CO-403

E-FOREST: A GENERIC PLATFORM TO BUILD THEMATIC GEOPORTALS

COUSIN J.L., PESTY B.

Inventaire Forestier National, NOGENT SUR VERNISSON, FRANCE

1 CONTEXT AND OBJECTIVES

The INSPIRE Directive (Infrastructure for spatial information in Europe) requires the European Commission to establish a Community geoportal and Member States to provide access to their infrastructures. Consequently, many geoportals have fulfilled these requirements by proposing different types of dissemination (global dissemination at national level, Legally Mandated Organisation (LMO) dissemination at national level, Spatial Data Interest Communities (SDIC) for thematic dissemination with different data providers, etc.).

These Spatial Data Infrastructures (SDI) have arisen thanks to the interoperability of the metadata, data, networks and services. This interoperability - mainly based on the ISO/TC211 and OGC (Open Geospatial Consortium) standards - is designed for geographical data. But what happens for complex geo-referenced observations?

In 2008 the Joint Research Centre of the European Commission launched a call for tender called the "Framework contract for the provision of forest data and services in support to the European Forest Data Centre". A consortium composed of several National Forest Inventories (NFIs) in Europe and led by the French National Forest Inventory won the contract. The first step in the project was to build the information system for the NFIs at European level as a basis for providing services to the European Commission. The consortium had to find a technical solution to build a geoportal able to deal with structural geo-referenced observations.

Unfortunately, the situation is complicated when the geoportal has to manage geo-referenced observations in order to describe natural phenomena such as forests or soil conditions. In such cases, it is frequent to have relationships between the different entities (tables) with "one to many" cardinality. For example, a forest plot describes forest stands composed of trees that can be split into different elements such as trunk and branches for which measurements are made. This kind of complex data is naturally represented in a hierarchical structure, which can be done in a database. Handling such data require understanding the relationship between the entities.

Currently, common geoportals do not manage these structured data. Well known standards such as WFS (Web Feature Service) cannot easily be used to exchange these data because the feature (geometry) is in relationship with different entities and not only a few attributes (columns). Moreover, geoportals cannot provide sophisticated tools to query and analyse data: it is only possible to retrieve the attributes linked to the selected feature.

Another aspect also has to be taken into account: the data providers. In our context (forest inventory, biodiversity observation), some data providers cannot provide complex web services on the Internet because they do not have a sophisticated enough information system. Therefore, the proposed system must be as simple as possible to facilitate data gathering.

Consequently, we studied different standards in order to propose a solution that takes into account the constraints (keeping the submission procedure as simple as possible), the required functionalities (metadata description, data gathering, data query) and the context: forest inventories (from raw data to statistical results) and possibly biodiversity protocols as well.

The SDMX (Statistical Data and Metadata eXchange) is an initiative to foster standards for the exchange of statistical information lead by major organisations (EUROSTAT, the United Nations…). The SDMX provides standard formats for data and metadata, together with content guidelines and an IT architecture to exchange data and metadata. This protocol is very well suited to exchange statistical results (statistical tables) but not raw data. This solution was therefore not retained.

The TDWG (Taxonomic Databases Working Group) is developing standards for the exchange of biological/biodiversity data. In this case, a dataset is modelled as a graph of identifiable objects. Objects are defined by an ontology understandable by both humans and computers. Globally unique identifiers are used to link objects across the network and a common exchange protocol is available to search for and retrieve data. This protocol is called TAPIR (TDWG Access Protocol for Information Retrieval). This standard could have been retained but the process for modelling each new dataset was too costly for data providers.

We therefore propose to develop an entirely new web-based platform, called E-forest, capable of managing unknown datasets. The only assumption we can make on a given dataset is that some information about its location will be available.

The main functionalities of this platform are data description, data gathering, data querying, data treatment and data dissemination. The particularity of this project is to provide a generic system based on metadata descriptions.

The purpose of this article is to present this E-forest platform from a technical and user point of view.

2 APPROACH AND METHODS

2.1 Metadata and standards

"Metadata" can be defined as data about data. Metadata describes how, when and by whom a particular dataset was collected. It also describes how the data is formatted. Metadata is essential for understanding information stored in a database.

When we speak about metadata, one of the first reflexes is to think about Dublin Core.

The Dublin Core Metadata Initiative popularized the idea of "core metadata" for simple and generic resource descriptions. Using only 15 base text fields, a Dublin Core metadata record can describe physical resources such as books, images, etc.

In our case, the Dublin Core concept can be applied but we need to be more precise for the data contained in the dataset. As none of the data we need to manipulate are known beforehand, the system must be able to describe them.

The ISO19115 standard, which is the standard for geographical metadata, could also be useful in our context.

Three levels are commonly distinguished in the ISO19115:

-Discovery metadata. What datasets am I interested in? This question is usually answered by answering three questions: what, where and when?

- Exploration metadata. Do the identified datasets contain sufficient information and are they pertinent for my own purposes?

- Exploitation metadata. The purpose of this level is to provide information in order to compute the data for further analysis. At this step, if an attribute linked to a geometry corresponds to a list of codes, this level provides information about the code value and its signification.

In the framework of E-forest, the ISO19115 standard does not fit the need very well. On one hand, it requires certain mandatory fields which are not necessary in our context. On the other hand, it would have to be extended with new information in order to provide information about the location.

Consequently, we have only kept the Dublin Core concept and have extended its implementation keeping in mind that the system has to be easy to configure.

2.2 Implementation of metadata

The metadata is the core of the E-forest platform. It is used to describe the type of data manipulated by the E-forest platform.

The metadata database provides a structure to store dictionaries of data used in the system. These dictionaries are called "logical descriptions of the data".

As the E-forest platform does not know which data to deal with, the metadata database also contains information about its physical location. Where are the data stored? How can the data be queried? What are the relationships between one set of data and another? All this information is called the "physical description of the data".

For each function of the application, the system starts by reading the description of the expected dataset and then generates the corresponding code.

Configuring a new dataset is then only a matter of adding more information to the metadata database in the logical and physical descriptions.

2.3 Logical description of the data - dictionaries

The first step in setting up the system is to describe the data that the application manipulates.

For example, when dealing with forest inventories, we have location data, information about the plot (exposure, crown cover …) and information about the trees (species, height …).

Fives tables are used to describe a dictionary (Figure 1):

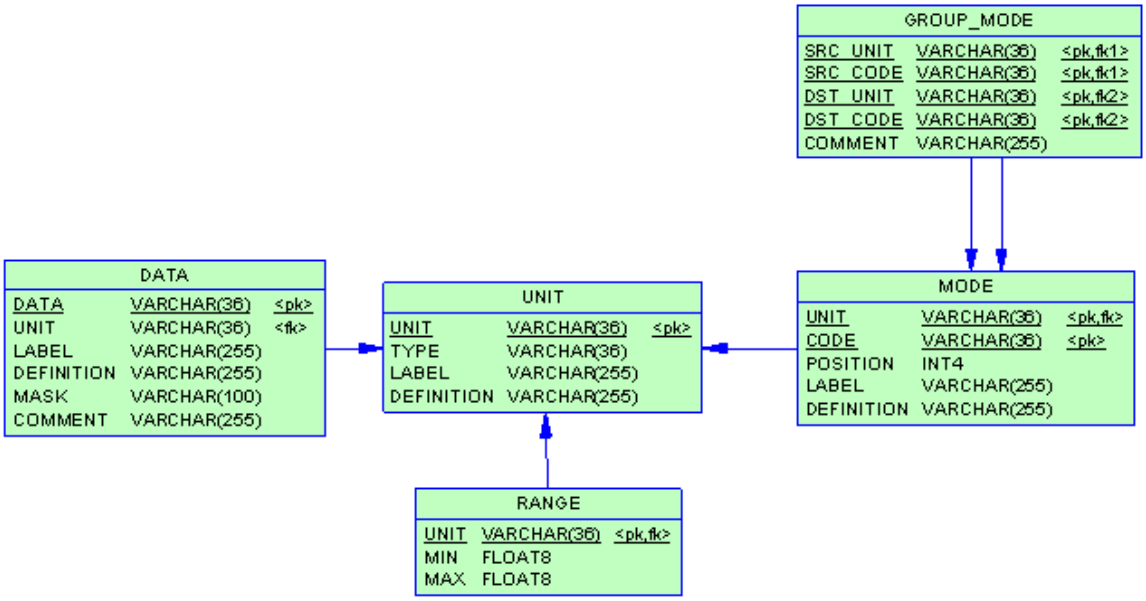

Figure 1: Metadata database – Logical description of the data

The DATA table contains the description of all the different items of information used in the system (a tree identifier, a species, a plot coordinate, a date of inventory …).

