Design and Implementation of Client Interface Framework in *BODHI*

A Project Report

Submitted in partial fulfilment of the

requirements for the Degree of

Master of Engineering

in

Computer Science and Engineering

by

Bandi Sreekanth Reddy

Department of Computer Science and Automation
Indian Institute of Science
Bangalore – 560 012 (INDIA)

January 2002

Contents

List of Figures

Acknowledgements

I take privilege to express my deep sense of gratitude and sincere thanks to my guide Dr. Jayant R. Haritsa for his valuable guidance, genuine suggestions, encouragement.

I would specially thank my colleague *Nagesh* for his extensive cooperation during the period of project. I would thank *Srikanta* for valuable suggestions in the project work and for his ever helping hand. I thank the remaining *DSLites*: *Leela*, *Kanth*, *Pankaj*, *Amit*, *Maya*, *Vikram* and *Suresha* who made DSL a homely place to work. I would always cherish the time that I had spent with them.

I take the occasion to express my acknowledgement to all instructors, providing their valuable advice and encouragement throughout my post graduation. I am grateful to all my friends, who have cooperated me during the course.

Abstract

Based on an object-oriented modeling approach, a database system called **BODHI** (**Bio-diversity Object Database arcHItecture**)[1], intended to store plant bio-diversity information, is being developed completely around public-domain software. The unique feature of the Bodhi is that it integrates diverse types of data, including *taxonomic characteristics*, *spatial distributions*, and *genetic sequences*. Web enabled graphical user interface to this system is designed and implemented. The interface is of form-fill type and separate forms are designed for querying over each of the three diverse domains and one form to query over the entire domain. The interface is very intuitive and users can easily express their queries.

[1]Gautama Buddha gained enlightenment under the Bodhi tree.

Chapter 1

Introduction

1.1 Motivation

BODHI [9],[10] is specifically designed to cater to the needs of biodiversity applications. It is a *native* object-oriented system that naturally models the complex objects ranging from hierarchies to geometries to sequences that are intrinsic in bio-diversity domain. A variety of sophisticated indexing strategies are incorporated to efficiently access various types of data, and a rule-based query processor is employed for optimizing query execution. The DDL of Bodhi is adopted from Object Definition Language, specified by ODMG (Object Data Management Group). OQL is Query Language for Bodhi and it is a declarative query language standard proposed by ODMG and has been widely accepted as a query language for OODBMS. OQL can invoke methods written in the host language and any host language methods can include OQL queries.

Although the functionality that a system provides to users is important, the way in which it provides that functionality is just as important. It is imperative to have a good and web enabled visual interface to Bodhi both for querying and visualizing the results since

- the end-users are biologists and they don't know Object Query Language(**OQL**) [8] which is supported by Bodhi for data manipulation,

- it is essential to have a means of visualizing sequence data as well as multidimensional spatial data, and
- World Wide Web(WWW) is a good medium for information exchange.

1.2 Overview

Interface to Bodhi is designed and built which can be used to query the database server and view the results in an easy manner. Use of form fill-in interface is the traditional way of querying and so we developed form fill-in interface to our system. The two components of interface module are GUI-Client and GUI-Server. Web browsers can display HTML forms[7] which allow users to enter data. GUI-client comprises of HTML forms which contain control elements (checkboxes, radiobuttons, menus, etc.), and labels on those controls. It allows users to express their needs in an easy manner. GUI-server comprises of gateway scripts and interacts with query processor of Bodhi. Common Gateway Interface(CGI)[3] is a standard whose specification defines a way for Web servers to communicate with external programs, and vice versa, so that the external program can generate HTML, images, or whatever, and have the server treat it the same as HTML, images, and so on not generated by an external program. We are using Apache web server as HTTP server and CGI scripts are written in Perl.

The users of Bodhi may be interested to explore data pertaining to only one of three domains –taxonomy, sequence and spatial or their query can be constrained by data over the three domains. We provide a query form each corresponding to one of the three domains and another for the entire data of the system. Besides we developed one simple search form which facilitates the search for an object in a specific *extent* (extent of a type is the set of all instances of the type) and another form which supports queries like:

- Query1 *select all Shrub Species that share a part of their habitats and have a Chromosomal DNA sequence score of over 70 with Magnolia-champa .*

In Chapter 2, architecture of the interface module is explained, in Chapters 3, 4 and 5 taxonomy, spatial, and sequence data and the corresponding query form is explained respectively, Chapter 6 discusses the the other two search forms namely simple object search and complex search and Chapters 7 describes the related work and conclusions.

