



Network Activity D  
-  
Developing and Maintaining Databases

Report D3.2.1

Detailed UI architecture study including html mock-up

Patricia KELBERT  
MNHN Paris – BGBM Berlin  
March 2006

## Table of contents

1	The new BioCASE / SYNTHESYS user interface (UI)	3
1.1	Query Tools	4
1.1.1	Query Screens	4
1.1.2	Query Results	4
1.1.3	Menu items	5
1.1.4	User's preferences and cookies	5
2	Issues	6
2.1	Internationalisation	6
2.2	Development in Python	6
2.3	Thesaurus	9

## Table of figures

Figure 1-1	General sitemap	3
Figure 2-1	Code extract from kid template	7
Figure 2-2	HTML generated code	7
Figure 2-3	HTML skeleton	8
Figure 2-4	Thesaurus – UI overview	9

## Glossary

**Cookie:** An HTTP cookie is a packet of information sent by a server to a World Wide Web browser and then sent back by the browser each time it accesses that server. HTTP cookies are used for user authentication, user tracing, and maintaining user-specific information.

**W3C:** World Wide Web Consortium: an international consortium where member organisations work together to develop standards for the World Wide Web.

**UTF-8:** 8-bit Unicode Transformation Format

**Template:** a page, which can be inserted into another page via a process called transclusion (capability for documents to include sections of other documents by reference).

**Python:** an interpreted, interactive, object-oriented programming language.

**XML:** Extensible Markup Language - a simple, very flexible text format derived from SGML

**MVC:** Model-View-Controller: an architecture that separates an application's data model, user interface, and control logic into three distinct components so that modifications to one component can be made with minimal impact to the others.

**JavaScript:** a scripting programming languages, mostly used to write functions that are embedded in or included from HTML pages and interact with the Document Object Model (DOM) of the page to perform tasks not possible in HTML alone. For example, opening pop-up, validating web form...

# 1 The new BioCASE / SYNTHESYS user interface (UI)

The BioCASE UI provides access to two different kinds of data:

- Collection metadata from the CORM database (DB), a daily updated cache of the connected BioCASE National Node databases
- Unit level, initially from the SYNTHESYS Cache that is derived from the GBIF Index, eventually from the data providers in the GBIF system.

The new site map can be represented as follow:

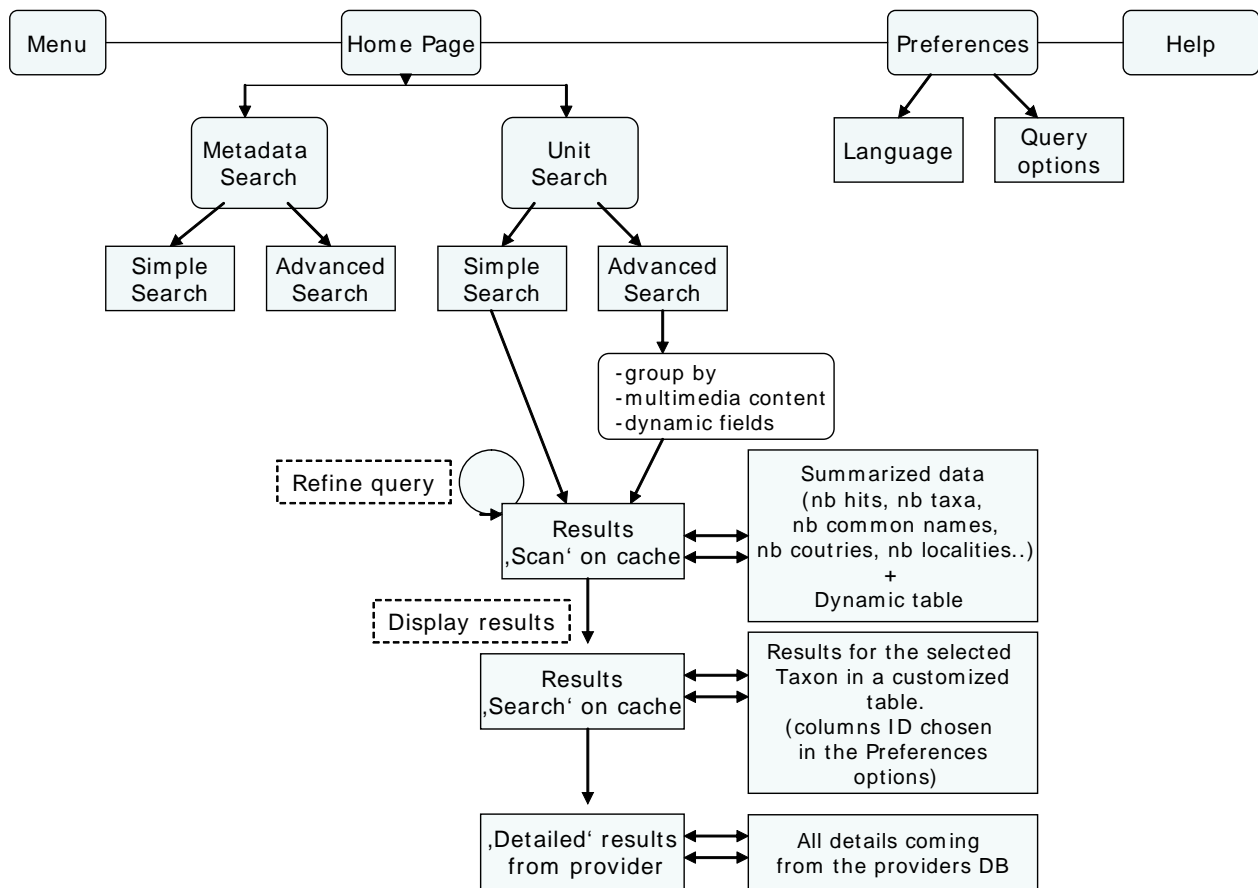


Figure 1-1 General sitemap

## 1.1 Query Tools

### 1.1.1 Query Screens

The user can access the query tools from the Home Page:

- He can directly perform a simple search (i.e.: enter a scientific name, which will be searched in the Specimens & Observations DBs, or enter a keyword that will be searched in the Institutions & Collections DBs).
- He can follow a link to perform an advanced search.

For the metadata advanced search, the portal V.1 will be used (with the new design)

For the units' level advanced search, the query screen is dynamic: the HTML form with text fields and select list is dynamically updated according to the user's choices.

In this report, only the properties of the "Units level search" will be discussed, which have been implemented within the SYNTHESYS project's Networking Activity D.

### 1.1.2 Query Results

Query results are displayed in three steps:

#### 1. Results from the scan of the cache :

- Data statistics:
  - Number of hits
  - Number of distinct scientific names (= taxon names)
  - Number of distinct common names
  - Number of distinct genera
  - Number of distinct countries
  - Number of distinct localities
  - Number of records of a certain type ("Record basis": Fossils, Specimen, Observations...)

A link in the statistics as well as a tabbed interface allows selecting a list of the items (taxon names etc.), from which one or more entries can be tagged to further narrowing down the query. In the result lists, individual items can be chosen to display the full detail the provider offers.

#### 2. Results from the search on the cache :

- Data's statistics:
  - Number of hits
  - Number of hits with multimedia content
  - Link to the map's illustration for the geographical distribution of the results
- Table with data corresponding to the selected item(s). The table columns that have to be displayed can be chosen in the user's preferences.
- Link to download the table

Users can select a single item from the table, and access detailed information describing these data.

In case users want to retrieve the details for the whole result list, a confirmation's page is displayed to warn that it might take a lot of time and resources.

If the choice is confirmed by the user, an archive (.gzip with XML and XSL files) will be generated. The URL where the archive will be available is unique and is displayed. The user can choose to get an email with the URL as a reminder when the job is done.