Each DATA possesses a UNIT which corresponds to different types of units:

- a continuous unit type often represents a physical measurement (metre, kilogramme)
- a nominal unit type represents a list of codes such as species codes,
- an ordinal unit type represents a sequence of numbers,
- a specific or structured type represents units such as a comments, coordinates,…

The application manages the data in a specific way depending on the type of its UNIT.

A UNIT can be a STRING, an INTEGER, a NUMERIC, a COORDINATE, a BOOLEAN, a DATA, a RANGE or a CODE.

If a UNIT is a CODE, the list of available modes must be described in the MODE table.

If a UNIT is a RANGE, its minimum and maximum values must be described in the RANGE table.

When a data has a nominal unit type, it can be useful to carry out semantic aggregations. A hierarchical organisation is proposed using the GROUP_MODE table. Thus, a very complete list of tree species can be automatically aggregated into a simplified list of species such as broadleaves - conifers.

2.4 Physical description of the data - location

As we have seen, the logical description corresponds to a dictionary containing the definition of all the data. For a generic application which does not know the data, more information is needed to know where the data are stored. This additional information is called the "physical description of the data".

The notion of a FIELD corresponds to the physical implementation of a DATA.

A FIELD can be one of three types (an inheritance allows us to specify the type):

- A TABLE_FIELD describes the location of a field in a database (a column in a table).
- A FILE_FIELD describes the location of a field in a CSV file (the position of a column in a Comma-separated Values file)
- A FORM_FIELD describes the HTML form element corresponding to the data.

The FIELD_MAPPING table describes the links between two fields.

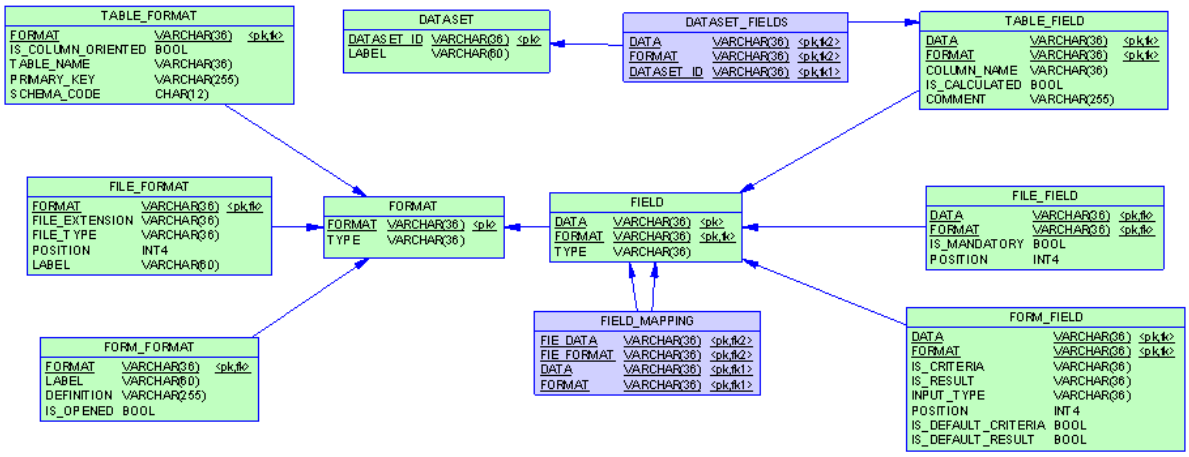

Figure 2: Metadata database – Field types

The FORMAT table corresponds to a set of fields. This concept can be derived into a table format to store the data in a database (TABLE_FORMAT), a file format (FILE_FORMAT) to store data in a text file like a CSV file, or a web form (FORM_FORMAT) to present to an application. Based on the inheritance principle, the system can be extended if required to add a new type of storage.

The DATASET table corresponds to a set of FIELDS (DATASET_FIELDS table). This is another way of grouping fields from a thematic aspect. This concept corresponds to a protocol during field operations: tree measurement, flora observation …

2.4.1 Description of the storage of the data in a database

The most common way to store data is to use a database. A database has its own structure (a schema contains tables which contain columns) and the relationships between the tables have constraints that must be respected. The order of insertion and deletion inside the table must also be respected to comply with the integrity of the data. New metadata tables are mandatory to describe these constraints.

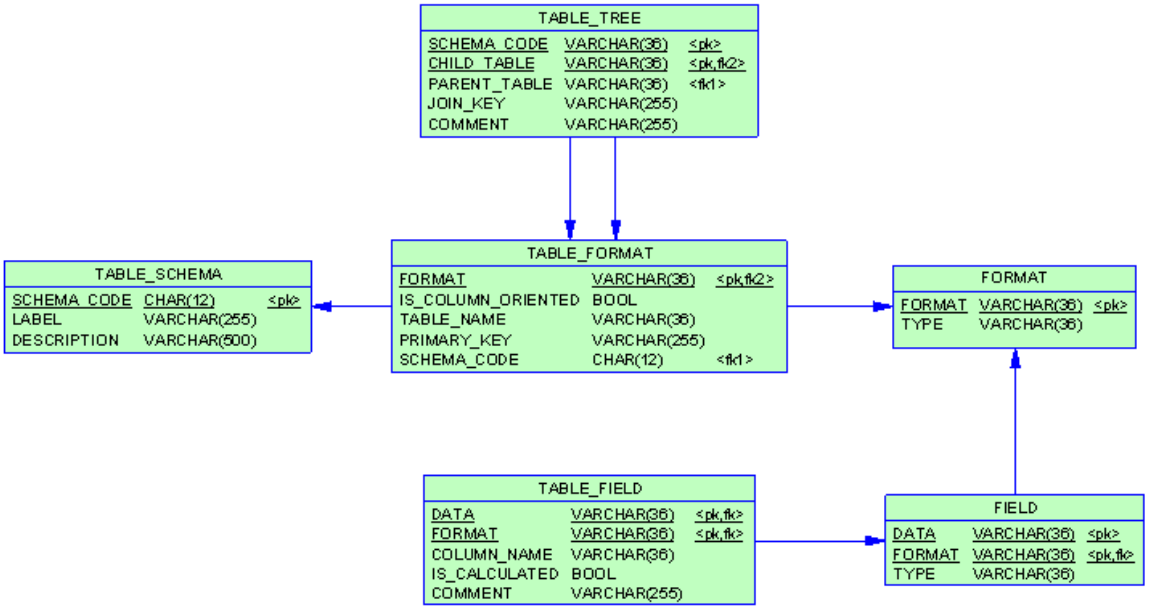

Figure 3: Metadata database – Description of the storage

The TABLE_FORMAT and TABLE_TREE metadata tables contain the information needed to handle the hierarchy between tables.

The TABLE_SCHEMA is a way to group tables by schema or database.

The TABLE_FORMAT table describes the physical name of the table.

The TABLE_FIELD table describes the real name of the column in the table. Some columns that are calculated using a trigger in the database can be flagged, so that the system ignores them when doing an import.

The system can potentially use any existing database structure that complies with the following requirements:

- The system is designed to use PostgreSQL and PostGIS.
- Each datum should be linked to one and only one geographical location (stored in a geometry column).
- The data are stored in a hierarchy of tables but only one child table can be used at any given time (for a given dataset).

Let us expand on the last point: the database structure can store information related to a location in any number of tables. For example a "plot_data" table can be linked with two child tables like "tree_data" and "flora_data". In order to insure that the query module is not too complicated to configure and use, only one of the two child tables can be requested at any time. In this case, two different datasets have to be defined.

Describing an existing database is an easy step and can be used as a starting point when setting up the system (the logical description of the data is done as a second step).

2.4.2 Description of the HTML forms used for querying the data

Once the system knows where to look for the data, the user interface can be described.