Chapter 2

Architecture

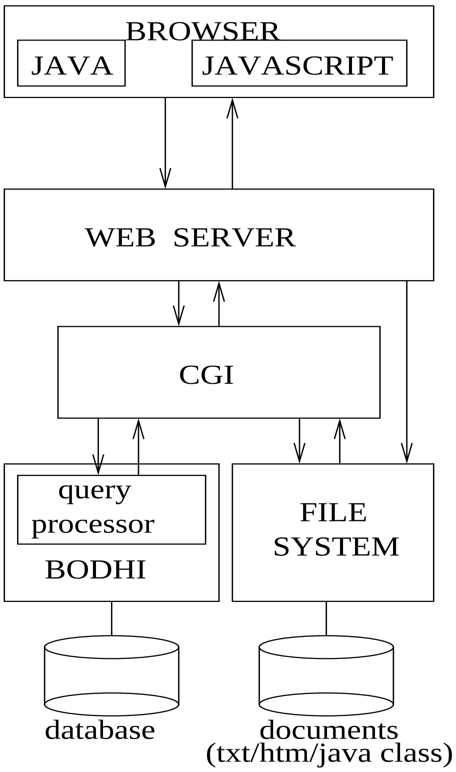

Figure 2.1: Architecture

The architecture of the interface module is shown in Figure 1. Web Server serves the user's request of query forms. User's browsers should be Java and Java Script enabled because spatial data viewer is implemented as Java Applet and Java Script is used to validate user's request and to convert the request into OQL at client-side itself. Next the web server executes CGI script on behalf of user. Scripts are written in Perl. CGI scripts mostly control the interaction between browser and database, while Java Script effectively controls the interaction between user and browser content. The OQL query is sent to CGI using POST method. User's can request the output either in HTML or XML[22] format. The CGI scripts interact with query processor of Bodhi and the results whose contents are tagged are outputted. If HTML output is requested, XML Parser written in Perl parses the tagged results and generates HTML output. We are using the behaviour of the parser controlled by **Start** and **Char** handlers to generate the output in the form of table. If the result is requested in XML format, the output of the query processor will be directly sent to the user. The reason behind providing the option to request the results in XML is that the requested data can be easily exported to other similar repositories as the structural relationships that exist inside the data can be embedded into XML output. The current state of information exchange amongst various bio-diversity data repositories is not very satisfactory. However, with the advent of XML, many research groups are proposing Document Type Descriptors(**DTD**s) in individual fields of ecology and genetics[2],[11].

Chapter 3

Taxonomy Data

Taxonomy data is about the relationships(*phenetic* and *phylogenetic*) between plant species. Partial schema of the Taxonomy data is shown in Figure 2. In the query form every *attribute* of each *extent* is listed in the database. During query formation, the users can click the corresponding checkbox in order to see the value of this attribute in the result, or they can specify the predicate of the query by clicking one or more listed values of this attribute or can do both. For some attributes (e.g. leaf axis type) which can have limited possible values in the system are listed while the attributes (e.g. species name) which can have huge possible range of values are provided with *text* controls so that users can type the values. Users can type one or more possible values. To illustrate, consider the following query template:

- *Give the names of the* **order**, **family**, *and* **genera** *of a Magnolia-champa*

This query can be expressed by clicking the corresponding checkboxes of **order**, **family**, and **genera** attributes and the species name will be specified in the corresponding input text. The name of the species will be in the output if the corresponding check box is also clicked. Taxonomy query form is shown in partial in Figure 3. By default the output is HTML type and the output is shown in table. The nice thing about our query form is that one form is sufficient to express all most all queries.

