logic. Each of them is implemented respecting the MCV (Model Controller View) pattern which means the separation of the different logic layers (database interaction, business logic and presentation).

The MCV architecture allows to easily and fast build a Web GIS application for data viewing and exploration. In case of emergency data publication can be performed almost immediately as soon as data production is completed.

The server side system is based on Python language and Django web development framework, while the client side on OpenLayers, GeoExt and Ext.js that manage data retrieval and user interface. The MCV pattern applied to javascript allows to keep the interface generation and data retrieval logic separated from the general application configuration, thus the server side environment can take care of the generation of the configuration file. The web application building process is data driven and can be considered as a view of the current architecture composed by data and data interaction tools. Once completely automated, the Web GIS application building process can be performed directly by the final user, that can customize data layers and controls to interact with them.

1. INTRODUCTION

In the Information Age, Internet is the main source of almost any kind of data. The story of internet is recent and brief, about 20 years, during which it has evolved continuously and quickly. The information that it provided in the first years was flat, like pure text or documents. In the last 10 years the type of information shared by the Web has dramatically changed starting from that simple beginning. Now it is common experience to access videos, music and even maps by means of Internet. The concept of adding geographic information to information itself has spread on almost every site.

Maps are everywhere because the question “where does this come from?” has assumed the same importance as the “what”.

Georeferencing and managing spatial information is not the same as managing non spatial information because there is an increase in complexity due to the fact that geographic information is not absolute. Coordinates refer to a specific spatial reference system (SRS), and in the world there are thousands of SRS. Moreover many are the ways of sharing spatial information by means of specific standards. Whoever hosts spatial data must allow the gathering of data in different formats and different SRS, in a compliant way to common standards: geographic information managing is standardized by Open Geospatial Consortium (OGC).

The case of humanitarian emergencies is the core example of how the management of information and the usage of it are totally displaced. Whoever manages and keeps spatial data can be in a headquarter office whereas people in the field can use information without worrying about getting it physically. The only needed requirement is an internet connection, which is nowadays granted by satellite communication even if on the ground there is lack of infrastructures.

Information Technology for Humanitarian Assistance, Cooperation and Action (ITHACA) is a non profit association founded by the Politecnico di Torino and the Higher Institute for the Environmental Systems (SITI) born as a joint project with the World Food Programme (WFP) of United Nations (UN). This collaboration with WFP characterizes research projects in the field of humanitarian emergencies and the goal of the present paper.

One of the core aspects that determine an effective emergency response is data availability. This term means both immediate availability after the event and quality availability, having the most up to date and precise data.

To meet the challenge of rapid WebGIS development and data sharing in case of emergency, ITHACA focused its WebGIS project on the development of an architecture composed by Free and Open Source Software (FOSS) tools integrated and tuned to

allow the building of WebGIS and web applications in short time but maintaining an high level of customization.
The idea is to have a unique server architecture composed of data publishing and management systems able to output data in different ways based on the web HTTP protocol and respecting international standards.
Web applications should act as custom views on the architecture and web services should allow access to data regardless of the requesting source.
The last step of the development is the integration of the FOSS framework called GeoNode in order to increase the flexibility and the speed of data and metadata sharing.
To reach this goal it is important to plan accurately each step of the architecture, from the hardware part to the software component. Particular attention must be dedicated to the database preparation and to the server side technology.

2. TECHNOLOGY

The current ITHACA architecture is based on FOSS technology, starting from the operating system to the end user interface library. It is built upon the following components:

- Debian 6 operating system
- Apache web server
- Tomcat application servlet
- PostgreSql + PostGis as spatial database
- Geoserver + GeoWebCache as OGC compliant map engine
- Django + Python as server side application environment
- Ext.js, GeoExt and OpenLayers as client side user interface libraries
- GeoNetwork as metadata catalog
- GeoNode as data management and sharing portal