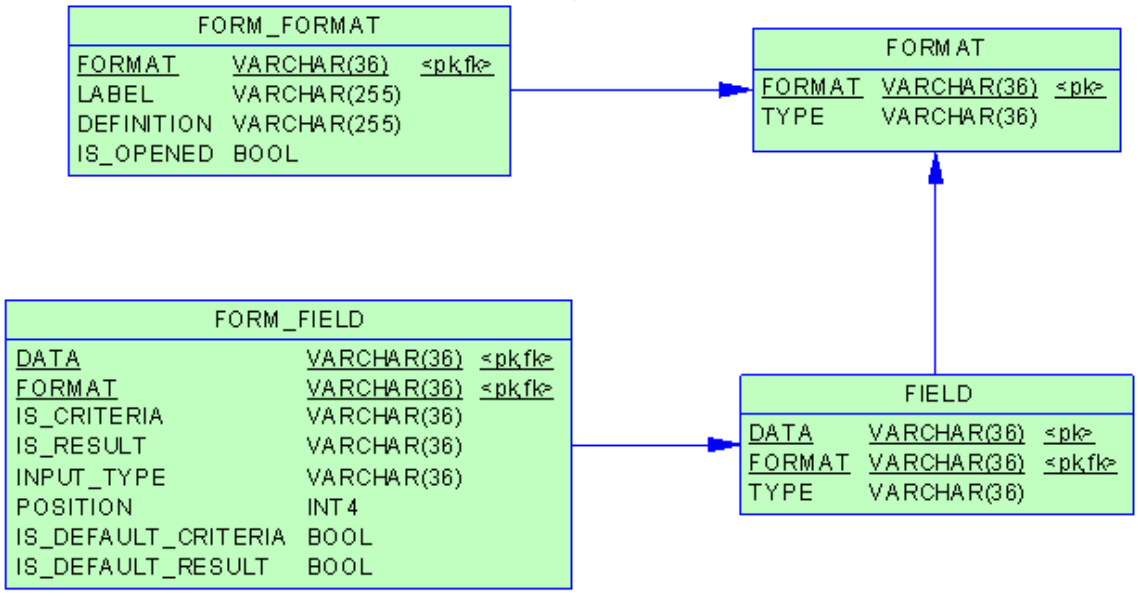

Figure 4: Metadata database – Description of the presentation of data

The FORM_FIELD and FORM_FORMAT tables contain information about web forms. Currently, web forms mainly concern the query module.

A form field can be defined as being a potential query result (it can be displayed in the result grid) and/or as being a query criterion. The input type tells the system which HTML object will be used when displaying the query result (a date selector, a select box …).

The mapping between the columns in the database and the HTML form elements is done using the FIELD_MAPPING table (Figure 2). This allows the system to link a FORM_FIELD to a TABLE_FIELD (usually with the same name).

Once the storage and the presentation of the data are described, the system is ready to be used to query an existing database.

2.4.3 Description of the files used to import data

The system is designed to simplify the importation of a batch of data.

The "DATASET" can also be represented by a batch of flat CSV files. A dataset can be related to a set of fields using two different ways: Through a set of CSV files during the data importation or through a set of table fields during querying. The risk of inconsistencies is avoided using some automatic checks that are made during the configuration of the system The data provider can submit a set of data, check and visualise its content and if necessary the submission can be cancelled and the data replaced.

Once the dataset has been described, the system is able to check the conformity of the submitted data with its logical definition. For example, if a field is linked to a unit of the type CODE, then the value of the field must be one of the expected modes.

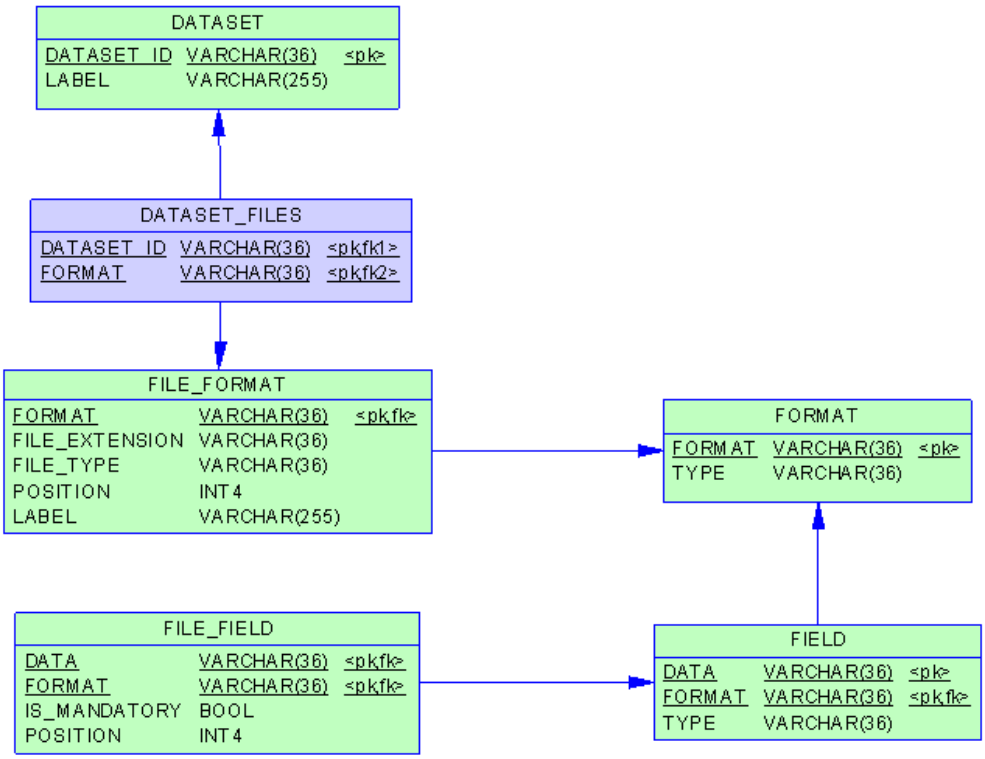

Figure 5: Metadata database – Description of the file datasets

The mapping between the file entries and the columns in the database is done using the FIELD_MAPPPING table (Figure 2). This allows the system to link a FILE_FIELD to a TABLE_FIELD. Note that the FIELD_MAPPING table can also link a TABLE_FIELD to another TABLE_FIELD; this can be used to copy data from one database schema to another.

3 RESULTS OBTAINED: PRESENTATION OF THE APPLICATION

3.1 Functional architecture

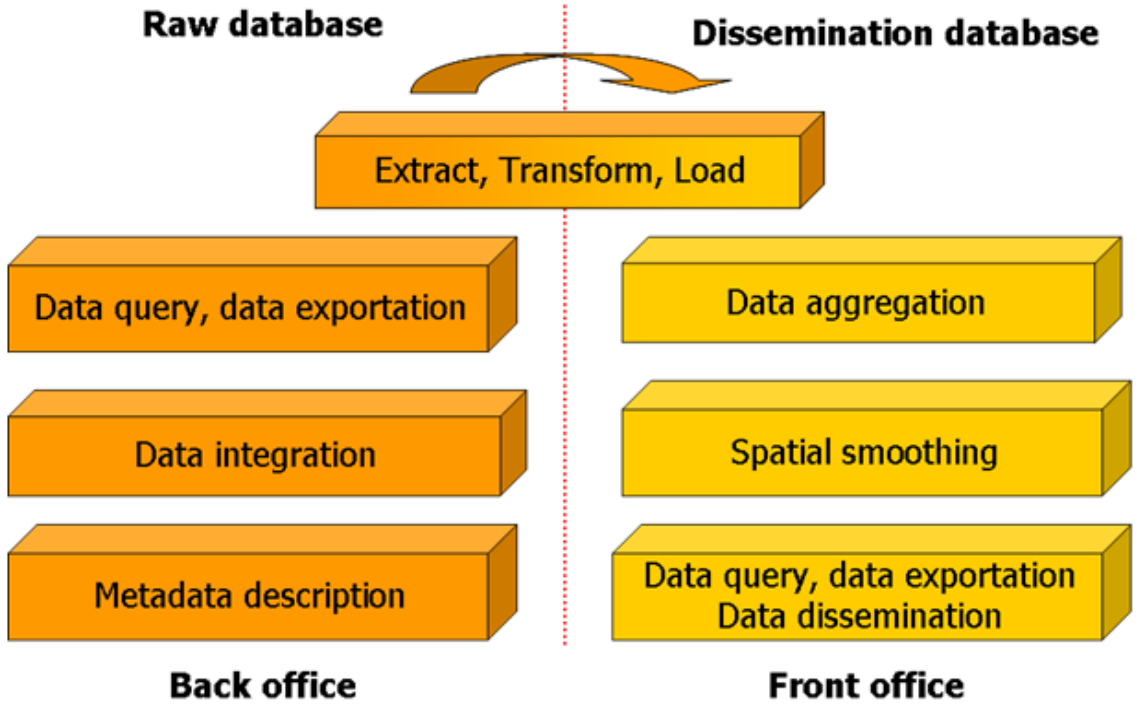

Figure 6: Functional architecture

From a functional point-of-view, we can distinguish the back-office treatments that are carried out on a "raw" database used by the data providers and the front-office treatments that are carried out on the "public" database. This separation allows the data providers to work in the back office without any consequence to the data dissemination in the front office site. The passage from one database to another is done after the validation of the data using the data Extract, Transform and Load module.

