Web 2.0 brings GIS to the World Wide Web

Dirk Frigne

DFC Software Engineering

Email: dirk@dfc.be

Direct line: +32 9 236 54 12

Brugsesteenweg 587, 9030 Gent, Belgium

Introduction

In the world of Geo-ICT the original data sources play a vital role. Mutual exchanging and adapting information between these data sources are one of the cornerstones of what one call the "Spatial Data Infrastructure" (SDI), or in other words the geographical information highway.

Who says data sources thinks of databases. One of the first attempts to manage these geographical data easily within the Flemish government consisted to set up a unique database model where all the geographical data could find their places. This central treatment works as far as it is sufficient to make the data available only within the own organization. However, this way of working has also its restrictions.

One of those restrictions is that collecting and centralizing these data are only one part. Keep them up to date and ensure that several users can use them in an efficient manner is a complete other task. When these data can come also from outside the organization and have to be delivered to third parties, also outside the organization, these restrictions are of such nature that a solution on the basis of a central architecture is practical not feasible.

The geGIS project, started up by the Flemish government in 2006, was an initiative to offer a solution for the above-mentioned problem. This paper describes the technical architecture of geGIS and explains, by means of a proven project "in the field", why it can be used as best practice for a cross-border SDI node. geGIS is developed as an open source project. Up on today there are on a regularly basis user conferences, with more and more users using the system in the real world. (more information about the project can be found on http://www.gegis.org)

Context

A spatial data infrastructure (SDI) is a software stack which allows bringing geographical information in a simple manner from the producers to the users. One can define it as the necessary network of ways and highways to bring goods and services from the authentic source to the destination.

There is an enormous demand to such infrastructure. A lot of data processing requests contain a “where it is” element and geographical information contains a lot of attributes. Many users need the same data, possibly in another format. What for one party is source data means for another party reference data.

A (ancient) manner to set up a spatial data infrastructure consisted in collecting all data centrally, making a meta-index (catalog) so that users get insight in which data is available and defining a way to distribute the data (in practice this means mailing shape files or CD's). An important disadvantage to this manner of work was the fact that the time between bringing certain data up-to-date and the delivery of these adapted data was much too long for the users. Furthermore a complete organization around version management is needed to be able to manage these adaptations in a correct manner.

We live today in a world of web services and fast network connections. A modern SDI generally rely on these web services. A second important element exists in the use of open standards, so that data exchange occurs in a controlled and especially standardized way.

Within the GIS world we are speaking of OGC standards. These exist already longer than the ICT standards around web services and are unfortunately not entirely compatible. Initiatives are, however, undertaken to incorporate these OGC standards within the ICT standards, but this process will take anyway a couple of years.

Authentic sources[1]

Geographical data can be split up in several layers, which can be combined with each other. Each layer, or a subset of this layer can be assigned to an owner or an authority which is responsible for these data. We call such an entity the authentic source of this data layer.

These sources must be stored in an efficient manner and it is at this point that database technology is needed.

Within the Ministry of the Flemish Community the political sectors RWO, LNE and MOW have collected all relevant geographical data in a central ORACLE 10g database server. The data on this server are on the one hand updated by several fat client GIS applications. This infrastructure is known as the `Mercator environment' and was set up from the ex-LIN cell (environment and infrastructure).

[1] http://www2.vlaanderen.be/e-government/english/authentic-sources/index.html

The disadvantage of the Mercator is the need to install for each application an expensive web mapping server and the difficulty to anticipate on the rapidly modifying needs. This system works for all “internal” customers. However, more and more data must be exchanged with “external” customers. This exchange is in two ways: own data has to be made available, external data must be integrated.

In 2005 the current political department RWO of the Flemish Community sent out a tender to consult and manage geographical information in an efficient manner. The most important criteria were:

- low license costs per web mapping application (electronic window)
- to be able to modify data by means of a standard browser
- based on web technology
- a generic system which is also useful by other administrations
- a look and feel of a desktop application
- covered by SDI principles

The winning proposal was an architecture based on open source components, where the missing links were developed and released also as open source components. The project received the work name geGIS and was carried out by DFC software engineering as the main contractor.