This architecture has been defined by choosing the components through the best available technology in order to avoid the user to install plugins or client side applications to access ITHACA data and web maps; on the other hand this architecture allows the integration with server side analysis tools like GRASS Gis or QGIS.
The high modularity of the components allows to respect the MCV (Model Controller View) development pattern that defines three main areas of logic:

- the model or database interaction, that is the interface through which the relational database is defined and queried
- the controller or the server side logic, which takes care of data preparation, application security, and URL matching
- the view or the template system, which receives data from the controller and renders the actual web pages or web scripts (javascript)

Django is a powerful tool that enhances and simplifies the server side MCV programming pattern, even if on the client side that is not as easy as on the server one.
ITHACA spent a big effort on the client side MCV architecture definition in order to speed up and automate as much as possible the WebGIS building process.
The concept is to transform the javascript classic procedural programming pattern by dividing the business logic from data retrieving and application configuration. The aim is to separate the static from the dynamic parts and allow rapid development by filling just the last part to have a working application.

GeoNode is a FOSS platform that facilitates the creation, sharing and collaborative use of geospatial data. The project aims to surpass existing Spatial Data Infrastructure (SDI) solutions by integrating robust social and cartographic tools.
Based on the same technologies above described, GeoNode acts as a common publishing environment where registered users can upload and share data as well as normal users can search for public data and use them to build their own web application, print maps or download data for their purpose.
At its core, the GeoNode has a stack based on GeoServer, Django and GeoExt that provides a platform for sophisticated web browser spatial visualization and analysis. Atop this stack, the project has built a map composer and viewer, tools for analysis and reporting tools.
GeoNode uses the same technologies as ITHACA uses and goes in the same direction: to automate the creation of web application to allow rapid mapping and data sharing adding tools to face emergencies.
GeoNode is a system which manage and publish spatial data using a distributed SDI in order to have data available to everyone, users can add their own data or use the available one, they can build a WebGIS with custom styles and can even share the chosen style. GeoNode integrates both Geoserver and GeoNetwork and thus can publish metadata on demand respecting different formats.

3. IMPLEMENTATION

The MCV pattern has driven the software architecture as well as the hardware one. In order to keep the data separated from the business logic and to avoid single points of failure, the design of the hardware architecture was:

- a Storage Area Network (SAN), which is dedicated to data hosting
- two twin web servers, one for development purpose and one for production, which are dedicated to data retrieval and serving.

Having the business logic separated from data allows easier SAN internal architecture.

The database structure is based on the FOSS Relational DataBase Management System (RDBMS) PostgreSQL and its spatial extension PostGis. Two separate instances of the database are present, one for each server, having each the data source stored on the SAN.
The logical database structure is composed by two locations:

- the development database, where all of the tables which are mirrored from ITHACA SDI are stored;
- the production database that hosts the tables which are needed by single projects and are optimized by means of spatial indexes.

Web servers have the same software structure as described in the technology chapter. The Tomcat application servlet hosts both Geoserver and Geonetwork components.
Geoserver is the data publisher. It is compliant to many OGC standards. Their implementations make it possible to serve data in both raster or vector formats. Each request to Geoserver can be performed via REST (Representational State Transfer) interface: this means that resources can be accessed by simply compiling a URL with correct parameters.