The data query module is exactly the same in the back and front offices; the only change is the linked database and its description in the metadata.

3.2 Technical architecture

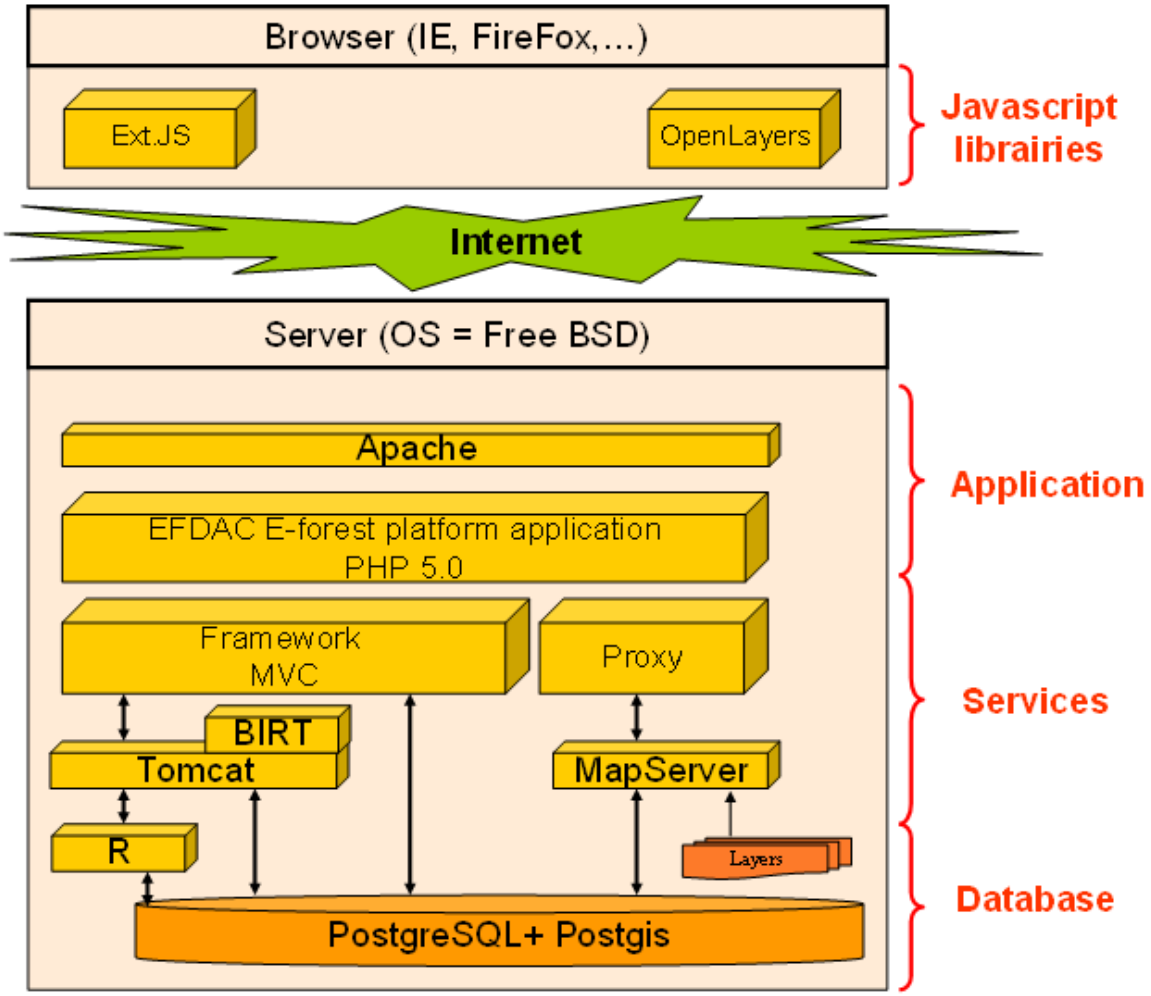

Figure 7: Technical architecture

Technically, the application is built using open-source tools and libraries. On the client side, the Ext javascript library allows the user to build complex interfaces and the OpenLayers library allows them to manipulate maps.

The web site is build in PHP with the Zend Framework and makes the interface between the client side and the web services. The Zend Framework is an MVC (Model-View-Controller) framework that forces the separation of the presentation from the logic; each controller is a module and each module uses a separate back-end service.

The web services (like the integration service, the report generation service …) are built in java and hosted on Tomcat. The interpolation service also makes use of the R library for some geostatistical calculations. Alongside the java services, Mapserver is used to render the maps in the web site and Tilecache is used to accelerate their display.

3.3 Configuring the system

3.3.1 Metadata configuration

In order to simplify the administration of the system, an OpenOffice spreadsheet document is used to manage the metadata. This allows for easy updates of the lists of codes using copy and paste from other documents.

The referential integrity constraints in the metadata database ensure the greater part of the consistency of the system and a SQL script carries out some additional checks during importation (a checkbox form field can only be used with boolean data, a select box should be a list of modes …).

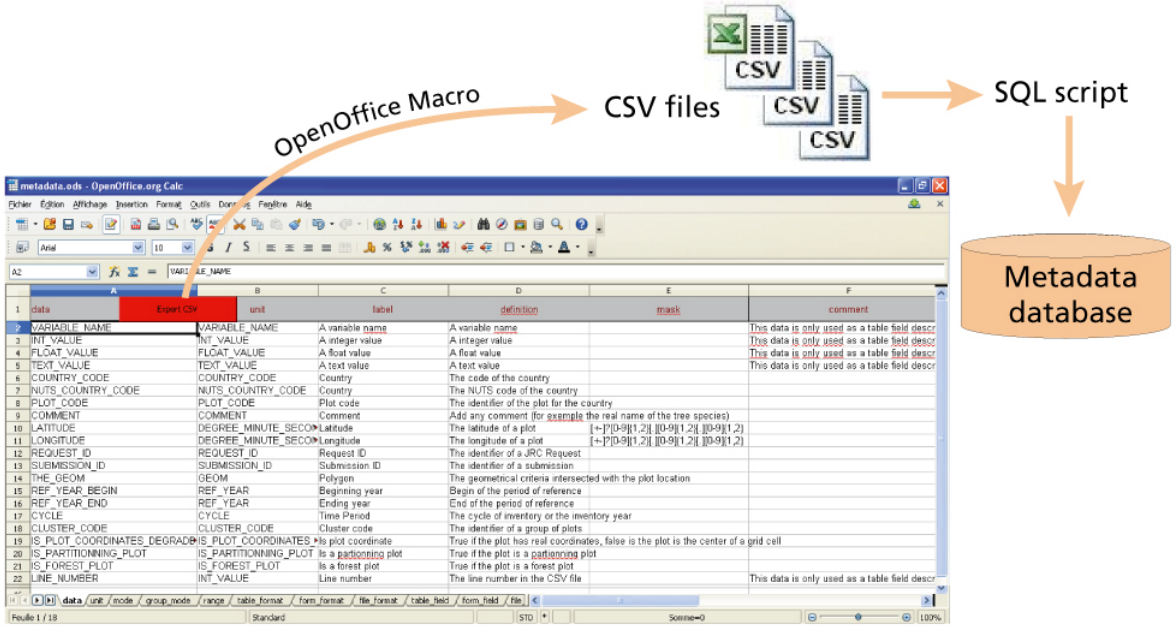

Figure 8: Administration of the metadata

3.3.2 Layer configuration

The configuration of the available layers is stored in the database and the mapserver mapfile. The database contains information about the table of contents (TOC).

