European Cultural Heritage Online

ECHO

PUBLIC

Contract n°: HPSE / 2002 / 00137

Title: NECEP (Non European Components of European Patrimony)

This document includes the following features:

- **the NECEP prototype specifications (report D2.1)**
- **technical realization (report D2.2)**

Author(s) : Laurent Dousset

Concerned WPs : WP2

Abstract:

This document describes the data structure specifications, the technological prerequisites and the technological implementation of the NECEP protoype. Additionally, it includes instructions for content providers wishing to add or modify data in the prototype.

Published in : (see http://www.ehess.fr/centres/logis/necep/)

Keywords: metadata, sql, php, xml, prototype, ethnology, anthropology

Date of issue of this report: 30.10.2003

Project financed within the Key Action
Improving the Socio-economic Knowledge Base

WP2 Deliverable D2.1 and D2.2
specification Report / technical realization

NECEP

(Non European Components of European Patrimony)

This document includes the following features:

- the NECEP prototype specifications (report D2.1)
- technical realization (report D2.2)

version 2.2

Laurent Dousset
Paris
31.10.2003

Content

1. Introduction

The NECEP (Non-European Component of European Patrimony) project is about the constitution of a database, website and xml-files repository and meta-data set on social groups or societies in relation to their material, as well as immaterial culture, including social and cultural data, descriptive articles, photographs, 3D QuickTime objects, sound-files and movies. Its aim is to contribute to a better understanding and illustration of the objects and online material held by European Museums on non-European societies and to make these available to researchers and the public.

The proto-type, which is accessible online at http://www.ehess.fr/centres/logis/necep/, contains yet a limited number of societies through which the working-principles are illustrated. The NECEP content is meant to grow on the basis of the available proto-type through the contribution of specialists on each society, such as linguists, anthropologists and historians.

The NECEP project is rather particular within the ECHO project because DTDs in ethnology had to be entirely defined and could not be constituted through the accumulation and adaptation of external specifications. Moreover, the content produced can be considered “meta-data” itself (meta-data on societies already is interpreted information). In this sense, the NECEP proto-type is an expandable *annotation* of material held by institutions, such as museums.

It follows that interoperability issues between ECHO current and potentially future partners are vitally important for NECEP to be fully operational and useful. Important interoperability frameworks with MPI-Nijmegen and the RMV museum have been created.

2. Discussion Process

Many discussions took place to define the metadata structure necessary for an anthropological description of a society. These discussions included the primary content provider team headed by Roberte Hamayon, but also Maurice Godelier and Laurent Dousset in a first step. In a second step, other partners such as RMV were consulted to include elements of their metadata structure in the consideration of a NECEP proto-type database structure.

It is not possible to actually list the number of days/dates on which such discussions took place. On average, it is safe to state that 10 hours discussions a month including the principal participants, and everyday discussions between individual members, reflects an accurate estimate of the amount of interaction that took place for the creation of the NECEP proto-type.

In addition, at least one member of the French NECEP team participated in each major ECHO event and workshop to exchange information and obtain advice. Laurent Dousset also spent 3 days in Nijmegen (MPI, Peter Wittenburg et al.) discussing pragmatic questions of interoperability and integration of the proto-type within ECHO (WP2) and its common search engine (see Peter Wittenburg's Specification Report WP 2.1).

3. Technological Choices: TAMIDOU

The NECEP proto-type had to answer three interactive necessities:

1) A simple but controlled procedure for adding, editing and deleting content.
2) A mode for reproducing the content in a user-friendly and potentially attractive environment.
3) A coherent mode for reproducing the data in a machine-readable format for interoperability issues.

Moreover, the technological choices had to be cost-efficient (or costless) and quick to implement, for the entire NECEP proto-type and its illustrative content had to be provided within the set timeframe from scratch.