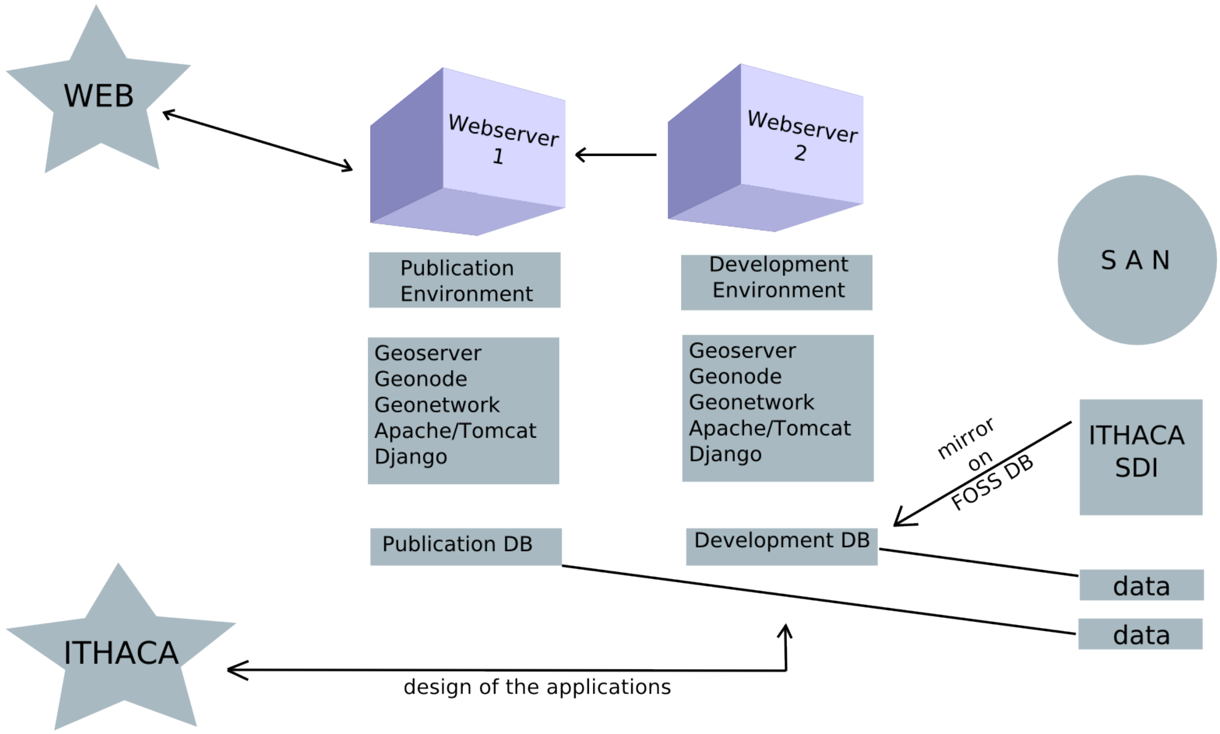

Figure 1. Framework schema

The installed version of Geoserver is the one that is distributed together with GeoNode. This version has a modified authentication method which is shared with the GeoNode authentication system in order to be accessible by GeoNode itself; beside it has an integrated module for advanced PDF printing.
The adopted version of Geonetwork is instead a standard distribution with the add-on of the simplified metadata schema used by GeoNode. The Apache installation manages Django folders both for the ITHACA application and for the GeoNode installation. The python code is executed by the WSGI module. By using Django libraries such as the model section, database interaction and data preparation for the web rendering are ensured.

4. WORKFLOW

To quickly respond to an emergency it is necessary to be prepared in advance as much as possible. This is the idea behind the explained architecture and it therefore means to fill the system with all the available and probably needed data.
ITHACA developed and mantains a Global Spatial Database Infrastructure (GSDI) with worldwide information. This infrastructure is built upon commercial tools and is mirrored on the FOSS part of the architecture (base on PostgreSQL/PostGis database).
Geoserver publishes the data by applying custom styles: this allows to have them always available in different raster or vector formats thus avoiding their duplication; besides the server side environment is programmed to allow rapid client side interface customization, by pre-defined common tools that are considered useful for a WebGIS to work.
Once ITHACA is requested to build a WebGIS, the main work is to configure the javascript client by choosing the map parameters, the desired layers and controls.
However the MCV approach to the javascript part has significantly simplified the task of the end user interface building. By separating the javascript logic from the configuration allows to just take care of choosing the content of the application instead of the tools. The configuration file is written using the JavaScript Object Notation (JSON), a list of key - values pairs with all dynamic parts of a web map. The JSON notation can be written and understood by almost all the programming languages, even python. Part of the ITHACA application configuration file, in fact, is written directly by the server side logic and this happens even in the GeoNode implementation.
GeoNode acts as a main data management gate and a metadata search tool. Exposing a simple web interface it allows users to search for data and metadata, to use them to build a simple web map and print it in a custom layout. For registered users it is possible to upload geographic data and complete them with metadata. GeoNode will automatically publish the data through Geoserver and the metadata through Geonetwork. Users can even comment uploaded data and metadata in order to stimulate effective publication.
GeoNode is thus an environment that works in parallel to the web applications and is dedicated to data and metadata management while web applications are dedicated to the data analysis and sharing through specific tools and custom interfaces.