3. Results from the provider
  - Full details display
  - Link to download the XML

### **1.1.3 Menu items**

The menu has two levels that can be expanded and collapsed. It provides access to the query tools pages ('Search'), the registry information ('Registry') and general information ('About'). JavaScript is used for dynamic UI updates.

In order to fit to the BioCAsE, SYNTHESYS and BioCASE specifications, information about the BioCASE project will be moved to this search portal.

### **1.1.4 User's preferences and cookies**

A lot of information can be accessed through the cache interrogation, for example the unit's properties (ID, Scientific Name, Genus, Country, Location, Collection date, Collector, Institution...). Some of these data might not be relevant for some users; then the default display will only contain the taxon name (scientific name), country and the institution.

A cookie will also remember if the user prefers to access the Simple Search Query tool or the Advanced Search Query Tool.

Users can select other fields in their Preferences page, and they can choose a preferred language. The preferences are stored in a cookie, which enables the web site to switch to the user's preferred language.

A mechanism of browser recognition for the user's linguistic environment can be used to automatically load the portal in the current language.

Regarding the objective of internationalization, some rules have been respected.

## 2 Issues

### 2.1 Internationalisation

The internationalisation rules have already been defined in the NAD 3.1.1 report.

The layout and styles of the UI have been written by taking into account the internationalisation issues:

- all font properties (family, size, colour...) are described in the CSS file, which can be customized for each language if needed.
- according to the W3C, UTF-8 is used for the character encoding

For each web page and for each available language, templates will be created.

A script will make it possible to automatically generate new files for new languages. This script will take as entry two types of data:

- the list of templates with variable names
- a properties file, containing couples of values:

variable name	=	value in specific language
variable name2	=	value in specific language
...	=	...

and will return the list of templates with the variables replaced by their values in the specific language.

These templates will be interpreted by a Python application and then displayed as an HTML file.

### 2.2 Development in Python

The HTML pages will be generated with CherryPy.

CherryPy is a Pythonic, object-oriented web development framework, which allows developers to build web applications in much the same way they would build any other object-oriented Python program.

CherryPy can work with several templating packages, such as the Kid Template package.

Kid is a simple template language for XML based vocabularies written in Python.

Some parts of the HTML can be dynamically updated, through Python methods, such as the navigation trail (breadcrumbs).

For example, in the master template (which defines body, page, header, trail, menu and footer), Python code can be inserted as follow:

```
<ul>
  <li>
    <a href="/index" title="home page">Home</a>
  </li>
  <p py:for="title, href in navlist" py:strip="">
  <li>&#187;
    <a py:content="title" href="{href}">text comes here </a>
  </li>
</p>
</ul>
```