The objective of geGIS 1.2 is not to be an alternative for desktop GIS applications, but rather a supplement. As geGIS contains simple editing functionalities a lot interesting applications become possible.

The requirement to be build a generic system which is also useful by other administrations enabled the idea to establish a user group. Actually this group meet each other 4 times a year. In the meantime this user group has been enlarged to about 15 administrations, local governments, utility companies and private companies. Gradually the idea grows to start with a next version. This version will have more GIS desktop functionalities in a browser environment such as snapping, split up and adding plots, producing complex objects etc…

geGIS architecture

The geGIS software stack (Figure 1: geGIS 2.0 tiers) consists of 3 different layers.

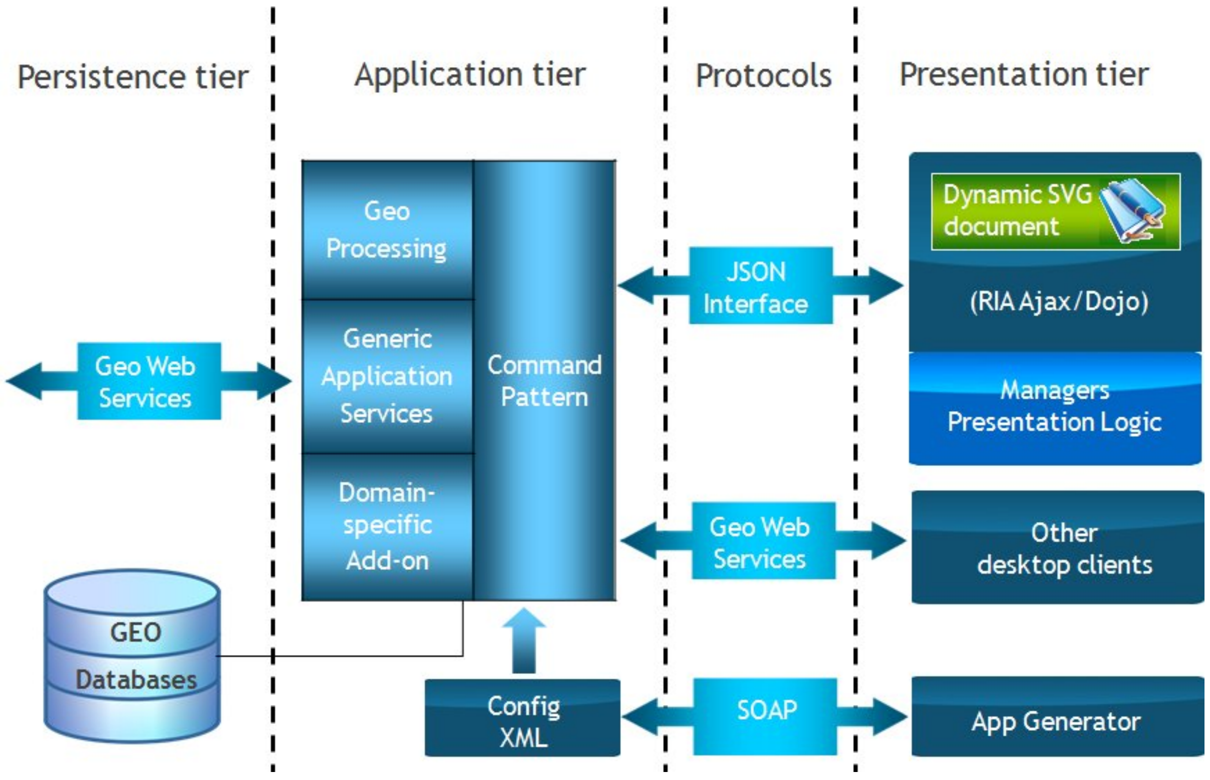

Figure 1: geGIS 2.0 tiers

The presentation layer

This layer has been developed entirely on the basis of AJAX and is situated in the browser. At starting up a geGIS application these components are first downloaded, started up and afterwards the geographical data are picked up.

The presentation layer has been built using Geomajas WebApps. This open source framework makes building Web maps applications very easy. Geomajas have been built using the Dojo toolkit. Because of this a good design and an object oriented treatment becomes possible.

