Sharing botanical information using geospatial databases

João Silva, Cristina Ribeiro, and João Correia Lopes

Faculdade de Engenharia da Universidade do Porto
Rua Dr. Roberto Frias
s/n 4200-465 Porto, PORTUGAL
{ei04032,mcr,jlopes}@fe.up.pt

Abstract. Current botanical databases store taxonomic and specimen data, and can be queried using standard text-based queries. A simple example is retrieving all recorded specimens of *Metasequoia glyptostroboides*. In some botanical databases, this might result in a set of georeferenced results. However, they lack the ability to provide spatial queries, such as finding all occurrences of *Metasequoia* living in the FEUP campus area.
We propose a flexible platform for querying a spatially-enabled database. This platform can support a botanical database, extended with spatial querying capabilities. The details of the underlying database are hidden from its clients, but efficiency in handling spatial data and its spatial querying potential are essential. For full backwards compatibility, spatial queries can be blended with standard queries. This solution is technology-independent from the clients' point of view and uses the Darwin Core standard format for botanical information exchange.

1 The botanical domain

Botany is a large information domain, with over 287 000[1] different identified species, as of 2004. Also, Botany has a very structured model for representing information. Since Carl Linnaeus invented *Linnaean Taxonomy* [2], this model has been used as a standard to classify living organisms into categories, called *taxons*. As an example, the taxonomic classification of the olive tree is shown in Table 1.

1.1 Why does Botany need GIS?

The basis of most botanical databases has been the taxonomical classification. Large online databases like the PLANTS database [3], the Harvard University Herbarium (HUH) [5] or the Global Biodiversity Information Facility (GBIF) [1] are widely known and used. These databases include a basic way of associating

[1] in http://en.wikipedia.org/wiki/Plant.

Cladus (general)	Eukaryota
Regnum	Plantae
Divisio	Magnoliophyta
Classis	Magnoliopsida
Familia	Oleaceae
Genus	Olea
Species	Olea europaea

Fig. 1. Taxonomic classification for the olive tree

locality elements. The person who identifies an occurrence can append written references, such as "On Van Stadens Berg, nearest to Galgebosch"[2].

GBIF [1] takes the next step towards georeferencing. It offers a very large volume of georeferenced information, regarding not just plants. This information, however, comprises mostly geographically approximated occurrences, retrieved from historical records. A similar approach has been used in geographical systems dealing with forests [17].

Geospatial information is very important for botanists, because it allows them to trace the occurrences of specimens much more precisely, and match them to other geospatially-oriented phenomena, such as weather patterns in a specific area. This kind of information can be valuable in establishing cause-effect relationships between phenomena and provide insight on the evolution of endangered species or invasive plants, for example.

1.2 Botanical information exchange - Darwin Core

Since a lot of botanical information has already been catalogued in information systems, a new-born system for this domain must be able to interact with currently existing ones. Interoperablity is therefore one of the challenges of building online services to support biodiversity mapping [15]. For a system to be interoperable, it must use a standard information exchange format. XML (eXtensible Markup Language) is a widely used format for information exchange, being especially adequate for data transfer throughout the Web. Botanical data is no exception, and there is a proposed standard for this kind of information, *Darwin Core*, currently pending final approval by TDWG [4]. While not being (yet) a *de-facto* standard for biological information exchange, *Darwin Core* is already extensively used. Some good examples are several large projects in the biological domain[3].

[2] From the HUH [5] database, for the specimen with Collection Number 4697.

[3] Darwin Core, version 1.21 schema (revised version) is used in GBIF, MaNIS, HerpNet, OrNIS, and FishNet2 databases [10].

2 Geo-Botanical Repositories

There are several excellent public botanical databases. However, their query capabilities are built on top of text parameters. When a user queries the database, the parameter can be the required value for any level of taxonomic information or the *Collection Code*, for example. In a more *pseudo-spatial* approach, querying about the specimens in a specific country can yield georeferenced results, such as in the GBIF database [1]. However, only the results are spatial (points and place marks), not the queries themselves, e.g., users can specify values for certain text-based parameters (such as the botanical occurrence's country of origin), but cannot use arbitrary polygons as part of the query's restrictions.