The choices can hence be summarized as follows:

- Webserver: **Apache on Linux** open source / costless
- Database system: **MySQL** open source / costless
- Scripting language for interaction with webserver: **PHP** open source / costless

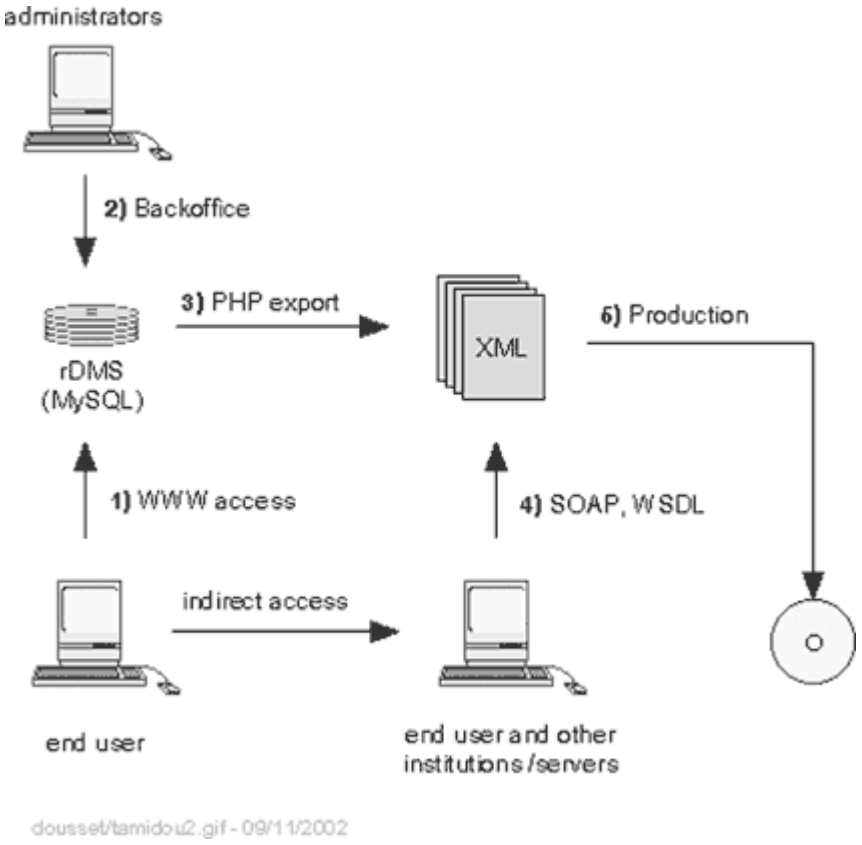


Figure 1: Shorthand illustration of the implementation of the NECEP proto-type

The content is stored in a series of tables of a MySQL database (see below in this report for more information on the structure of this database). Authorized content providers may edit the database through what is called the Backoffice (nr 2 in Figure 1). This backoffice is a simple