The application layer

The application layer is responsible for converting and integrating the geographical information from the different authentic sources to a format that can be handled in the Geomajas WebApps environment. We use SVG or VML technology so that the geographic data can be used in the different actual modern browsers. Thanks to this vector format we are able to present and manage geographic data in a consistent way from the user side to the storage. The big advantage of this technology is indeed that editing of geographical and other information are made possible within the browser and that this information can be processed further in the backend. (and this is why end to end solutions become possible).

The persistence layer

To store the data and read them again, the Hibernate framework is used. Data are not only read in, but also stored by means of this framework.

An important concept is also reading and writing data by means of the OGC standards. Because of this aspect, geGIS can act as an editable SDI node. We use the geoServer framework to implement the OGC standards. GeoServer is a reference implementation for the open GIS standards “web map server 1.1.1 – WMS” and the “web Feature server 1.0 and 1.1 – WFS”. The combination of the far-reaching editing possibilities of the Geomajas framework and the capability to work with open standards using the geoServer framework makes from geGIS an interesting architecture to set up editable SDI nodes.

The front-end architecture – The presentation layer

At the front-end a rich client application (RIA) has been built on the basis of Geomajas Webapps. Geomajas stands for Mapping with Asynchronous Java script And SVG. The DOJO framework is used as underlying application framework. Geomajas Webapps are based on specific GIS related DOJO widgets that we created to ease the construction of Web Based applications. By combining and configuring these widgets within an html page, one can compose rapidly and efficiently a powerful GIS application. The most important widgets are:

MAP widget: this can be used alone or in combination with other map widgets. Each map widget can be used to present several data layers and to produce or to modify the geographical attributes of the presented data.

Legend widget: this widget gives extra information on the presentation of the layers. Because of this the user can get an insight in the meaning of the presented information. The widget can be established so that only the visible layers for example are presented.

LayerTree Widget: this widget ensures an user-friendly control of the different layers. Users can decide which layers they want to turn on or off, if they want to visualize some attributes and if they want to activate the snapping mode of a specific layer.

FeatureTable widget: This allows visualizing, selecting or modifying the alphanumeric data related to the objects in the layers.

The purpose of the widget architecture is to provide a mechanism for web designers, so they can integrate GIS behavior in their websites. Configuration is used to provide the widgets with the appropriate behavior.

Figure 2: Geomajas Webapps architecture gives some insight in the way events are handled in this widget based architecture. We get also an idea how the interactivity is being stimulated in this web based application environment.