2.1 What is right about current solutions

Several requirements for botanical databases have already been met. These proven approaches serve both as directions and examples.

The sharing of information is the first strong point in a botanical database. The GBIF database [1] offers a range of Web Services [6] that allow other applications to interact with it. This is good because it allows the sharing of information in interchangeable formats, such as *Darwin Core*. Selecting the correct formats for geospatial data viewing is also very important. With the spreading of solutions such as Google Maps and Google Earth, end-users are becoming more accustomed to using geospatial data in their everyday life. GBIF [1] saw this as an opportunity and now offers part of their data in KML[4] format for public download and viewing. Another important feature of a botanical database is its openness. Botanical databases gather information from various parts of the globe. Local institutions catalogue their specimens and send in the information, which is made publicly available. To allow a database such as this to grow, it must provide users with a simple way to insert specimen information and to retrieve it.

2.2 What is missing in current solutions

There is much untapped potential for a different approach:

> What if the user wants to specify an arbitrary *polygon* and query about the specimens recorded *within*/*outside*/...[5] the polygon?

This kind of questions can only be answered by spatial querying, supported by a geospatial database. A simple example is now presented to demonstrate the potential and usefulness of this approach.

In Figure 2 we can see a map of the Iberian Peninsula. In this map, the banners are placemarks for some fictional botanical occurrences. It seems obvious

[4] Keyhole Markup Language.
[5] Or any OpenGIS Consortium [8] Specification spatial restriction operator.

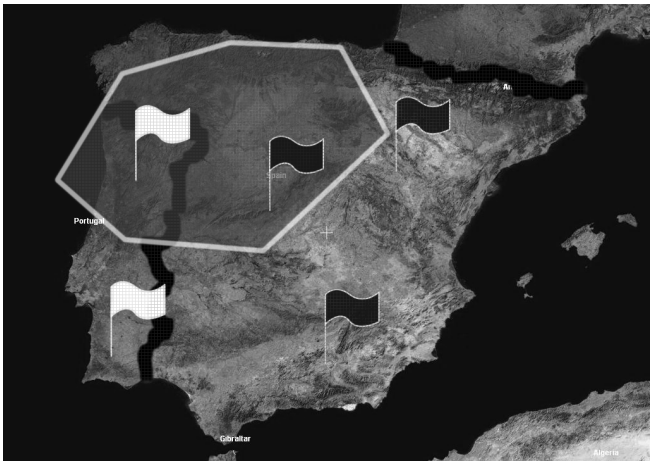

Fig. 2. The Iberian Peninsula, some botanical occurrences and a polygon

that botanical occurrences are not constrained by country borders, so it makes sense to search for occurrences in user-specified areas. It would be interesting for the user (person or machine) to use a polygon such as the one in Figure 2 as the query restriction, or "country border". In this case, the system would retrieve the two banners inside the polygon (a dark one and a light-coloured one).

This sort of *fine-grained* geospatial restrictions yields several interesting uses, such as:

- Finding patterns between botanical and weather-related phenomena in a given region;
- Tracing species propagation and identifying potential pests or invasive species;
- Providing information to tourists interested in visiting places where certain species of plants exist;
- Helping botanical parks to compose their daily tours using spatial queries. (see Section 6 for more details);
- Helping pharmaceutical companies keep trace of places where certain botanical species are discovered;
- Helping environmentalists keep trace of species evolution in a given area;
- Helping architects in charge of city planning/landscaping to keep track of the specimens planted in any city garden or park.

Each of these uses can translate into a client for the system, using its spatial querying capabilities to support the client's business logic.

3 A spatial querying system and its requirements

We propose a spatial query language aimed at botanical repositories. Its use will be illustrated in a botanical data server and some example clients.

3.1 The data model

The data model underlying the repository captures several concepts pertaining to the classification of species and the cataloguing of specimens, linking these to their geographical location. This model [18], includes classes for taxonomic classification, such as `Familia` or `Genus`. Physical locations are represented by `Geometries`. For specimen classification purposes there is an `Occurrence` class. Other classes such as `Tour` help in the management of a botanical website. More detail can be found in the full project report [16]. The data model includes the concepts available in the Darwin Core (DC) standard. This grants interoperability to the system, because it is able to easily import and export DC formatted data.