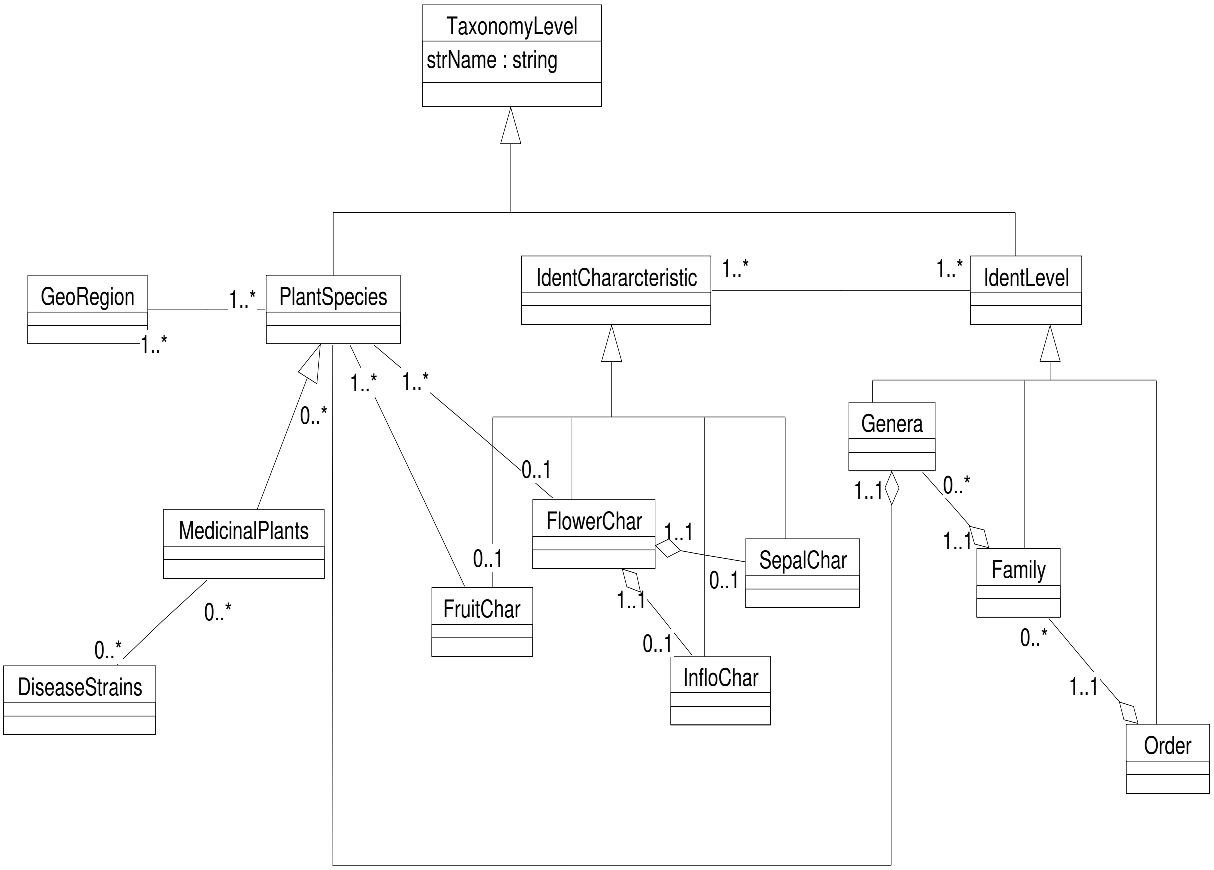

Figure 3.1: Partial Schema of Taxonomy Data

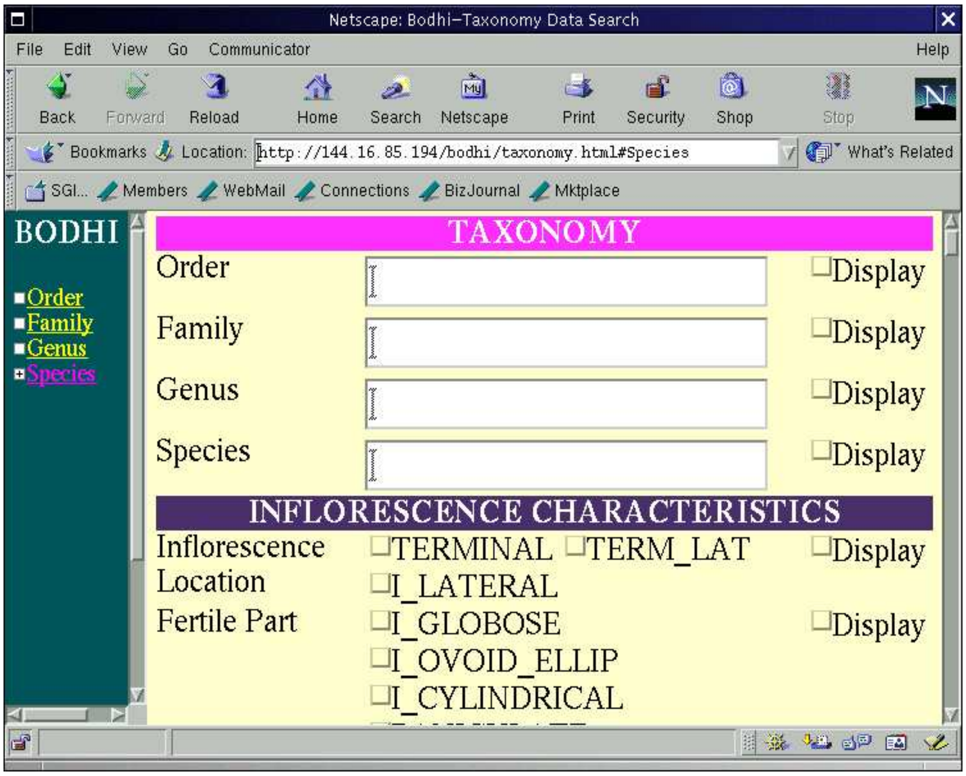

Figure 3.2: Taxonomy Query Form

Chapter 4

Spatial Data

Spatial data or geographic data forms a key component of a bio-diversity data repository. Bodhi provides a set of spatial data primitives (*point, line, polygon, collection of polygons, polyline*) to represent single spatial objects like country, state, forest, river etc., as well as to represent spatially interrelated collection of objects. Schema of the spatial data is shown in Figure 4.