LAYER_DEFINITION		
LAYER_NAME	VARCHAR(50)	<pk>
LAYER_LABEL	VARCHAR(100)	
MAPSERV_LAYERS	VARCHAR(500)	
ISTRANSPARENT	BOOL	
ISBASELAYER	BOOL	
ISUNTILED	BOOL	
ISCACHED	BOOL	
MAXSCALE	INT4	
MINSCALE	INT4	
HAS_LEGEND	BOOL	
TRANSITIONEFFECT	VARCHAR(50)	
IMAGEFORMAT	VARCHAR(10)	
OPACITY	VARCHAR(3)	
COUNTRY_CODE	VARCHAR(36)	
HAS_SLD	BOOL	

LEGEND		
ITEM_ID	INT4	<pk>
PARENT_ID	INT4	
IS_LAYER	BOOL	
IS_CHECKED	BOOL	
IS_HIDDEN	BOOL	
IS_DISABLED	BOOL	
IS_EXPENDED	BOOL	
NAME	VARCHAR(50)	
POSITION	INT4	

Figure 9: Layer definition tables

It is possible to provide information about the layer such as the mapserver layer name, the minimum and maximum scale, the opacity, etc. in the LAYER_DEFINITION table. The LEGEND table contains information about the layer's hierarchy, its position in the TOC, etc.

Thus, all the system is fully configurable.

3.4 Data integration module

The data integration module is used to fill the raw database with the data submitted by the providers.

This module:

- Checks data conformity according to the metadata
- Inserts the data into the raw database
- Can run some post-insertion checks (configurable knowledge base)

- Generates a PDF report with the list of errors or warnings.

Once the data are imported and checked, the provider can validate the submission in order to mark them as being available for use.

If the data contain some errors (conformity problems), the errors are logged and reported and the complete submission is rejected. If the data contain some warnings (raised during the post-treatment checks), the data provider is invited to check the suspicious entries before validating the submission.

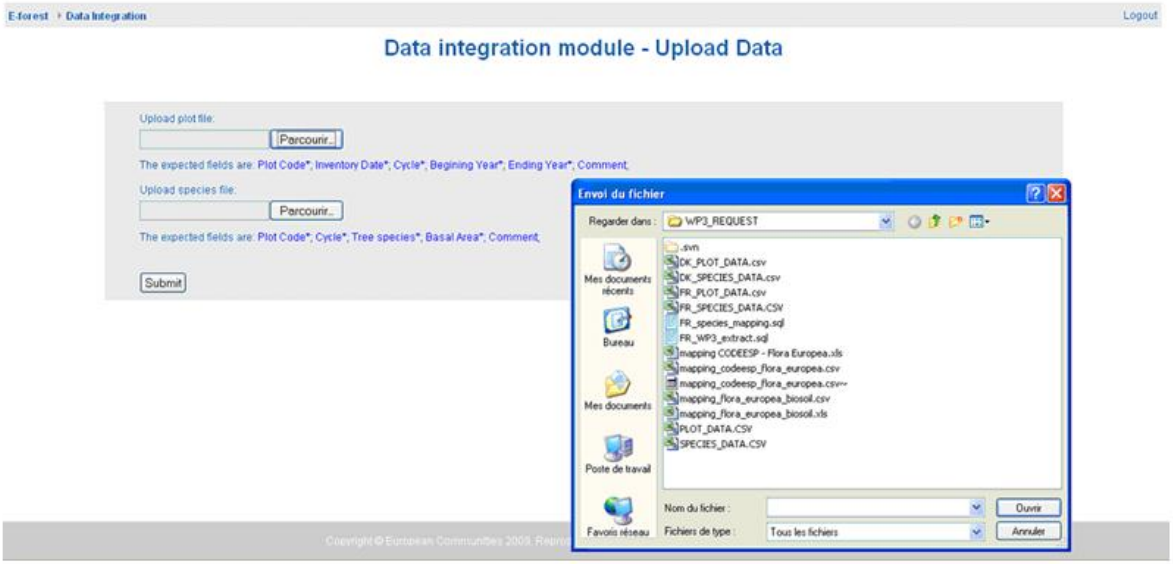

Figure 10: Data uploading

3.5 Data querying module

The user selects a dataset to query. The application then dynamically generates a query panel with the available fields (criteria and column) for the dataset.

The query panel form elements are built using their description in the metadata table (date selector, select box …).

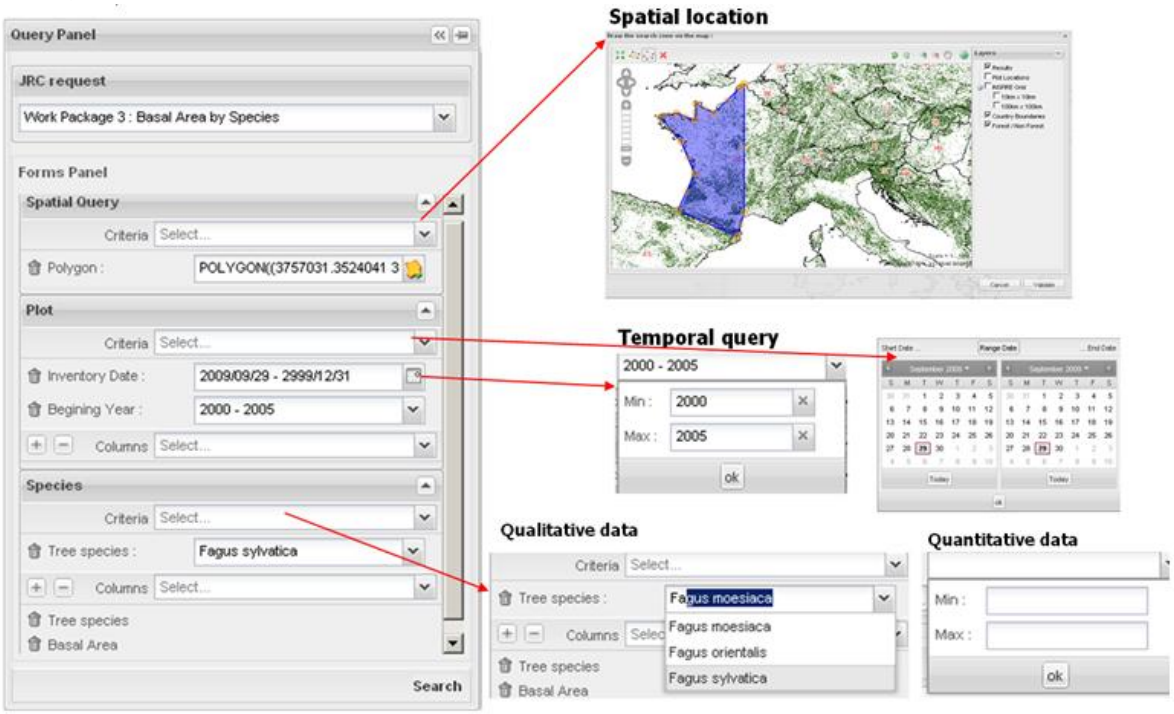

Figure 11: Query Panel Criteria

The user can select the field he is interested in and add other selection criteria. Different types of criteria are shown in Figure 11.

When the user clicks on the “Search” button, the application generates an SQL request corresponding to the query and displays the results both as a map and an array of attributes.