3.2 The query language

The query language, intended to access the information in the repository, was designed with the following features in mind:

1. **Simplicity**
   Hiding intricate low-level implementation details of spatial database access, as these are mostly irrelevant to end-users. As an example, to retrieve all specimens inside an area, the user only needs to know what a polygon *intersection* is.
2. **Flexibility**
   Allowing for elaborate or simple queries, depending on the end users' needs.
3. **Expressiveness**
   Users can access the whole set of features offered by the underlying database.
4. **Technology independence**
   Client programmers can access the central database, regardless of the programming language that they use or other technological constraints.
5. **Interoperability**
   Allowing the language to be easily used by machines.

To fulfill all these requirements, the *PlantGIS* query language was developed. *PlantGIS* takes its inspiration from the standard SQL notation, some *Darwin Core* attribute names and, of course, the *OpenGIS Consortium*(OGC) spatial restriction operators [8], such as `equals`, `disjoint`, `intersects`, `touches`, `crosses`, `within`, `contains` or `overlaps`.

The language is very simple, as can be seen in its graphical representation [20]. It resorts to the meaning of each of the spatial operators defined by OGC [8]. To compose the query in the example, the user has to know that the `within` operator will include all occurrences *spatially within* [8] the argument polygon. To use a more informal description, this operator will return polygons "fully inside" the argument polygon.

To provide a high degree of flexibility, the language supports the `AND`, `OR` and `NOT` boolean operators. The precedence is given by the level of each query

in the XML tree. Common *text-based* querying capabilities widely used in other databases are also available here. Both types of queries can be combined freely, as demonstrated in the example query [19].

Using XML allows interaction with a broad range of clients. From handheld terminals to full-fledged Web servers, there is almost always an XML parsing library available. Using XML transported via Web Services, technology restrictions are dramatically reduced.

3.3 Information output formats

Query results are produced in two main formats: *Darwin Core* and KML. KML is an interesting format for exchanging spatial information. It is Google Earth and Google Maps' default document format. Some botanical databases also offer their results in KML format. GBIF, for example, uses it as its geospatial information exchange format.

4 A botanical data server

All botanical databases have their specific needs; however, all of them share the need for geospatial information and querying. With this in mind, the notion of a central server application to serve only the clients' common needs becomes an interesting alternative. The specific needs of each domain are left for clients to implement. The database would then grow by exchanging its geospatial querying capabilities for some information about the clients' specimens.

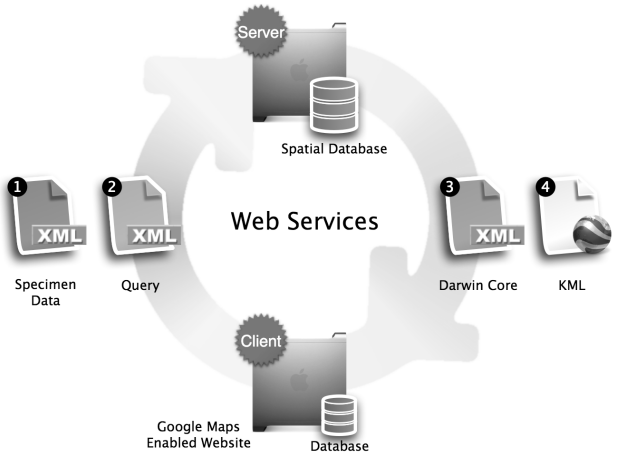

Fig. 3. Client/Server interaction

Figure 3 is a high level physical diagram, depicting the interactions between the server and its client applications.

Clients can send *specimen occurrence* data to the server, to publish new botanical occurrences. The server processes the request and responds to the client with a global identifier. This identifier can, from then on, be used by the client to have direct access to that occurrence's record in the server's spatial database.