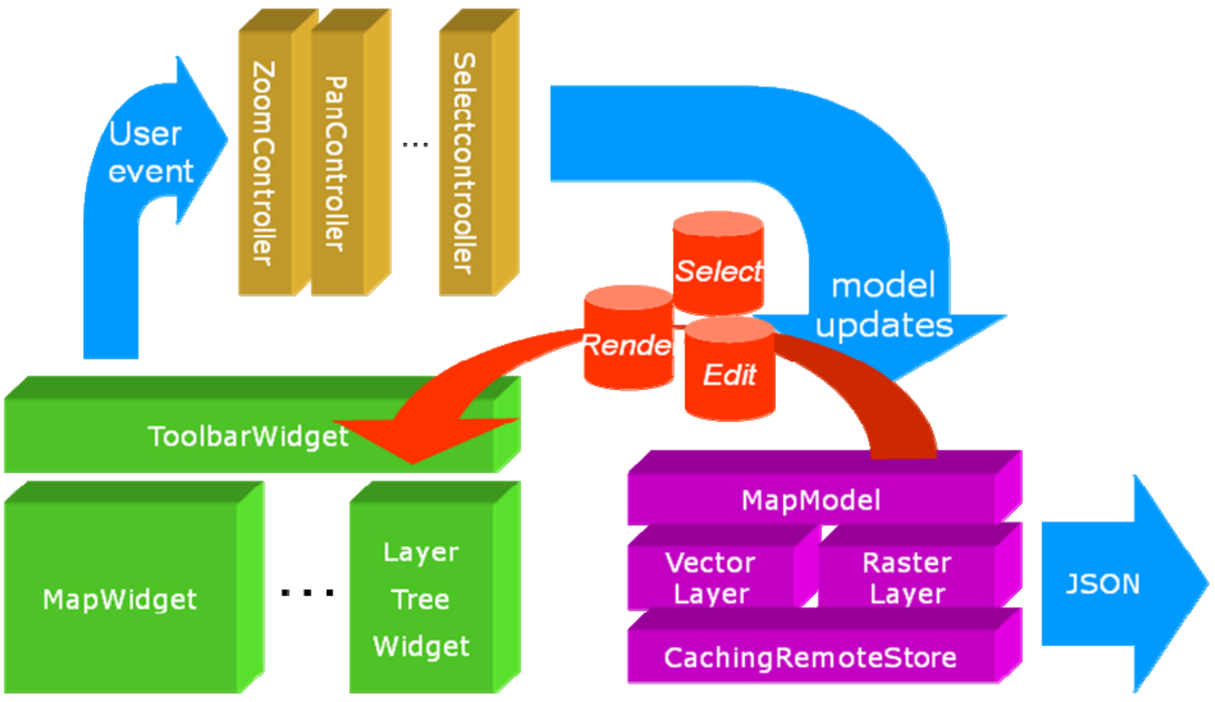

Figure 2: Geomajas Webapps architecture

When an event is captured (mouse movement, button pressed, scroll wheel movement, …), the event is passed to the relevant controller. The main task of the controller is the translation of the event to the action requested to update the model.

The model (in purple colors) is managed completely on the client side of the Geomajas environment. It is the model that can decide on its own force to re-render the impacted widgets (green colors) or to get first more information from the server.

All the communication between the widgets is orchestrated by the principle of 'DOJO' topics. These topics are a sort of communication pathways for specific events. The rendering of a certain map widget can be accomplished by attaching the model, responsible for the content of the map to the render widget. The map widget is always listening to the render topic, so when this topic is notified by render events, the map widget can take the appropriate action to start to render itself by activating the right painters.

If necessary, the model can decide to go for a request to the server, because there is need to new, not yet cashed data. The decision for the communication between the client and the server was taken in favor of JSON. This because JSON is a appropriate light weight protocol, very well suited to handle at high speed the data requests between the two environments. Because the Geomajas framework is an open source framework, and we control both the client and the backend side, we can optimize on a proprietary protocol between client and server, optimizing on speed and functionality, without giving up on openness (open standards, open architecture).

After 2 years of evolution, we are satisfied and happy with the choice for the DOJO framework. This choice has the advantage that we can concentrate our efforts (of our restricted rescources) on the development of the typical GIS and mapping problems, where DOJO is evolving to a mature OO based software framework to develop web based AJAX applications.

the application generator

An important not functional requirement for the geGIS architecture is the possibility to set up the platform by configuration. In the first version (geGIS 1.2) there exists a restricted user interface to couple authentic data sources and to convert shape files to a database table, whereby the geographical information per record is stored in a "geom" object.

Most of parameters with respect to the application generator must be established however by filling in a XML file by hand. Fortunately there is a validation tool so that errors can be avoided as much as possible. However, to this component there is still some work to do.

By extending the user interface of the configuration tool it will be possible to help the user in making the correct choices to fill in the right XML parameters. In this way, the threshold to use this software reduces enormously. Since it is open source we expect that building a more user friendly interface will take a prior position in the wish list of that software.

The configuration tool in the current version allows one to describe which layers must become visible, from which scale and what you can do with these layers. Depending on your role you can adapt layers by defining new objects, you can snap on objects of a layer to create other objects, or you can simple consult information and ask their attributes. The details of all parameters are described in the documentation by means of a XML diagram.

The aim is to cooperate with another OSS project to share the configuration. Desktop tools have the same needs to set up layers, styles and other parameters. We think at projects like UDIG and gvSIG. The idea is to use such a software as a local configuration tool, where the configuration can be set up and tested before the parameters are used to distribute the maps via the geGIS server. The idea of the 'Author' of the AutoDesk MapGuide software.

geGIS as SDI node

The fact that the application layer uses both Hibernate to read in and write data and geoServer to forward data on the basis of the OGC standard to other systems, makes this architecture extremely useful to act as Spatial Data infrastructure (SDI) node. (see Figure 3: geGIS as SDI node).