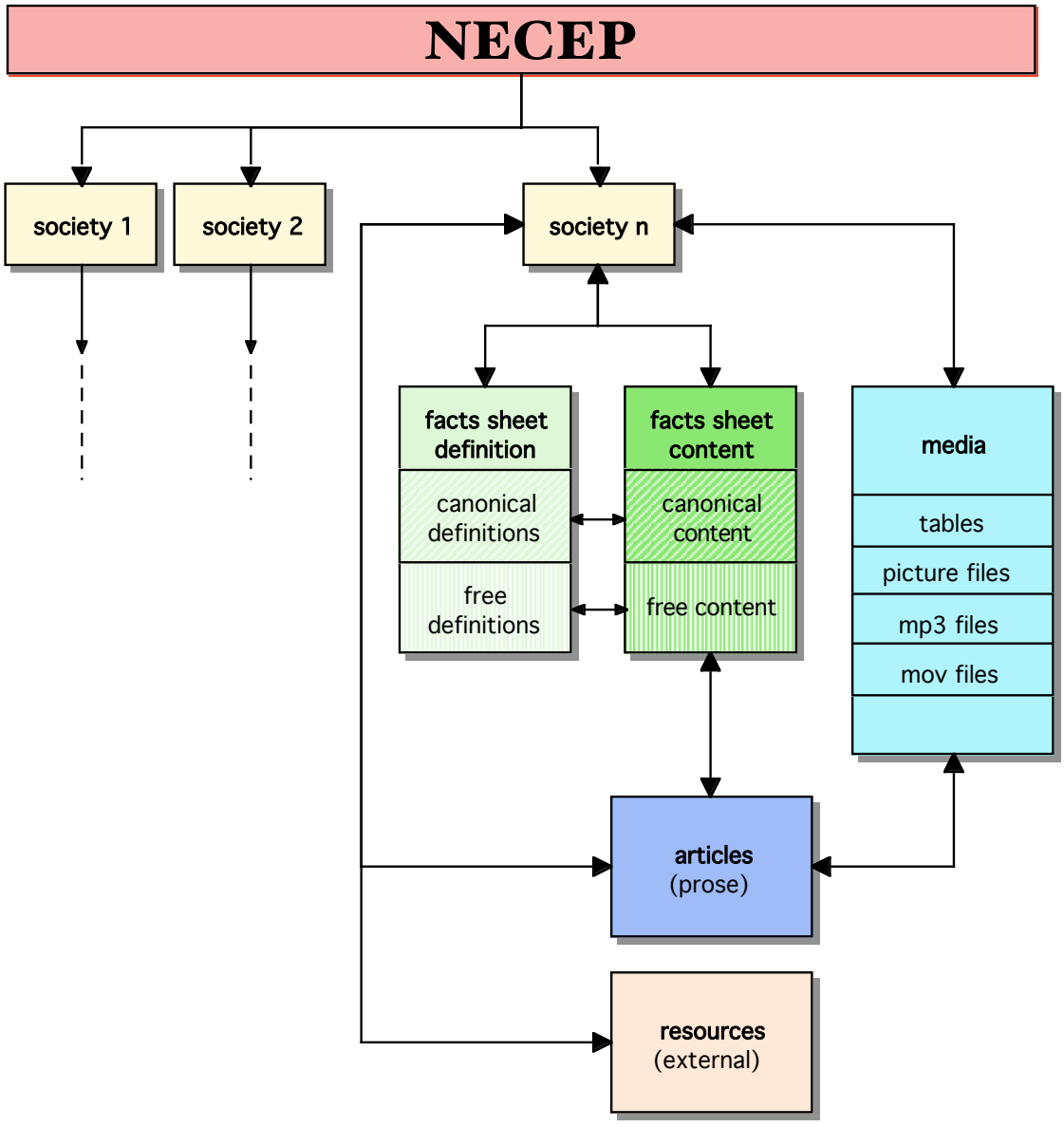
web interface supporting multiple simultaneous users and prepared in PHP. Access to the backoffice is through user/password authentication that have been provided by the System Administrator, currently Laurent Dousset. The content is reproduced within templates that can easily be modified and are sent to the webserver (nr 1 in figure 1 above). The relation between the templates and the database is provided by PHP scripts. The administrator has the authority to export the entire database structure to an XML directory system for interoperability issues (nr 3 in figure 1 above). This export/parsing is undertaken by PHP scripts as well (see below for more information on the XML directory system).

We have used as a working name for this type of implementation the appellation **TAMIDOU**.

The setup of the database, the templates and PHP scripts were undertaken by Laurent Dousset in about 900 working hours. New templates are being developed by Transversales, a company based in Marseille, to create a more aesthetic interface.

4. Logical structure of the NECEP database system

The database, its tables and its logical (relational) organization are as follows (please keep this figure in mind while reading this report):



Definitions:

society: for each society, the NECEP database contains a full record, or at least its empty framework, which can independently be accessed. This framework contains the facts sheet definition and a reserved identification number. This identification number (n) is handed over to all elements that relate to a specific society, such as media, articles etc.

facts sheet: the facts sheet is a collection of short information for a specific society. The facts sheet contains, among others, those elements that shall be used in interoperability issues, such as country, continent, language name, alternative names and spellings used. The facts sheet is divided into two parts. The first part is identical for all societies, that is, it contains a fixed and pre-defined number of questions for which, hopefully, content providers have produced a simple and short answer. The second part is free, that is, content providers are allowed to create their own questions and answer these. The facts sheet is also divided into themes such as social organization, material culture etc.
Moreover, the facts sheet is divided into two tables: one (definition) contains the questions, the second (content) contains the answers. Both are identified by the relevant society identification number.