Queries, formatted in *PlantGIS*, are sent to the server. They are validated against the *PlantGIS* query language specification. As for the results of the query, they represent specimen occurrences (points or polygons). The available formats for output are *Darwin Core* and KML. *Darwin Core* output ensures the interoperability of this system, and KML is used to improve result visualisation by humans. Clients can include Google Maps visualizers for query results received from the server. At the same time, KML files can also be opened in Google Earth for offline viewing.

4.1 Client scenario

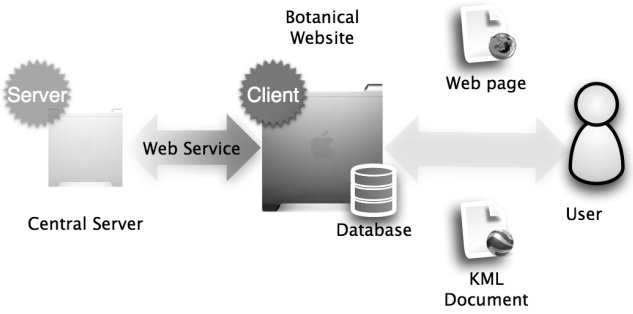

Fig. 4. Physical diagram for a client scenario

An hypothetical client scenario is shown in Figure 4. In this case, the client is a website, with its own database and business logic. The user interacts with the client, which in turn uses the server's spatial querying capabilities to provide users the information they need.

5 Technologies

The server application was designed to be an API. As such, it must be flexible, open, and easily expandable. Thus, any of the technologies selected had to be based on *open-source* software.

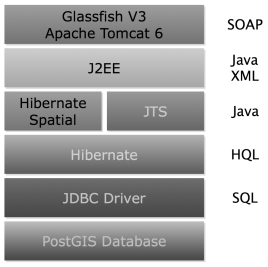

Fig. 5. Technologies used in the server application

Figure 5 shows the choice of technologies for this system. To the right of each layer is the predominant language/information exchange format it handles.

PostGIS Database PostGIS is an geospatial extension for the PostgreSQL DBMS. This extension adds dedicated spatial data types, such as `Geometry`, and spatial operators from the OpenGIS Consortium. The most prominent *open-source* alternative to PostGIS is MySQL with Geospatial Extensions. However, one feature sets the extension of PostgreSQL apart from MySQL with Geospatial Extensions: its full OpenGIS Consortium standards compliance [12,11].

Hibernate Hibernate is a widely used object-relational mapping solution for Java. It makes database interaction transparent to the programmer, mapping each table in the database to Java classes. Its query language, HQL, is translated by the framework to SQL in run-time and injects it into the database. This introduces additional overhead in the system but allows for complete Object Oriented querying, since HQL can include references to Java object fields and not only table/column names. An equally important feature is database vendor independence. Since no *native* SQL is actually written, the database adaptor (JDBC Driver) can be changed if one decides to replace the underlying database. The Hibernate framework will just map its HQL to a different SQL dialect to encompass the change. Also, using Netbeans, database changes are quickly dealt with because the IDE allows for automatic regeneration of the corresponding Java objects, and since there is no hard-coded SQL, the IDE itself helps with the process of *refactoring*.

Hibernate Spatial Hibernate natively supports the most common database datatypes (`integer`, `varchar` and so on). Geospatial data types, such as `Geometry` are handled by Hibernate Spatial, allowing Hibernate to map these data types and making their usage transparent to the programmer. This plugin takes HQL with geospatial data types and generates the native SQL (depending on the underlying JDBC adapter) to take advantage of the indexes and optimization of spatially-enabled databases.

Java Topology Suite (JTS) Java Topology Suite is an efficient library for handling geometry-related operations. It provides methods for calculating `union`s, `intersection`s and other operations between `Point`s and `Polygon`s. Hibernate Spatial works in conjunction with this library to perform all the necessary calculations when querying the database using spatial restrictions[6] and return the records that match the criteria.

6 The *FEUP Park* Client

To evaluate the usefulness of the developed server application, a "test case" client application was developed. This application was named *FEUP Park* and simulates a small botanical park's most common use cases[7]. It demonstrates the usage of the server's spatial querying capabilities.