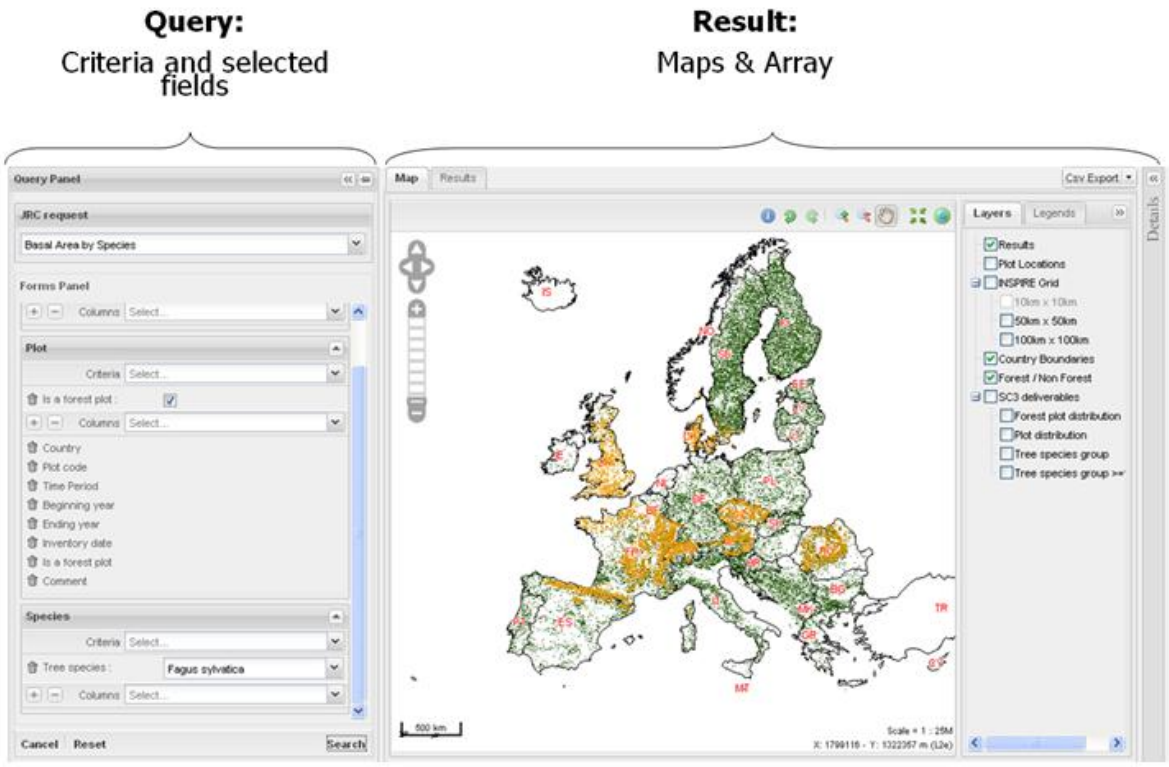

Figure 12: Results of the query

Once the results are displayed, the user can navigate through the result array; use the usual tools on the map; display the details for a specific location or export the results as a CSV file.

The map can also be exported as a PDF file.

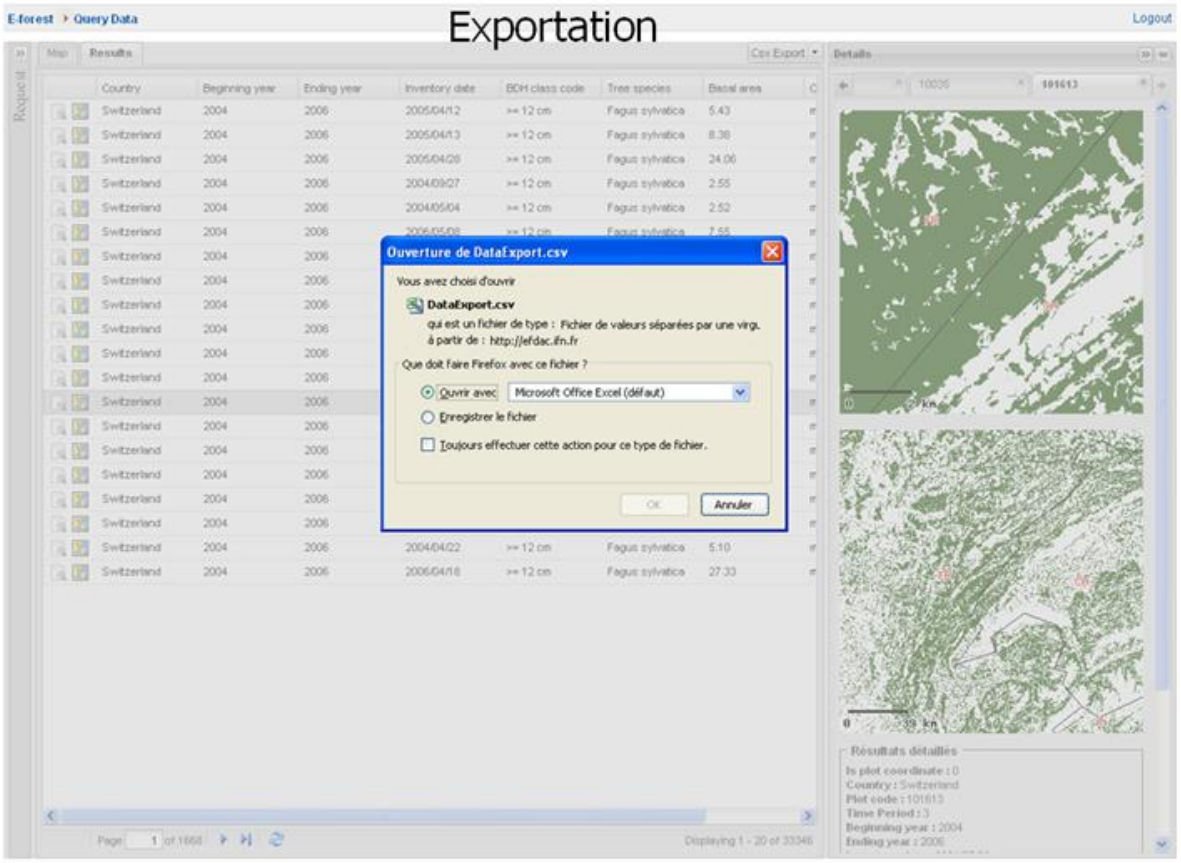

Figure 13: Exporting data

A simplified query module has been developed. It makes it possible to create predefined queries where the user has only one or two parameters to fill in. Theses queries can be categorised by theme and described with a short text.

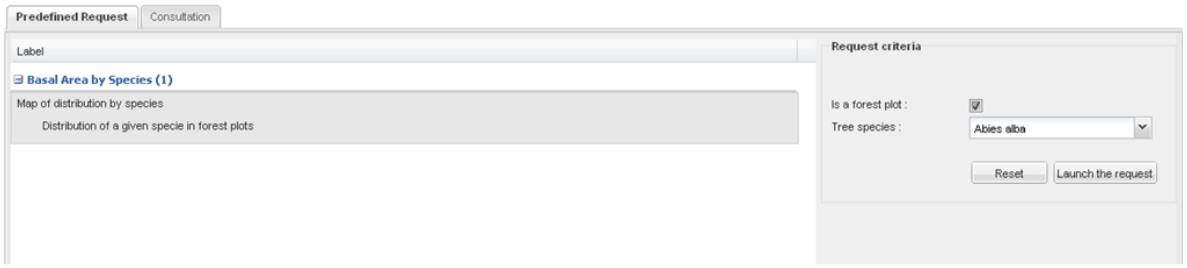

Figure 14: Simplified interface

3.6 Extract, transform and load module

The metadata database can also be used to copy data from one database location to another. This can be used to implement a staging system with the "raw" database visible only to the data providers and only the "public" database containing validated and transformed data visible to the user.

Like an ETL (Extract-Transform-Load) tool, a data mapping can be described between two database schemas and some optional transformations (SQL queries) can be triggered, for example, to hide confidential data, to calculate indicators …

Figure 15: Data transformation

3.7 Data aggregation module

An optional module has been developed and can be used to calculate aggregation information in a geographical area.

The user can select a quantitative value and an aggregation grid (that can be a regular grid or any type of geographical zoning). All the quantitative variables are automatically discovered by the platform using the metadata database.

The system calculates the average of the selected value on each grid cell and generates a result with a confidence interval. The implementation of different estimators is possible - a basic estimator has been tested in the framework of the project and a more advanced one taking into account each country sampling frame and plot weights is being developed.

The result is displayed as a new layer that is added to the map and can be exported as a CSV file.