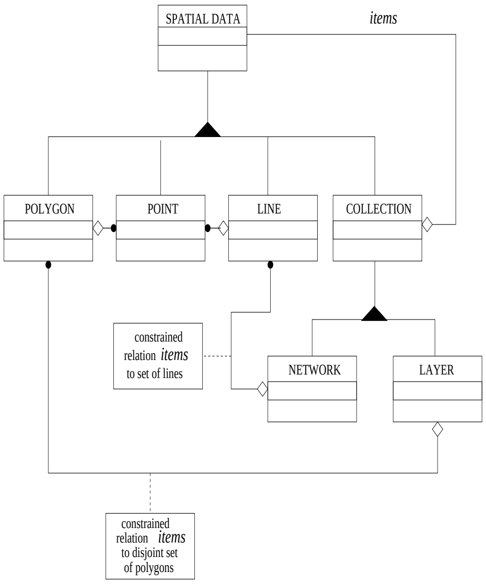

Figure 4.1: Spatial Data Schema

GIS(Geographic Information System) Viewer[5] developed at University of California, Berkeley is modified to suit our needs and used as our Spatial Data Viewer. Actually the viewer can also be used to edit the maps. We removed editing facilities of the viewer. The viewer is made to communicate with HTML FORM elements. This is the new feature that is added to. This is required to express the queries. Another facility that is added is that users can select regions bounded by rectangles(window query). For example users may be interested to know all plant species that are grown in some particular rectangular region. The viewer is implemented as a Parameterized Applet. A screen-shot of the viewer is shown in Figure 5.

A set of geographic information is called *layer*. A layer might be set of points or vectors that have some geographic meaning, e.g. set of cities in India. The viewer provides the users with a window over a set of layers, which are displayed one on top of other. Layers can be turned on or off, window can be moved around the layers (i.e.,"pan"), and "zoom" in and out are permitted. The viewer applet is made to communicate with HTML FORM elements. The viewer has the components–Action Bar on top, Layer Bar on left, Main View in the middle and Behavior View on the right of the applet window. Main View of the data can be panned using using scrollbars or via the OverView window. It can be zoomed using the Zoom interface. It calls on the various layers to repaint themselves and relies on them being painted. In the Behavior Bar, the Overview window shows the region currently visible in the MainView window in relation to the overall coverage. It calls on the various layers to repaint themselves and relies on them being painted. It tries to cache much of the repaint except when the layer visibility changes.

Consider the following query template:

- *select all* **Species** *that are grown in a particular* **forest**.

The above query can be expressed as follows. The checkbox corresponding to species is clicked. In the viewer the Forests layer is turned on and the name of the forest is selected using mouse. When the forest name is selected, the corresponding control name in HTML form will get the selected value assigned. Some more queries:

1. *select all* **Forests** *with a given* **landuse** *type*.

2. *select all* **Species** *with a given* **geo-region**. This is a multi-domain query.

The viewer also allow the users to view the data in one of the following formats.

Latitude/Longitude(LL) Latitude is the angle at the center of the planet between the plane of the Equator and a line through the center passing through the surface at the point in question. Latitude is positive in the Northern Hemisphere, reaching a limit of +90 at the North Pole, and negative in the Southern Hemisphere, reaching a limit of -90 at the South Pole. Lines of constant latitude are called *parallels.*Longitude is the angle at the center of the planet between two planes passing through the center and perpendicular to the plane of the Equator. Lines of constant longitude are called meridians.

Albers Equal Area(AEA) This is an equal-area projection. Scale is true along the one or two selected standard parallels. Scale is constant along any parallel. The scale factor of a meridian at any given point is the reciprocal of that along the parallel to preserve equal-area. This projection is free of distortion along the standard parallels. Distortion is constant along any other parallel. This projection is neither conformal nor equidistant.

Universal Transverse Mercator(UTM) The UTM projection divides the world into a regular grid of quadrangles, each 8 by 6 degrees in extent, called zones. Each zone uses formulas for a transverse version of the Mercator projection with projection and ellipsoid parameters designed to limit distortion. The UTM projection is defined between 80 degrees south and 84 degrees north. Outside these limits, the Universal Polar Stereographic (UPS) projection applies. The UPS has two zones, north and south, which also have special projection and ellipsoid parameters.

4.1 Implementation Details

Java 2D Graphics are used to draw Points and Lines. First the geographic world coordinates are translated into view port coordinates using the appropriate Projection.