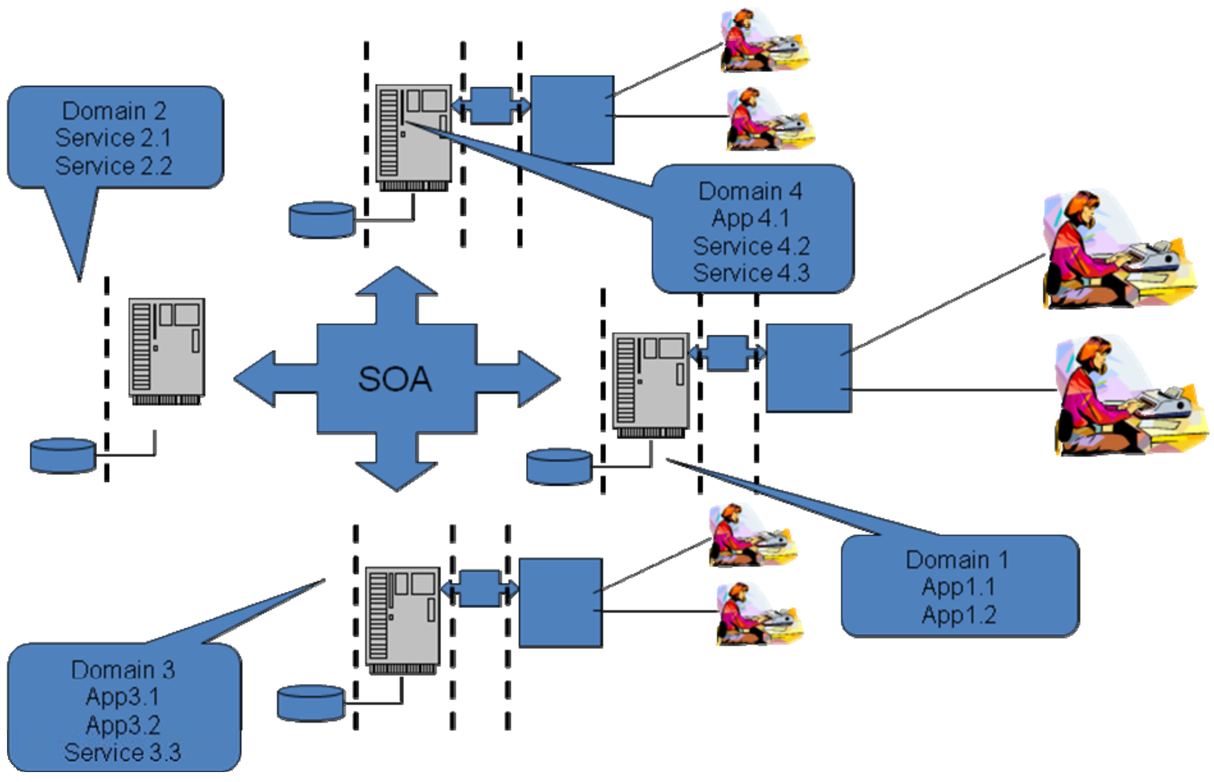

Figure 3: geGIS as SDI node

By means of the next realistic example we will describe the geGIS infrastructure such as it is implemented in Belgium by the Flemish department RWO:

The infrastructure is a combination of different domains working together, defining different applications, each with their own goal.

The free space is very limited in Flanders and before to offer new to be developed land parcels, government wants' to use all the possible very distributed parcels they can find. Domain 1 presents the field of the Unbuilt-up plots (the ROP server - register of Unbuilt-up plots). This is a server, keeping track where there are plots available to build on. The situation in Flanders is this way that there exists still hundred thousands of available Unbuilt-up plots, but they are split up and spread in the Flemish landscape that it is not obvious to trace them. This application server exactly has the task to register and manage these kinds of plots.

Application 1 is the main application for the register of Unbuilt-up plots, application 2 is an application to manage the saved information, and to deploy this information to the public.

We could now keep up all relevant data also under the domain 1. This is/was undoubtedly the way it is done today within a lot of applications. Well, it is exactly here that geGIS goes a couple of steps further.In the local geographic database only those relevant data which have to do with the Unbuilt-up plots are saved. One opted also to keep all the plots and their related information in the same database. The reason why is simple because at that moment of the project there was no SDI available.