**Figure 2-1 Code extract from kid template**

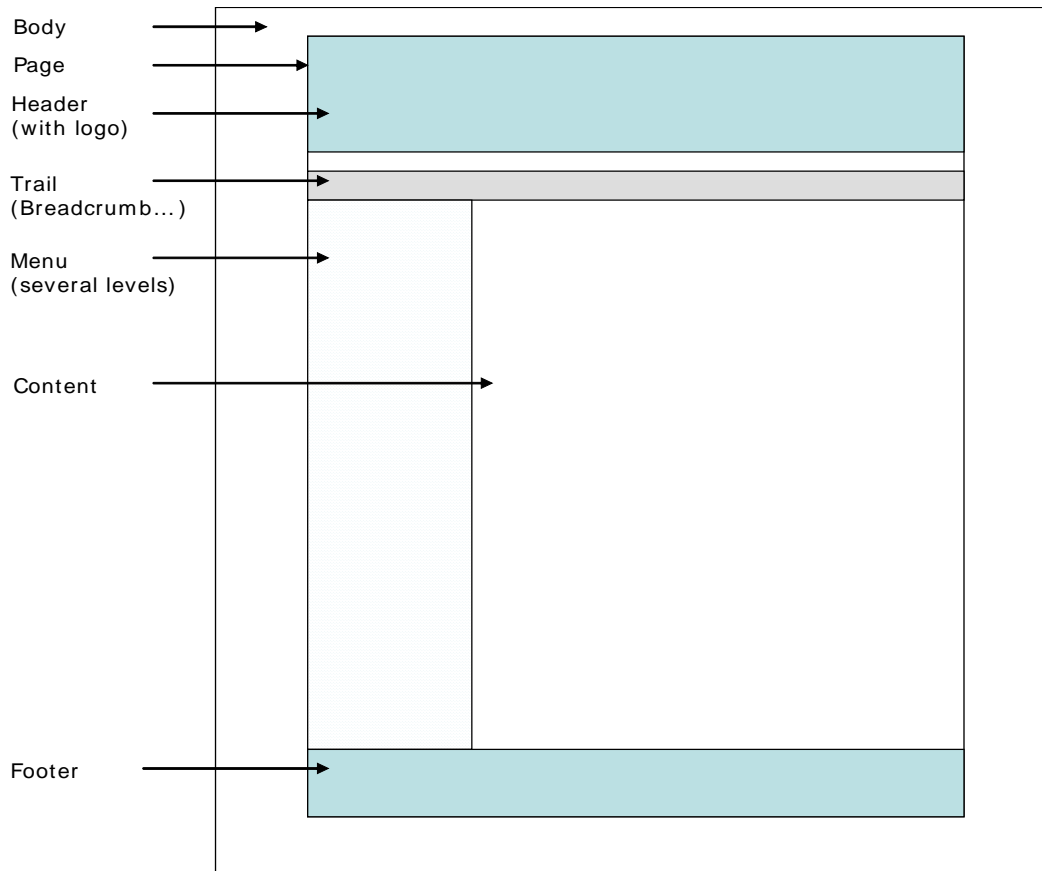
If the user is on the Units level search, the navigation trail is Home >> Search >> Units. The master template and CherryPy will produce the HTML code:

```
<ul>
  <li>
    <a href="/index" title="home page">Home</a>
  </li>

  <p>
  <li>&#187;
    <a href="" title="Search"> Search</a>
  </li>
  <li>&#187;
    <a href="units.htm" title="Units"> Units</a>
  </li>
</p>
</ul>
```

**Figure 2-2 HTML generated code**

Thanks to this template system, a skeleton can be defined, and all the other files extend it. If a developer wants to change for example the logo, he will only have to modify the skeleton file and the new design will then be applied to all the pages.