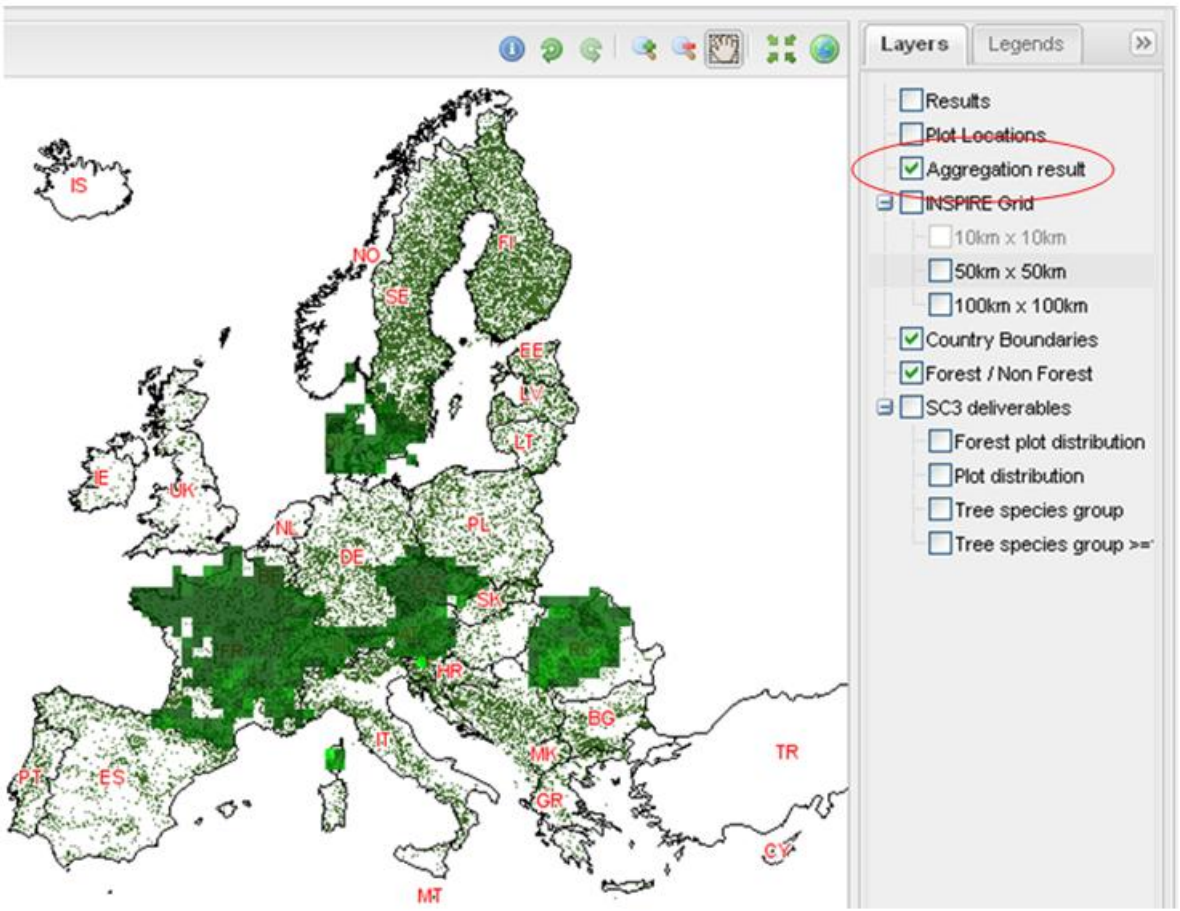

Figure 16: Data aggregation page

3.8 Spatial smoothing

Another optional module can be used to interpolate the data in order to generate a map of the result.

This module is currently limited to a simple IDW (Inverse-Distance Weighting) function but other algorithms could easily be implemented by calling different functions from the R library.

The result is added as a new layer to the map.

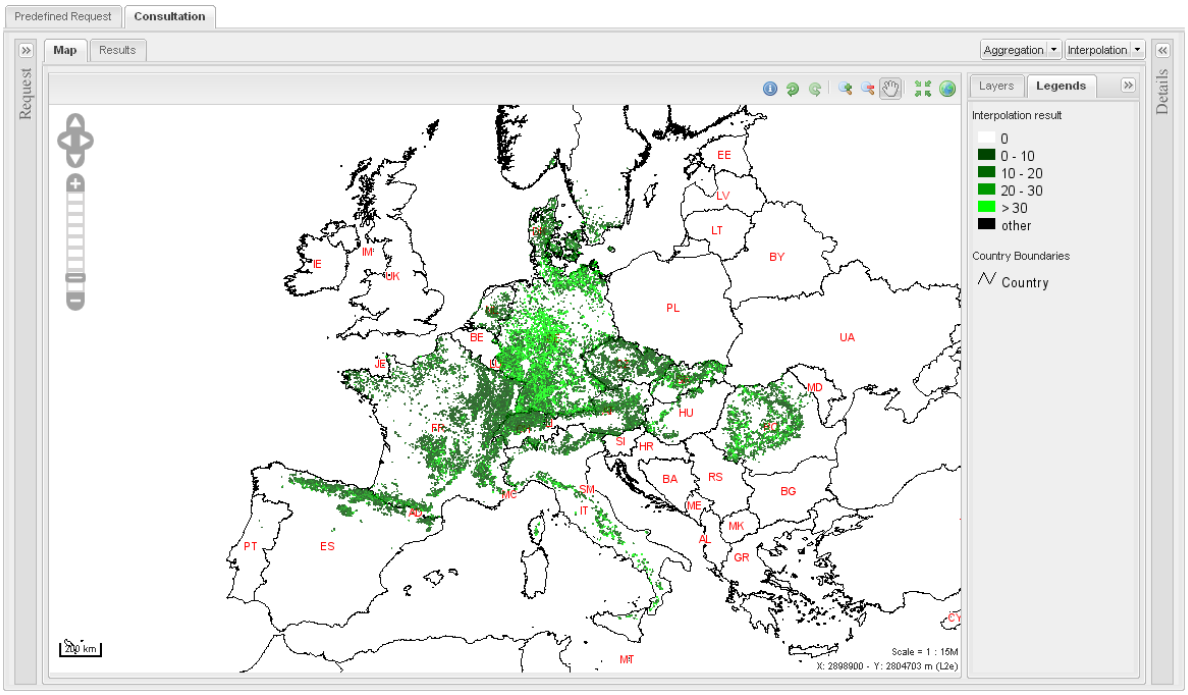

Figure 17: Data interpolation page

This module can limit the graphical effect caused by data provision with different densities.

3.9 Web services dissemination

Different web services can be provided through the E-forest platform.

As mapserver is used to establish a link between the platform and the layers or the PostgreSQL/PostGIS database, it is up to the E-forest administrator to disseminate the different web services (WMS or WFS). If the administrator does not want to expose the web services, a PHP proxy can be used to hide them. In that case, the proxy checks if the session identifier is correctly set (the user is authentified) and, if not, prevents GIS software from accessing the services.

3.10 Examples of use

The system has been developed as a basis for the European NFI Information System but because it is fully generic and configurable, other uses have emerged.

3.10.1 French NFI data publication

The application has been installed on the French NFI web site and is being used to disseminate raw data.

Three datasets are available: dendrometrical measurements and observations, flora observations, and poplar measurements and observations.

The application has also been installed at the ICAS (Romanian Forest Inventory) for their internal use.