5. APPLICATIONS

ITHACA adopts the previously described framework in all its web-mapping applications. Most of ITHACA research projects have the web as both input and output device.
Some projects are aimed at event preparation purposes.
ITHACA analyzes daily NASA MODIS satellite data to determine land area that are covered in snow in Hymalaian region. An automated procedure determines which part of the road network could have circulation problems and publishes it on the web. Users can print a custom pdf map of the area they

are visualizing on their screen, but also download data on their computers to perform further analysis or for they own purposes.

ITHACA has a running project aimed at flood events early warning. It exploits the NASA TRMM rain data to statistically compute critical rainy events. On specific areas (e.g. Bangladesh) also ground station precipitation data are considered.
The results of the analysis are mapped on a web application. Users can access the list of countries with an occuring critical event. Once a country is selected, the map zooms to the country extent and a marker is generated for each alerted basin of the country. The balloon which is connected to the marker is clickable and shows some information about the basin and the on-going events cvbsers can however access information about any basin or layer that is shown on the map.
Feeds from other sources (i.e. by GDACS - Global Disaster Alert and Coordination System) can be overlayed in order to have a comparison with the events the system highlights, offering more tools to have a better comprehension of what it is going on.

ITHACA's projects also involve post-event mapping: specific web applications are created for each emergency. The framework makes their rapid development easy.
On January, 12th 2010 Haiti was hit by a catastrophic earthquake which caused hundred and thousand dead.
The WFP, which has always been deployed in Port-au-Prince, has immediately started the post disaster response procedures, even many agencies spread around the world started producing any kind of analysis the day after the earthquake.
ITHACA, in collaboration with the WFP and the spatial agencies performed a damage assessment on high resolution satellite and aerial images by recognizing collapsed buildings, spontaneous camps and displaced people.
The WFP collected data from the field about food distribution points, food stocks, country operational bases, humanitarian corridors.
ITHACA was asked to build a web application for data sharing to allow both field and headquarter users to get the data as soon they were produced or updated.
Thanks to the MCV background architecture, the application was built in one day for the basic viewing capabilities; for the more advanced tools the entire development took three days.