Liang-Barsky [16] algorithm is used for clipping. The following briefly describes some of the main classes and interfaces in the Viewer.

viewer.ui.Viewer This is the Parameterized Applet and acts as the “environment” and is the interface between the data and User Interface(UI) side of the program.

viewer.framework.VEEnvInterface This defines the “environment” within which the Viewer runs. The environment acts as a link between the data and user UI sides of the code.

viewer.ui.Controller The functional root of most of the interface.

viewer.ui.ViewContext This is functional root of the data side of the program.

viewer.ui.Help It provides support for displaying help.

viewer.ui.MainView The main viewing window for the GIS data. It can be panned by direct manipulation as well as using scrollbars or via the OverView window. It can be zoomed using the Zoom interface. It calls on the various layers to repaint themselves and relies on them being painted from lowest to highest in terms of priority ordering.

viewer.ui.OverView The over view window for the spatial data. It shows the region currently visible in the MainView window in relation to the overall coverage. It calls on the various layers to repaint themselves and relies on them being painted from lowest to highest in terms of priority ordering. It tries to cache much of the repaint except when the layer visibility changes.

viewer.framework.VEObjInterface The base definition for most of the data side of the program. It defines the way in which VEObjects are read from <parameter> definitions and how they are written back out.

viewer.framework.GISFileReader This class is used to parse and read in <parameter> entries from a URL to load in the configuration data.

viewer.framework.Layer This is base object type for most of the data. It implements viewer.framework.VEObjInterface. The primary interaction between the interface and data side is through the paint(), overview(), and visible() routines.

viewer.framework.View This object stores information about the starting viewpoint (location and scale) when a set of layers is first loaded.

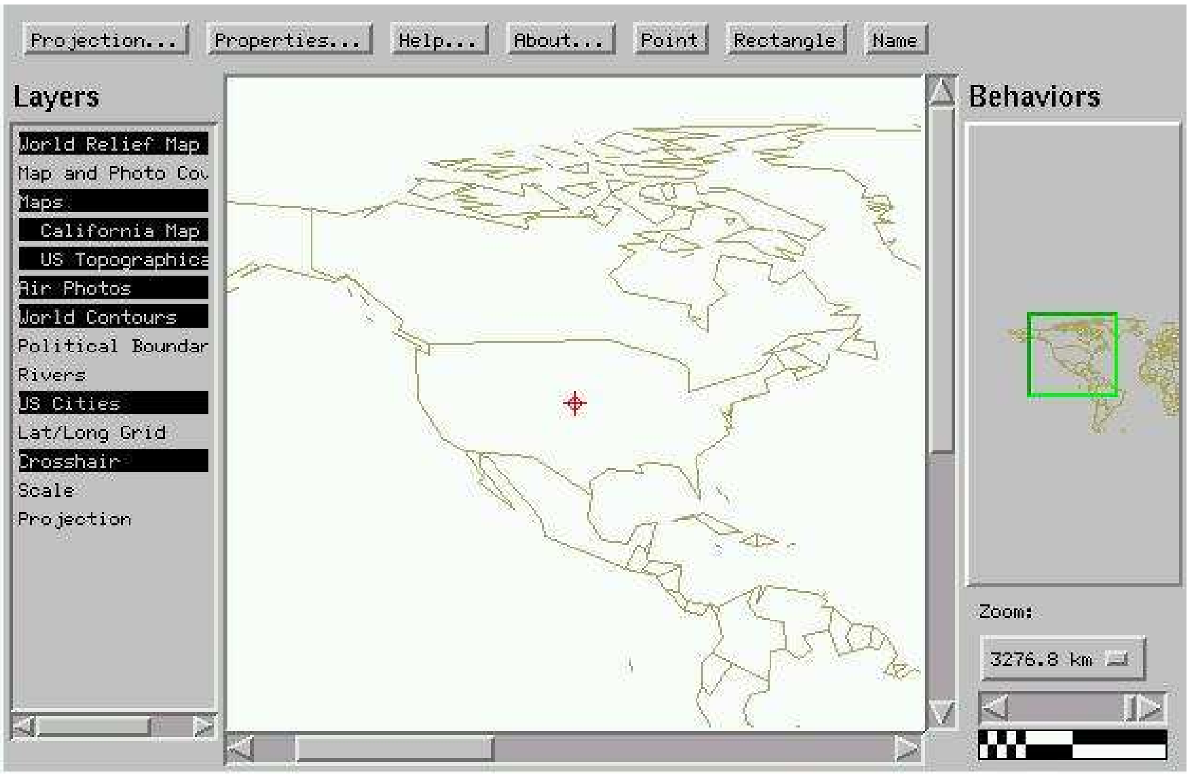

Figure 4.2: Spatial Data Viewer

Chapter 5

Sequence Data

The molecules that are of interest in bio-diversity are DNA and Proteins. These two sequences together form the genetic makeup of all species. DNA is represented as a long sequence based on a four nucleotide alphabet. Protein is a chain of simple molecules called *amino acids* and is responsible for various functions and properties that characterize the species. Protein sequences are represented as string of 20 alphabets. The data model provides *DNAStr* and *ProteinStr*, to represent the sequence information of segment of DNA molecule or Protein molecule respectively. Sequence data query form is shown in Figure 6.