articles: these are prose texts that are usually, but systematically, linked to facts sheet elements (questions-answers) to illustrate in prose some of the content provider's answers. Each article must be linked to a society, and to one society only.

media: these have illustrative purposes and are distinguished as illustrations/images (jpeg) sound files (mpeg3), movies (quicktime animations) etc. Each of these types of media is related to an independent database table with relevant information. The raw illustration files are available/retrieved from a predefined directory. Each media must be linked to an article and to a society.

tables: (not represented in the figure above) these have illustrative purposes and are linked to a society and an article.

resources: This is a simple collection of NECEP-external websites relevant for a society. Resources must be linked to a society.

5. Overview of software used to implement TAMIDOU

In order to satisfy the above-mentioned requirements, the following software has been implemented (and is required).

Platform	Description	Cost	Reason	Task
Operating System (OS)				
Linux	Open source OS that has proved to be powerful and reliable.	In principal *free*, however installation packages usually have a minimal cost involved	Ideal OS for server deployment; cheap	Operating System (This is the engine behind it all) (All scripts of this document have also been tested on Mac OSX v.10.1 and 10.2)
Server				
Apache	Open source server that has proved to be powerful and reliable. Many platforms available, but Linux platform probably ideal	Free	Reliable server (there are not many strong reasons why favour Apache over other webservers). The main advantage is probably its fast progress and bugfix, its openness to new protocols, techniques and languages and its Open Source philosophy.	Web Server (necessary software to be able to serve content/files over the WWW etc.). This is one of the engines between the outside world and the machine.
RDMS (relational database management system)				
MySQL	The widest used SQL type database (Structured Query Language).	*Free* (if no income produced with its direct use)	Free in principle. Fast, reliable, manages fulltexts. Disadvantage: no proper transaction management (PostgrSQL would in this respect be more appropriate, also because it can handle large objets and because it is entirely free. However, PostgreSQL has,	Store and manage data in such a way that management, update and modifications are flexible/possible. Filesystem type database. Supports many (all?) data formats.

Platform	Description	Cost	Reason	Task
			yet, no fulltext search capabilities (the psql website announces that work is being done on this, it is therefore necessary to stay informed).	
Scripting language				
PHP	Open source scripting language to handle interaction between the WWW and rDMS. Specially suited for WWW deployment but has also shell usage.	Free	There is no strong technical reason for favouring PHP over Python, Perl or Java etc. However, phpMyAdmin is probably a good reason for choosing PHP, as this offers an easy and free tool, even for the layman, for managing databases, tables and data. (I did not have access to phpMyAdmin on the server NECEP is hosted, as the connection allowed was only via SSH terminal)	1) Interface between browser and database. 2) Write content of rDBMS into pages that can be served by Apache (see above) to the WWW 3) Convert XML and DTD files into SQL compatible format and insert content of XML into rDBMS. 4) Convert content of rDBMS systematically or periodically into XML files. For further deployment (CD, DVD) and access by other servers/software (SOAP, WSDL)

6. Overview of the XML directory system and structure

The entire database content is mirrored into an XML directory system for interoperability issues and data exchange. The logical structure explained above is reproduced in directories as follows:

Societies: basic information and elements of identification are in: /xml/societies/ with as many files as there are society records. Each file is named using the identification number of the society (n) as /xml/societies/society_n.xml. This file also contains an index to all articles, tables and illustrations for this society.

Articles: all articles are stored at /xml/articles/ with as many files as there are articles. Each file is named using the identification number (n) of the article /xml/articles/article_n.xml. Each article file also contains a tag with the identification of the society itself.

Tables: all tables are stored at /xml/tables/ with as many files as there are tables. Each file is named using the identification number (n) of the table /xml/tables/table_n.xml. Each table file also contains a tag with the identification of the society and a tag with the identification of the article to which it is linked.