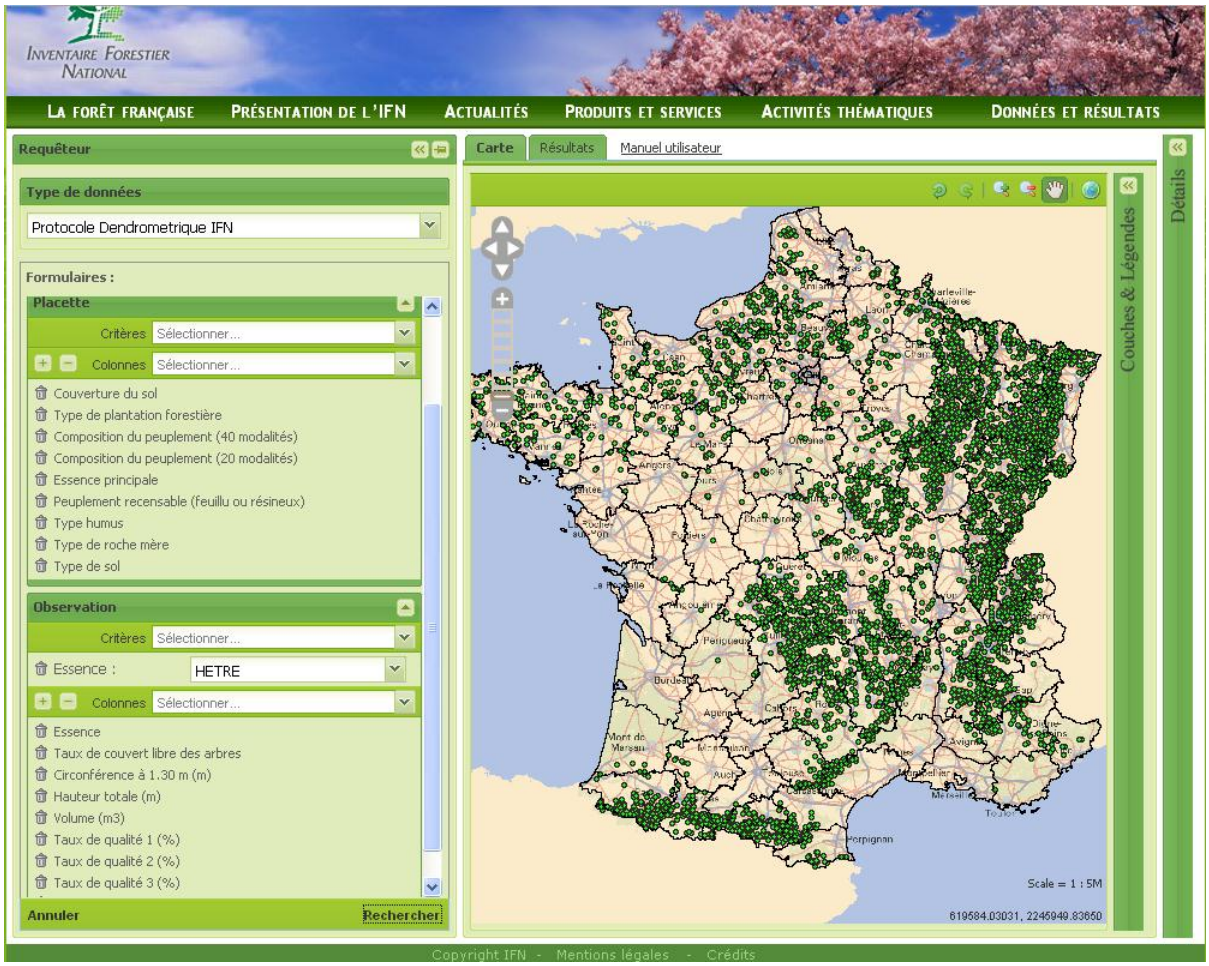

Figure 18: French NFI website

Link: http://www.ifn.fr/spip/?rubrique159

3.10.2 French National Forest Board risk assessment in mountainous areas

Here, the application is being used to monitor events (avalanches, floods, landslides …) that occur in mountainous areas.

This example of the application uses complex geometries instead of simple plot locations.

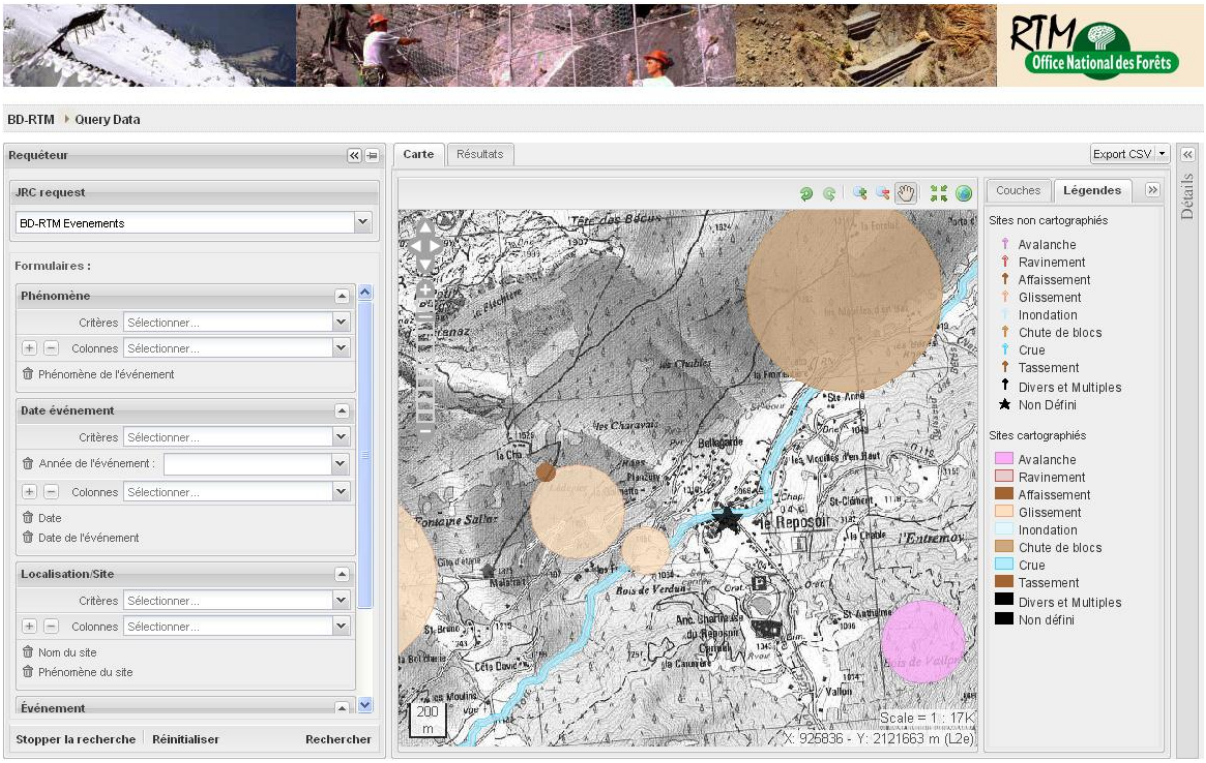

Figure 19: RTM Website

Link: http://rtm-onf.ifn.fr/

4 CONCLUSION

The power of this system lies in the metadata database which provides impressive possible extensions of the system while preserving its adaptability. For example, different improvements can be imagined. Firstly, data providers usually store their data in their own database. In order to avoid the complex data extraction process E-forest Platform, a suitable software interface can be proposed to automatically push the data from the data provider's premises to the central E-forest platform. This interface could easily be configured by adding data mapping between the database provider and the data expected by E-forest.

Another angle of improvement could be to propose a framework based on the same metadata database to develop smartphone or PDA applications. Using the metadata database, it would be easy to automatically build forms to be filled in during field operations and then to transmit compliant data to the E-forest platform using the Internet.

What are the next steps on the E-forest roadmap?

As the Framework contract with the Joint Research Centre of the European Commission is still on-going, new E-forest functionalities have been foreseen. The next step will be to add a design-based estimator to provide statistical results. If the data providers are able to comply with certain stratification conditions (i.e. systematic sampling, etc.) a generic estimation procedure will be available with error calculations.

E-forest is licensed under a European Union Public Licence, so the software and its documentation will soon be available for use on a source forge. Consequently, new functionalities will be proposed. For example, the French National Forest Board risk assessment in mountainous areas (RTM) has started to use the E-forest platform to disseminate their data on the Internet. Their participation has confirmed that E-forest is fully generic and that new functionalities can easily be added (improvement in the user interface). Thanks to other new projects with French partners, additional improvements will soon be made: for example, management of hierarchical lists of codes for taxonomic reference, a new module to provide generic web forms that can be filled with data to insert into the database, etc…

Even if the powerful internal E-forest metadata database is not completely based on classic standards, the platform provides a full generic framework to build new thematic geoportals in just a few hours. The user simply installs it, configures it, and describes the database or dataset they want to manage and then, the system is ready to provide all its functionalities using the Internet (data gathering, data query, interpolation, data dissemination…). Thanks to its open source licence, we hope that E-Forest will be the starting point of a new Community on the Internet interested in sharing and increasing the potential of the generic platform to build thematic geoportals.

5 LINKS

E-forest: http://efdac.ifn.fr/eforest/index/welcome (restricted access)
French NFI data publication: http://www.ifn.fr/spip/?rubrique159
RTM: http://rtm-onf.ifn.fr/ (restricted access)
European Forest data Center (EFDAC): http://efdac.jrc.ec.europa.eu/
INSPIRE : http://inspire.jrc.ec.europa.eu/
OGC: http://www.opengeospatial.org/
SDMX : http://sdmx.org/
Dublin Core: http://dublincore.org/
TDWG : http://www.tdwg.org/
TAPIR : http://www.tdwg.org/activities/tapir/
EUROSTAT : http://epp.eurostat.ec.europa.eu/portal/page/portal/eurostat/home/
PostgreSQL : http://www.postgresql.org/
PostGIS : http://postgis.refractions.net/
R : http://www.r-project.org/
Tomcat: http://tomcat.apache.org/
BIRT: http://www.eclipse.org/birt/phoenix/
MapServer: http://mapserver.org/
TileCache: http://www.tilecache.org/
Zend Framework: http://www.zend.com/en/community/framework
Ext JS: http://www.sencha.com/products/js/