A first service is indicated in the figure as domain 2, service 2.1. This is an ordinary ICT SOAP service to be able to navigate to a certain address and house number. The SOAP request is handled by the server and presented to the user with an appropriate user interface. The complete address navigation system can be activated or adapted by means of a configuration parameter so that one can only search via the name of the street or municipality instead of the house number.

Finally 2 other geGIS servers have been set up (domain 3 and domain 4) respectively a plan register and a server with district plan information. By means of the application 4.1 a municipal civil servant can update the plan register of its municipality. This is possible also if he has a local GIS application by addressing service 4.2. The ROP server will address also the 4.3 services to examine if a certain plot lies in an area where one can build or not. The same scenario happens with the district plan server, but then at the level of the Flemish district.

All this information is consolidated within the application layer of the ROP server so that it is possible to decide definitely if a plot where no house is constructed, is indeed an Unbuilt-up plot or not.

This example shows clearly how information from several authentic data sources can be combined and integrated into a new application with own specific application logic.

geGIS 2.0

Today several projects have been started up on the basis of geGIS. Several administrations set up pilot projects to be able to exchange data via a SDI solution.

4 times per year there is also a user meeting for geGIS users or people who are interested to get to work with it.A subject which is very hot today is the start of the geGIS 2.0 project. One of the objectives of this project is to make a functional description of the software stack which is needed to build a SDI node. The main goal is to develop geGIS 2.0 as the reference platform to create editable SDI nodes. The possibilities of integrating OGC servers with database servers in a domain driven application environment where the domain logic of a SDI node can be managed in a user-friendly way by means of a browser application, result in interesting opportunities to develop intelligent and maintainable SDI nodes. The fact that people can maintain their data, close to their real business needs, will guarantee them maintainable databases. Further information can be obtained from the internet sites of the two projects.

geGIS 2.0 Software stack

There is a first release for geGIS 2.0 as a sample of the Geomajas 1.3 software. This is just a sample how the userinterface could behave and what can be done by the Geomajas framework. The colors in the Figure 4: geGIS 2.0 architecture give an indication how the different opens dource projects can work together to create an editable SDI node. It should be interesting to organize a sort