Sequence Comparison is the most important operation in computational biology, and it forms the basis for many other complex manipulations. A fundamental concept in measuring similarity between two sequences is that of *alignment score* between two sequences. The most popular sequence similarity search algorithms are *BLAST* and *FastA*.Both are *heuristic* based algorithms, based on the notion of *Local Alignment*, using the alignment scores of segments of sequences, as a measure of similarity between two sequences involved in the search. Bodhi employs BLAST algorithm. Consider the following query template:

1. *List all DNA sequences in the database that are within a BLAST score of 70 with the given sequence.*

This query is expressed by filling or clicking the appropriate controls of the query form shown in Figure 6.

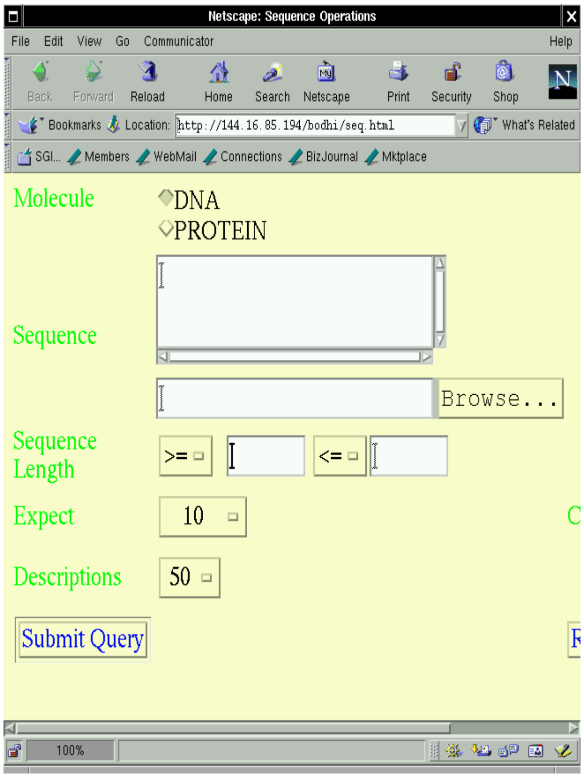

Figure 5.1: Sequence Query Form

5.1 Visualization of Output

The output of the above sequence similarity search programs can be several pages of text, and is difficult to analyze because of its length and complexity[21]. We used **MView**[17] for publishing the results of similarity search operations. The **EBI**(European Bioinformatics Institute)[12] operates a web service for BLAST2 and FASTA3, which uses MView, as do GeneQuiz and the GeneQuiz sequence submission services. MView is a tool for converting the results of a sequence database search (BLAST, FASTA, etc.) into the form of a colored multiple alignment of hits stacked against the query. It is implemented in Perl. A sample screen-shot output is shown in Figure 7. It provides several ways to color the alignments. The default layout is a single unbroken horizontal band of alignment. However, the users are provided with the option to break the alignment into vertically stacked panes. Cascading style sheets are added allowing more specific control of HTML elements. In particular, this enables selective coloring of text fore/backgrounds allowing alignments to use colored blocks instead of just colored lettering. Percent identities reported in each alignment row are calculated with respect to the reference sequence (usually the query or first row), as follows:

$$\frac{number\ of\ identical\ residues}{len\ of\ ungapped\ reference\ seq\ over\ aligned\ region} * 100$$

Multi-domain query form combines all the three forms explained above.

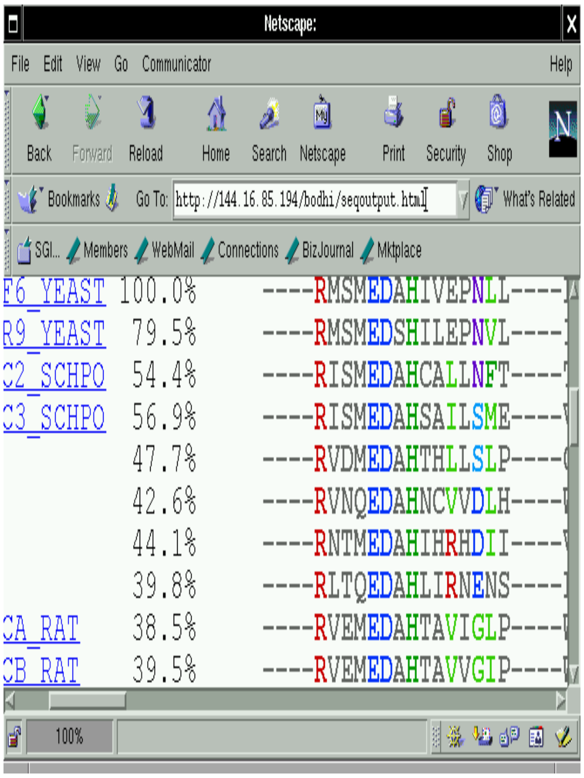

Figure 5.2: Screen-Shot of Sequence Output

Chapter 6

Query Forms

6.1 Simple Search

Querying using keywords is easily the most widely used form of querying today. We provide one simple search interface where users can enter the name of of object, and selects the type of extent it belongs to. This is a reasonable interface, and any casual user can understand it. The output of the simple object search is all data that is in the database related to the object queried in. Portion of the output is shown up as links. The Simple Object Interface form is shown in Figure 8. The results for the search of the word 'Champaca' in Plant Species extent is shown in Figure 9.