6.1 Insertion of new records

The client application is capable of inserting new occurrences. There are several interesting features in the web page designed for this purpose. Through real-time KML parsing, occurrence insertion is easier for the user. The system is able to receive KML files and show lists of the polygons and points contained in that file. These entities can then be selected by the user and allocated to the new occurrence. Also, the client can display uploaded KML files on Google Maps to assist the user in building the query. Figure 6 shows this visualizer; the two lists at the bottom show the names of points and polygons inside the uploaded KML file.

KML files can be generated from Google Earth (after manual tagging of points and polygons within the program) or even up/downloaded from GPS receivers. An example is the "GPS utility" software, that can be used with *Garmin* GPS receivers [13]. This makes geo-referencing easier, because users can walk around the park, tag the specimens' locations and save them to the GPS receiver. Then, using this Web interface, they can upload the generated KML and tag each point with its associated taxonomic classification.

Using data collected from the *Wikispecies* website, the system can fill in the upper taxons' values after selecting a value for a taxon. This is done in real time, using AJAX (Asynchronous Javascript and XML). Upon the submission of an occurrence, data is stored locally in the park's database and also sent to the central server.

6.2 Spatial Querying

There is a dedicated area for querying the specimen database, assisting users in building their queries.

[6] OpenGIS Consortium specification [8] restrictions such as `within` or `intersects`.

[7] This application is currently under development and a demonstration version will soon be available at `http://paginas.fe.up.pt/~ei04032/plantgis`.

Fig. 6. The client's Google Maps visualizer.

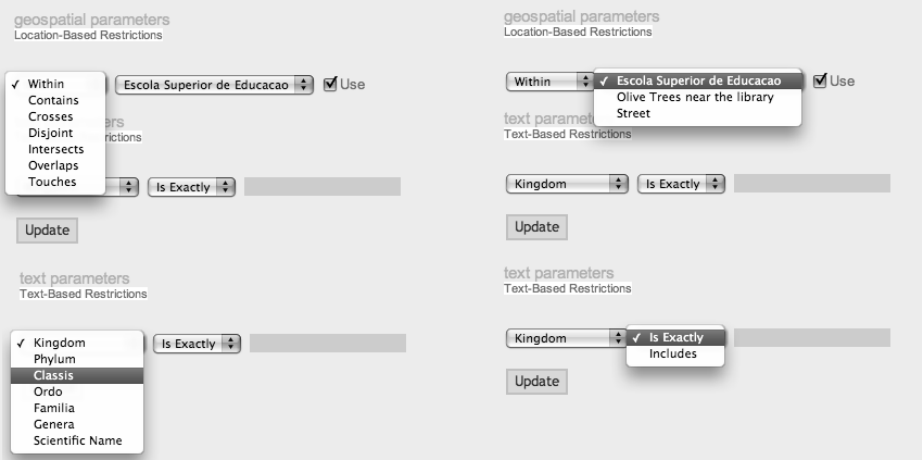

Fig. 7. Selecting operators and polygons while building a query

Figure 7 shows part of the query building process, with all the parameters and operators available in the client. The user can select several of these. The client translates their options to a *PlantGIS* query, sends it to the server, and retrieves the results in Darwin Core or KML. Here again, if the user has uploaded his/her own KML files, polygons inside them can be used as query arguments.

Designing Tours The Park Client interacts with the server but has its own requirements. To illustrate the support that the server can lend to such activities, a simple tour making page was developed. It is built on top of the query page. The user can order the results of a query (occurrences) and save a new tour. It then becomes available for viewing on Google Maps or Google Earth (KML exportation).

7 Conclusions

This proposal aims at enabling botanical databases with geospatial querying capabilities. To achieve that goal, it offers a query language for geo-botanical data. The proposed system is also able to exchange biological information with similar databases using standard formats. This interoperable, centralized system can be used by many types of clients, and some examples have been identified. One of these clients has also been demonstrated, showing some of the usages of the proposed solution.

8 Future Work

Some future development perspectives for this system include a client for mobile devices. This client will be capable of reading RFID (Radio-Frequency Identification) tags from botanical specimens in the field, retrieve their information and display it on the mobile terminal's screen. The portable terminals could be used by park visitors, replacing conventional tags as the main identification method of specimens. An experiment has already taken place in the context of this work, in a collaborative effort between FEUP and iUZ Technologies[8].