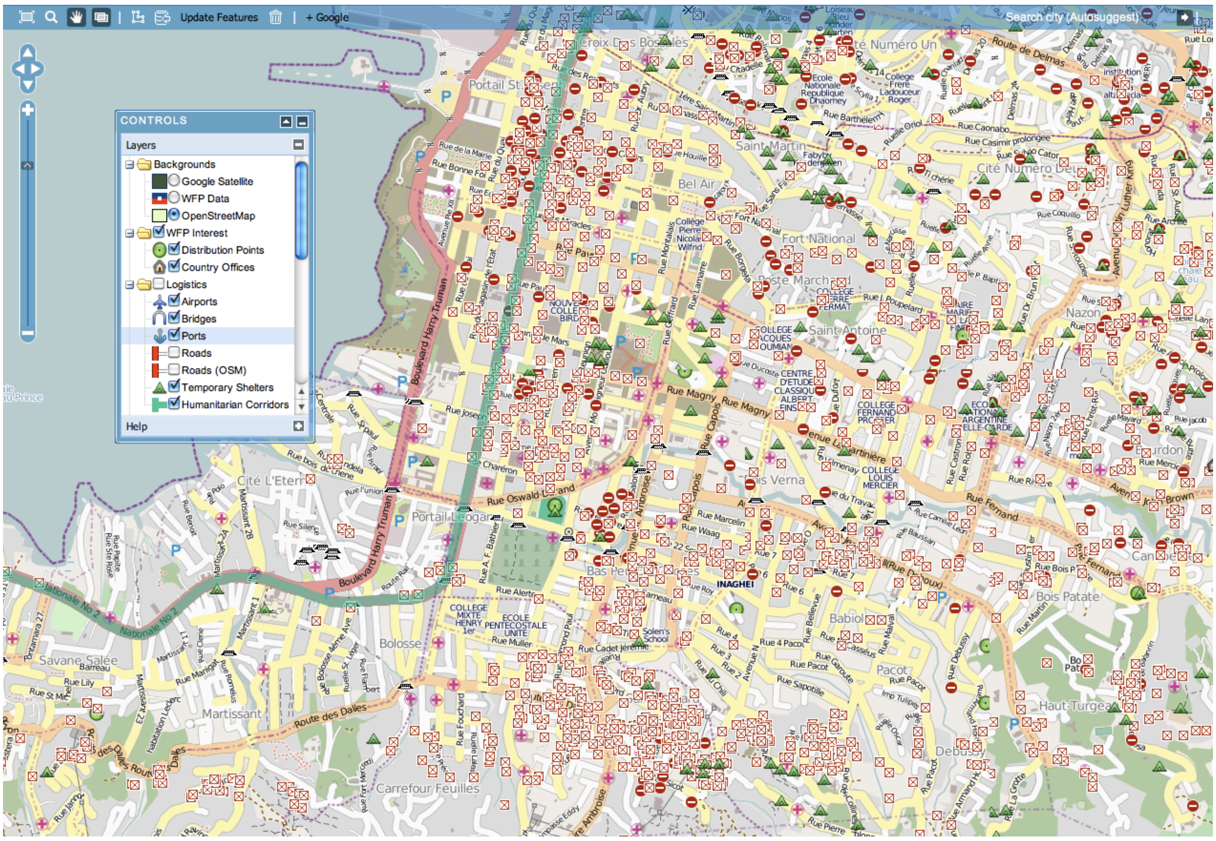

Figure 2. Haiti, view of Port au Prince with damage assessment

The system is a full map interface with toolbar and floating window for layer management. Displayed data are:

- ITHACA SDI or OpenStreetMap as base layers
- Distribution points
- Country offices
- Airports, ports, bridges, roads
- Temporary shelters
- humanitarian corridors
- damage assessment (collapsed buildings, damaged infrastructures, collapsed bridges, closed roads, restricted roads)
- population density grid
- custom user polygons

Application features allow to explore data through pan, zoom tools; a search field is locate on the right part this tool can be use to search a city or village and automatically zooms and re-centers the map on it, has an auto suggest function integrated which displays a list of possible matching names after typing at least four words.

The layers displayed in the floating window can be right clicked to set the opacity as well as zoom to the layer's extent.
In the toolbar is present an "add Google" button that dynamically adds Google among selectable base layers. This also dynamically loads Google javascript libraries. This solution was designed to avoid a massive javascript load when the page is accessed and have more features on demand.

An experimental feature is the possibility for a user to draw a polygon on the map and add it into the database. This can be used for social mapping purposes and stored data can be used to compare damage assessment with on the field contributions. The controls for this feature are located in the middle of the toolbar and are: "draw/edit polygon", "load existing data from the database", "commit the feature to the database" or "delete feature".

5.1 Geonode

GeoNode uses the GeoExplorer javascript application as map viewer. It is based on the GeoExt library and it is a powerful tool to view, export, save and print maps with customized layers and styles. Layers can be chosen by a list of default servers or from a custom source; the rendering style of vector data can be managed by the simple GeoExt Styler interface.