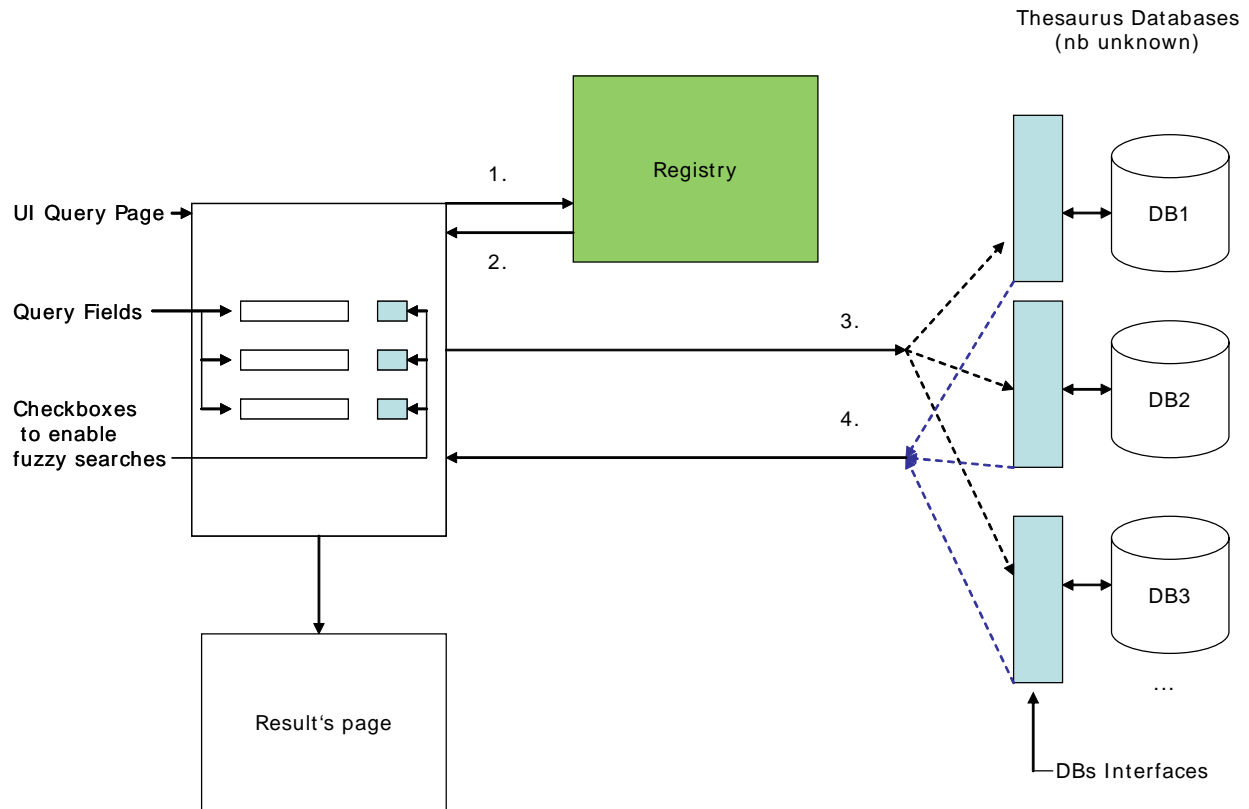
**Figure 2-3 HTML skeleton**

The User Interface MVC:

- HTML in browsers = the view
- events (sent and received) = the controller
- Python files = the model

## 2.3 Thesaurus

A thesaurus interface will be developed at a later stage, and will respect the W3C MVC. Requests on the Thesaurus DBs will be performed according to the following schema:



**Figure 2-4 Thesaurus – UI overview**

1. `getThesaurusCapabilities(field)`: go to the registry and ask for the Thesaurus that are relevant for a fuzzy search on the current query field
2. return the list of corresponding Thesaurus DBs with their properties to the UI Model
3. `getFuzzy(name)` (=getHigherConcept (name), getSynonym(name)...): go to the specific DB (either DB1, or DB2 or DB3) and ask for the higher concepts (synonyms...) for the current name entered by the user
4. return the results to the UI Model, which performs the search with the "fuzzy words" and displays the results in the UI View

### D3.2.1 - Detailed UI architecture study including html mock-up – March 2006

The registry will contain the list of Thesaurus DBs, with their capabilities: IP, login, password, concerned fields (i.e. DB for Countries, DB for Taxonomic names...), available queries (getHigherConcept, getSynonym...)

The Thesaurus DBs will have to implement a BioCASE/SYNTHESYS Interface, which will return the capabilities and make it possible to query the different DBs through generic queries.

The interface components informing the user about the query extension used are under discussion.

The HTML mock-ups (optimized for resolutions from 800x600 to 1024x768) are available under <http://ww2.biocase.org/synth-gui/GUI-DESIGN/TAO> .

- about.htm
- acknowledgement.htm
- advancedSearch.htm
- bioCASE.htm
- bioCASE\_and\_GBIF.htm
- confirmation.htm
- hosts.htm
- index.htm
- institutions.htm
- itineraries.htm
- metadata.htm
- preferences.htm
- result1.htm
- result2.htm
- result3.htm
- search.htm
- simpleSearch.htm
- unit.htm
- validation.htm
- static/
  - o css
  - o js
  - o images
  - o xml
- jsp\_html/
  - o Browse.jsp.htm
  - o MetadataAdvancedSearch.jsp.htm
  - o MetadataBasicSearch.jsp.htm