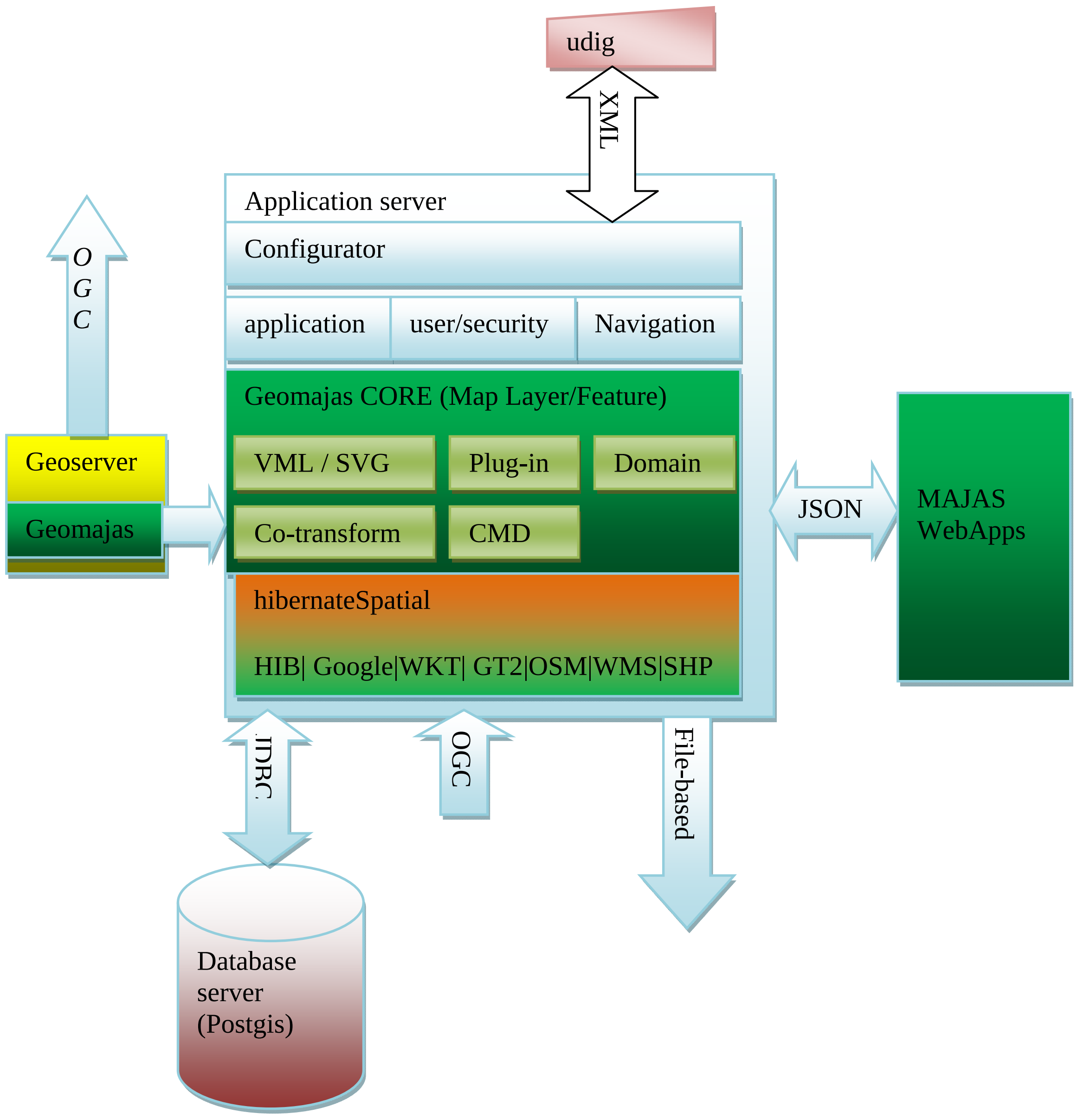

Figure 4: geGIS 2.0 architecture

of code sprint to establish a proof of concept of this idea.

The green part is the part filled in by Geomajas. Geomajas is responsible for the end to end handling of the data, and the translation from GML to more appropriate formats that can be used on the browser level.

To establish the persistence we use hibernate spatial (the red part of the drawing). Hibernate spatial is based on the hibernate framework and is extended to handle specific geographic formats. Most of the common geographic formats can be handled in this way. Hibernate can handle all sorts of datastorage, like files and databases. To store the authentic data that belongs to aa certain SDI node, we prefer to use Postgis as a geographic database. (The purple part).

An important part of the architecture is the possibility to support the OGC and other open standards.This is the reason to integrate Geoserver (the yellow part). Geoserver is used to upgrade the SDI node to a WMS/WFS enabled server. (for reading these formats, the hibernate layer is sufficient, but to generate these data, we use Geoserver).

At last, we want to integrate and reuse the configuration possibilities of UDIG, to configure the mapping needs for geGIS.

All the blue rectangles are the features belonging to the geGIS project and should be designed. Some parts are already in a prototype stadium in the previous release of geGIS, such as the navigation module and the configurator. For the other parts we should look if there are projects we can integrate, like the application paradigm, the user/security module etc…

The aim of this document is to establish the idea of an editable SDI node system, that can easily be deployed and extended by interested parties. We will extend this document with shared idea’s and the results of the technical discussions we organize on regular basis.

About the author

Dirk Frigne has more than 20 years experience in informatics. His thorough know-how built as a developer, architect, project manager and business manager forms the basis for a broad vision on the ICT developments where we are faced today and the new recent developments which outline themselves to the future.

As founder of DFC Software Engineering Dirk is already more than 15 years commit with that part of the data processing which is known as geo-ICT. Of the problems around piping management by means of CAD systems to managing and exchanging valid sources of information concerning SDI needs, he has a founded opinion which is based projects tested “in the field”.

Since 2 years he is working on his idea of an editable SDI node. Therefore he engaged some people to work on the first version of Geomajas, which is he basic technology for the editable possibilities of the framework. Know he wants to create a prototype for the geGIS 2.0 platform, a platform that can be the basis for sharing and integrating geographic information in a step by step distributed environment.