Figure 6.1: Screen-Shot of Simple Search Interface

6.2 Complex Query Form

The Multi-Domain Query combines all the three forms of taxonomy, sequence and spatial data. This form is not sufficient to express Query 1 listed in Overview Section of Chapter

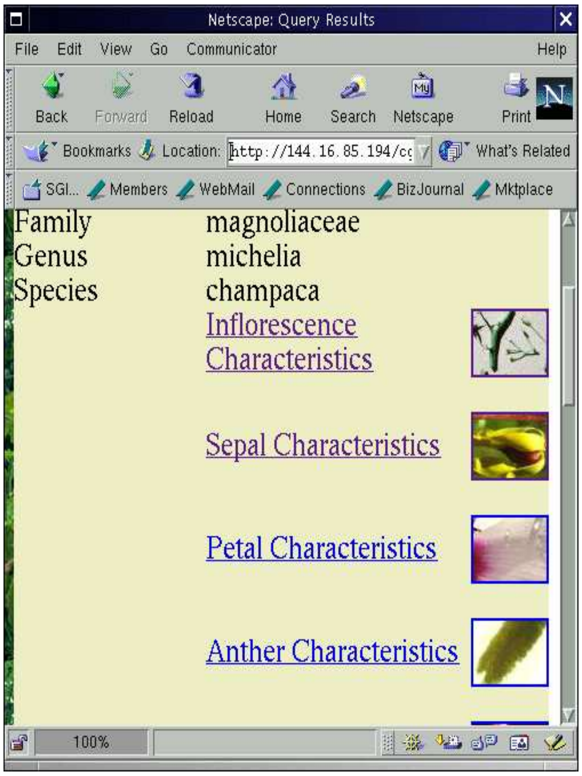

Figure 6.2: Screen-Shot of Simple Search Output

1. For these type of queries a self join across same extents is required. To support such quries we designed a query form as shown in Figure 10 to express.