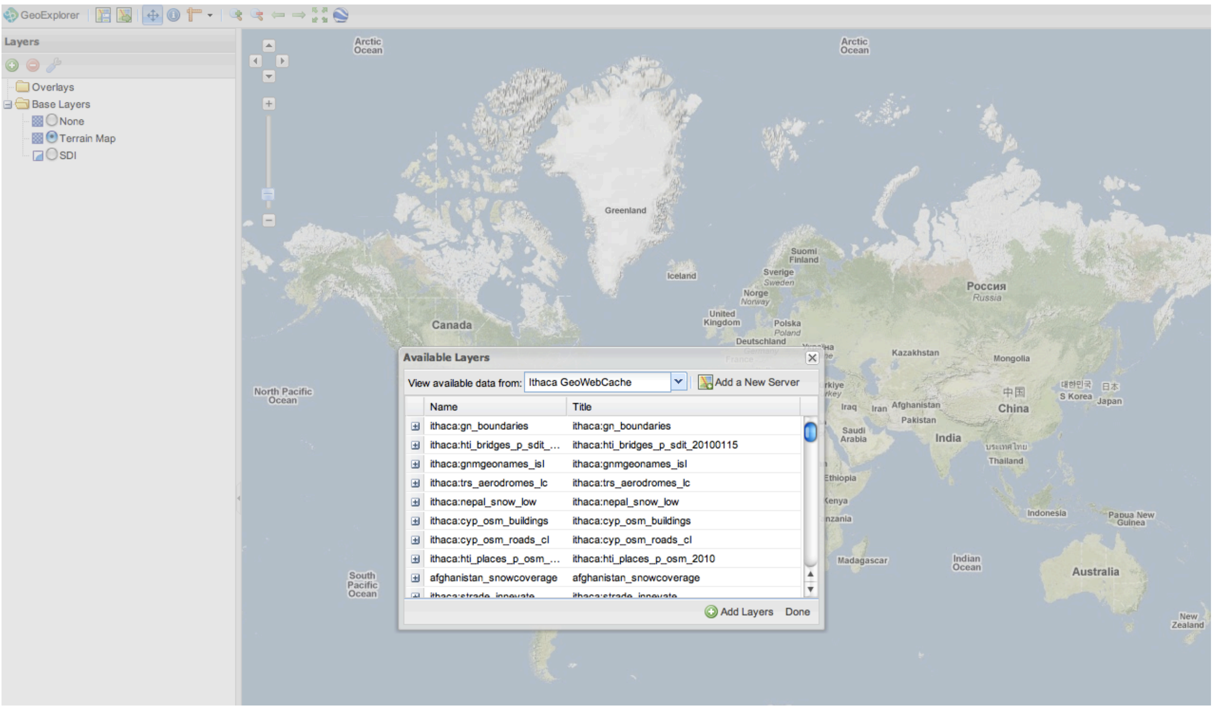

Figure 3. GeoExplorer view with layer adding window from Ithaca server

Main features are:

- Make it extremely simple to share data
- Provide user statistics
- Easily add comments, ratings, tags
- Allow collaborative filtering
- Provide rankings of best 'views' and data sets contributed—such as highest rated, most viewed, most shared
- Allow connectivity between several GeoNode instances to augment the collaborative potential of government GIS programs
- Simple installation and distribution
- Automatic metadata creation
- Versioned metadata
- Search via catalogues and search engines (Google)

GeoNode can be the platform where to integrate the experience gained during year of cooperation with the WFP.

Major features to be added are:

- Exporting capabilities (kml, shapefile)
- Printing capabilities (in custom and professional layout)
- Layout customization
- Routing capabilities

GeoNode APIs should be extended both for the server and client side keeping the modular approach and respecting the MCV pattern which has lead the entire development of the project.

6. CONCLUSIONS

The framework is completely made up of FOSS components in a way that both technical (standard compliance, freedom to develop and test, presence of needed functionalities) and economical (no license costs) reasons are taken into account.
The adoption of a MCV python based development framework makes standardization of ITHACA web applications easier: the presence of a unique folder structure, of a unique planning,

makes it easier for programmers to intervene, maintain and develop the code and the functionalities of an application.
The developed framework was successful in speed ITHACA web mapping capabilities up. This was useful in Haiti emergency. The adopted FOSS tools are the same that the starting GeoNode FOSS project chose as its component: this shows a spread shared vision about the web mapping issue.