Also, a more sophisticated client for building queries could be developed, enabling users to specify more complex queries than those currently supported by the demonstration web application.

References

1. Global Biodiversity Information Facility: GBIF Portal. `http://data.gbif.org/welcome.htm` (Consulted on May 2009)

[8] A small video was shot in the FEUP campus area, depicting the usage of this prototype client. It can be seen at `http://paginas.fe.up.pt/~ei04032/plantgis/paper/DemoRFID.mov`.

2. Wikimedia: Linnaean Taxonomy. http://en.wikipedia.org/wiki/Linnaean_taxonomy - (Consulted on May 2009)
3. USDA, NRCS. 2009 : The PLANTS Database. http://plants.usda.gov. National Plant Data Center, Baton Rouge, LA 70874-4490 USA. - (Consulted on May 2009)
4. Biodiversity Information Standards: TDWG: Homepage. http://www.tdwg.org/ - (Consulted on May 2009)
5. President and Fellows of Harvard College : Harvard University Herbarium. http://asaweb.huh.harvard.edu:8080/databases/specimen_index.html - (Consulted on May 2009)
6. GBIF: About - Using data from the GBIF Portal. http://data.gbif.org/tutorial/services - (Consulted on May 2009)
7. GBIF: Datasets - A - GBIF Portal http://data.gbif.org/datasets/ - (Consulted on May 2009)
8. Open Geospatial Consortium Inc: OpenGIS® Implementation Specification for Geographic information - Simple feature access - Part 1: Common architecture. - 2006-10-05
9. Biodiversity Information Standards: Purpose and design goals of the Darwin Core. http://wiki.tdwg.org/twiki/bin/view/DarwinCore/GeospatialExtension - (Consulted on May 2009)
10. Biodiversity Information Standards: Darwin Core Versions. http://wiki.tdwg.org/twiki/bin/view/DarwinCore/DarwinCoreVersions - (Consulted on May 2009)
11. MySQL AB, Sun Microsystems, Inc.: MySQL 5.0 Reference Manual http://dev.mysql.com/doc/refman/5.0/en/mysql-gis-conformance-and-compatibility.html - (Consulted on May 2009)
12. PostgreSQL Global Development Group: PostGIS receives OGC compliance http://postgis.refractions.net/pipermail/postgis-users/2006-September/013058.html - (Consulted on May 2009)
13. GPS Utility Ltd.: GPS Utility - Receivers http://www.gpsu.co.uk/gpsrecs.html - (Consulted on May 2009)
14. Network Working Group: RFC 2045 - Multipurpose Internet Mail Extensions (MIME) - Part One: Format of Internet Message Bodies http://tools.ietf.org/html/rfc2045 - (Consulted on May 2009)
15. Robert Guralnick and David Neufeld: Challenges building online GIS services to support global biodiversity mapping and analysis: Lessons from the mountain and plains database University of Colorado Museum, Department of Ecology and Evolutionary Biology Biodiversity Informatics - pp. 56-69 - February 2005
16. João Silva: Master's Thesis, Sharing botanical information using geospatial databases Faculdade de Engenharia da Universidade do Porto - Available July 2009
17. Paulo Costa de Oliveira Filho, Atilio Antonio Disperati et al. Um sistema de informacoes geograficas como suporte a um experimento florestal na flona de Irati-PR Unicentro, Depto. de Engenharia Ambiental PR 153, km 7 Bairro: Riozinho - 84.500 - Irati - Paraná - April 2005 (Article)
18. João Miguel Rocha da Silva Data Model http://paginas.fe.up.pt/~ei04032/plantgis/paper/data_model.pdf - May 2009
19. João Miguel Rocha da Silva An example of a query, formatted in PlantGIS http://paginas.fe.up.pt/~ei04032/plantgis/paper/Sample.xml - May 2009
20. João Miguel Rocha da Silva PlantGIS Language graphical representation http://paginas.fe.up.pt/~ei04032/plantgis/paper/pgis_tree.jpg - May 2009