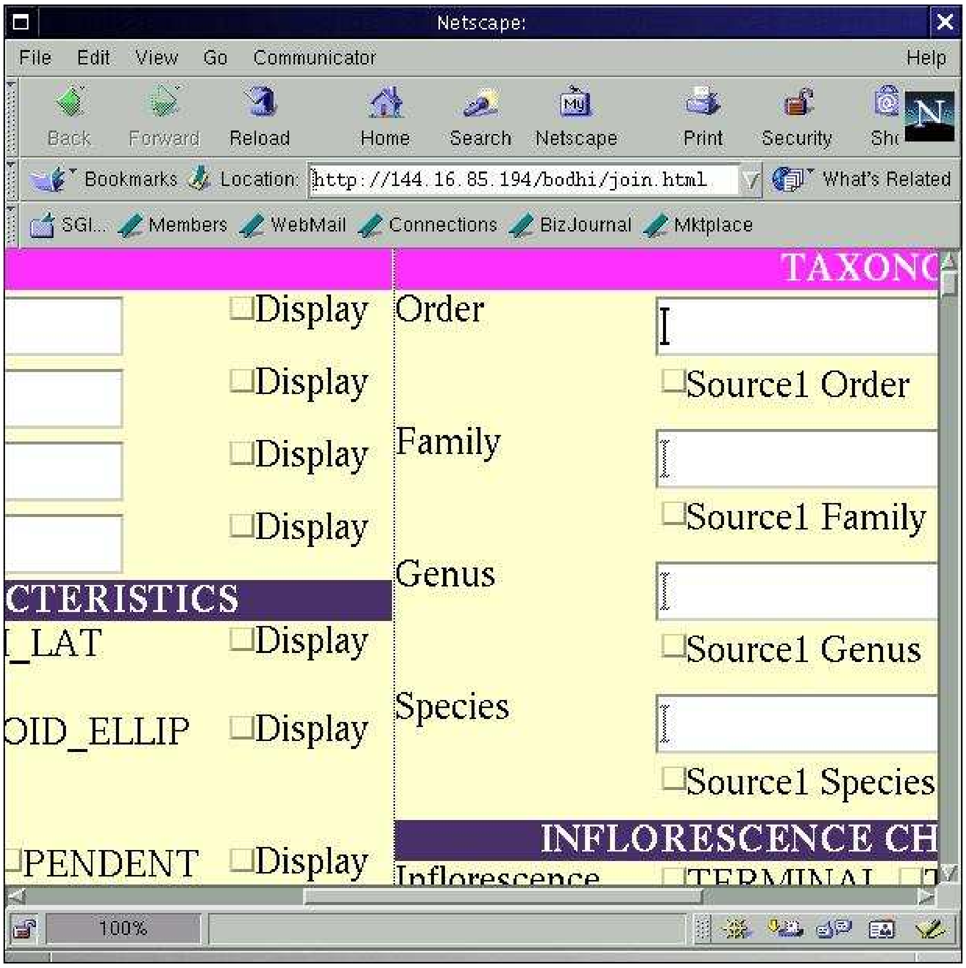

Figure 6.3: Screen-Shot of Complex Query Form

Chapter 7

Related Work and Conclusions

United States Department of Agriculture (USDA) maintains a database of plants and the database is accessible over the Net[20]. The database does not store molecular bio-data. But it stores the names of the states and country in which the plants are grown. But Bodhi stores the spatial data in vector form. The micro-level bio-diversity data, or genetic information of various species, has been growing steadily due to a multitude of genome sequencing initiatives. The specific data management issues in handling such data[15, 14] have been addressed in quite a few proposals. TIGR(The Institute for Genomic Research) is making a number of software tools freely available to the scientific community. Gene Splicer, Trans Term, Glimmer, MUMer, TIGR Multiple Experiment Viewer(TMEV), TIGR Array Viewer are some among them. The entire list can be found at[19].The list of Web GIS and Interactive Mapping sites can be found at[6].

We used GIS Viewer developed at University of California as Spatial Data Viewer. We are using MView for publishing Sequence Data. The interface we developed is specific to schema. It is believed that schema will be hardly changed for this sort of information repository. Even the schema changes,very little needs to be changed. A generic interface frame work can be planned as a future work.

Bibliography

[1] S. Altschul, W. Gish, W. Miller, E. Myers and D. Lipman, A basic local alignment search tool, *Journal of Molecular Biology*, (215), 1990

[2] ANZMETA DTD Version1.1, http://www.erin.gov.au/database/metadata/anzmeta/anzmeta-1.1.html

[3] Common Gateway Interface, http://www.w3.org/CGI/

[4] Forms in HTML documents, http://www.w3.org/TR/REC-html40/interact/forms.html

[5] GIS Viewer, http://elib.cs.berkeley.edu/gis/

[6] http://sunsite.berkeley.edu/GIS/intergis.html

[7] HTML Home Page, http://www.w3.org/MarkUp/

[8] R. Cattel, *The Object Database Standard: ODMG-93*, Morgan-Kaufmann Publishers, 1994

[9] B. Srikanta and J. Haritsa, Design and Implementation of Bio-diversity Information Management System, *Proc. of 10th Intl. Conf. on Management of Data (COMAD), Pune, India, December 2000*

[10] B. Srikanta, J. Haritsa and S. Uday, The Building of BODHI, a Bio-diversity Database System, To appear in *Information Systems Journal*, 2002

[11] Biopolymer markup language - BIOML, http://www.bioml.com/

[12] European Bioinformatics Institute, http://www.ebi.ac.uk/

[13] T. Boston and D. Stockwell, Interactive species distribution reporting, mapping and modeling using the world wide web, In *Proc. of the 2nd Intl. WWW Conf.*, October 1994

[14] N. Goodman, S. Rozen and L. Stein, A glimpse at the DBMS challenges posed by the human genome project, ftp://genome.wi.mit.edu/pub/papers/Y1994/challenges.ps.Z

[15] N. Goodman, S. Rozen and L. Stein, Building a laboratory information system around a C++-based object oriented dbms, *Proc. of 20th Intl. Conf. on Very Large Databases*, 1994

[16] Y. Liang and B. Barsky, An Analysis and Algorithm for Polygon Clipping, *Communications of the ACM*, 26, November 1983

[17] N. Brown, C. Leroy and C. Sander, MView: A Web compatible database search or multiple alignment viewer, Bioinformatics. 14(4):380-381, 1998

[18] T. Brinkhoff, H. Kriegel and B. Seeger, Efficient processing of spatial joins using R-trees, In *Proc. of the ACM SIGMOD Intl. Conf. on Management of Data*, May 1993

[19] http://www.tigr.org/softlab/

[20] http://plants.usda.gov/cgi_bin/topics.cgi?earl=advquery/adv_query.html

[21] Visualization of Biological Sequence Similarity Search Results, http://www.cbc.umn.edu/VirtLibrary/Chi/vis95/vis95.html

[22] Extensible Markup Language, http://www.w3.org/TR/REC-xml