7. FUTURE DEVELOPMENTS

Advanced tests about the impact of Geonode on the previous framework will be evaluated.

The framework itself needs the add on of standardized default analysis capabilities. First future goal is to define the default set, whereas following steps will be the development of the single tools and their graphical interface and the integration in the system.
OGC Web Processing Service (WPS) will be taken into consideration by adopting PyWPS (a python implementation of WPS). As an example a tool to compute the population which is present in a user defined area was developed: the area is drawn by OpenLayers drawing editing tools, whereas the computation is made by GRASS GIS which is installed on the server. LandScan^{TM} Dataset is adopted: it is a worldwide population database compiled on a 30" X 30" latitude/longitude grid, which has been developed as part of the Oak Ridge National Laboratory (ORNL) Global Population Project for estimating ambient populations at risk. What it still to be developed is the standardization of these kind of tools in the framework.
Another future work will be the design of specific data base structure for non standard GIS data such as georeferred non rectified photos and videos. ITHACA is developing a project for low cost mobile mapping systems, that was tested during emergencies.

References from Literature:

Agosto E., Dalmasso S., Paquali P. (2009) “Ithaca world-wide flood alert system: the web framework, Applied Geomatics, DOI: 10.1007/s12518-010-0041-x

Albanese A. , F. Giorgi, N.P. Premachandra, O. Terzo, R. Vigna (2009). “Application of an early warning system for heavy rainfall.” In: Proceedings of the Second International Conference on Earth Observation for Global Changes (EOGC 2009). Chengdu, China, 25-29 May 2009.

Agosto E., Dalmasso S. (2009). “Ithaca web services on a FOSS framework”, In: Cartography and Geoinformatics for Early Warning and Emergency Management - C4C Symposium, Praga

Agosto E., Dalmasso S. (2008). “Erogazione di web-services per la fruzione di un geodatabase a scala globale ”, In: Conferenza nazionale ASITA, L’Aquila

Albanese A., Disabato F., Terzo O., Vigna R., Giardino M., Perotti L. (2008). A preliminary approach to flood risk mapping and flood forecasting system In: LDCs. XXI ISPRS Congress, Pechino

Ajmar A., Perez F., Terzo O., (2008). “WFP spatial data infrastructure (SDI) implementation in support of emergency management”, In: XXI congress of the International Society for Photogrammetry and Remote Sensing.

Agosto E., Disabato F., Dalmasso S. (2007). “Servizi web per la gestione di emergenze ambientali”, In: Conferenza nazionale ASITA, Torino

Bendea H., Boccardo P., Dequal S., Disabato F., Giulio Tonolo F., Marenchino D., Perez F. (2007), “ITHACA project: a new approach for disaster management”, In: Joint CIG/ISPRS Conference on Geomatics for DIsaster and Risk Management, Joint CIG/ISPRS Conference on Geomatics for DIsaster and Risk Management, Toronto

References from websites:

Django, http://www.djangoproject.com/

DRY, http://c2.com/cgi/wiki?DontRepeatYourself

ExtJS, http://www.extjs.com/

GeoExt, http://www.geoext.org/

Geonode, http://geonode.org

GeoServer, http://geoserver.org/

GeoWebCache, http://geowebcache.org/

OGR, http://www.gdal.org/ogr/

OpenGIS(R) Web Processing Service document (OGC 05-007r7) v 1.0.0, http://www.opengeospatial.org/standards/wps

LandScan, http://www.ornl.gov/sci/landscan/

OpenLayers, http://openlayers.org/

PostGIS, http://postgis.refractions.net/

PostgreSQL, http://www.postgresql.org/

Python, http://www.python.it/

PyWPS, http://pywps.wald.intevation.org/

TRMM, http://trmm.gsfc.nasa.gov/

Groot R., McLaughlin J., (2000). Geospatial Data Infrastructure: Concepts, Cases and Good Practice. New York: Oxford University Press.