Media: meta-data for illustrations are stored in typified folders.
Illustrations (jpeg files) are in xml/media_metadata/images
Sound files (MPEG3) are in xml/media_ metadata /sounds
Each of these folders contains as many files as there are images, sounds etc. and are named using the identification number of the elements as /xml/media/images/image_n.xml etc. Each media meta-data xml file contains the identification number of the society and of the article to which it relates, as well as the absolute url to the raw/original media file.
The raw/original media files are not in the xml directory system, but are located at the root of the domain in:

/media/images/…
/media/sounds/…

For further information on XML file structures, check below the chapter XML management.

7. Function and modules list / summary (extract only)

Below is a general summary of some of the principal PHP functions used to handle and display the content of the database on the website. XML specific PHP functions are described in the *XML Management* part of this document.

```
CALL:        include ("db_connect.inc.php");
```
Establishes a generic connection to the MySQL database "echo". No other specific call needed. You have to change the connection variables in db_connect.inc.php to suit your settings (i.e. username and password)

```
CALL:        $content = glossary_parse($content);
REQUIRES:    include("glossary_parse.inc.php")
COMMENTS:    not recommended function, see add_glossary($content) instead
```
Returns a text ($content) that contains links to glossary.php?id=$word_id, where $word_id is the reference pointing to the glossary table

```
CALL:        add_glossary($id_article);
REQUIRES:    include("glossary_parse.inc.php")
COMMENTS:    this is the recommended glossary function
```
Creates an array of ids ($gloss_arr) of words found in the article with $id_article for which a link to a glossary needs to be established, calls a subfunction that does list the words for which there is a glossary entry with a link to glossary.php. Call this function at the place where you actually want to insert the glossary entries.

```
CALL:        insert_comment($article_id);
REQUIRES:    include("user_interaction/article_comments.inc.php")
COMMENTS:    displays user comments and form for user interaction
```
This function fetches user comments from the database relative to a specific article and displays the comments, as well as the form for adding new comments to the database/article.

```
CALL:        display_name_simple($id_soc)
REQUIRES:    include ("scripts/display_content.php");
COMMENTS:    displays the name of the handled social group
```

```
CALL:        display_name($id_soc, $font_face, $font_size, $font_color)
REQUIRES:    include ("scripts/display_content.php");
COMMENTS:    displays the formatted name of the handled social group
```
When calling this function, the designer can set font face, size and colour. Do not modify the variable $id_soc

```
CALL:        display_article_list($id_soc, $font_face, $font_size,
             $id_current_article)
REQUIRES:    include ("scripts/display_content.php");
COMMENTS:    displays the list of articles avaiblable for a social group
```
This function fetches all the article ids and title that are relevant for $id_soc and displays a list of the titles as a link to display them in articles.html

```
CALL:        display_article($id_soc, $id_article, $font_face, $font_size)
REQUIRES:    include ("scripts/display_content.php");
COMMENTS:    displays the article the user wanted to read
```
This function displays the article $id_article for the society $id_soc the user has choosen to read. If $id_article is zero, then the general facts page for this society is displayed. The script also inserts a previous and next to read article link.

```
CALL:        insert_image($id_article, $max_width)
REQUIRES:    include ("scripts/display_content.php");
COMMENTS:    displays the images linked to the article displayed
```
Max_width sets the widht of the thumbnails. The height of the thumbnail is adapted proprtionnally. The thumbnail itself is a link to a pop-up window that is resized in accordance to the image's original size.

7. Database tables, SQL queries and PHP programs

Following is a detailed and discussed list of, and instructions for creating, the various modules of the proto-type application.

Each subchapter usually contains:

- Description: the general description of the module or sub-module.
- Benefits: this part summarizes the benefits of specific functions and justifies why they should be used.
- Limitations: inconveniences associated with the use of a module or a function
- Table structure and SQL commands: this part describes the basic table structure and lists the SQL syntax to be used in MySQL for creating the tables. The syntax can be copy-pasted from this document into either mysql –u <your log in> –p or into the phpMyAdmin SQL form.
- PHP functions: this parts describes and lists the php functions that create the interface between the SQL tables and the website. The entire code is not reproduced in this document, but is available in electronic form. However, the calls that are necessary for implementing the various modules and functions are mentioned without exception.
- Backoffice: This part mentions if the backoffice area for administering the SQL tables has been implemented with php. Again, a detailed list of the backoffice functions is not reproduced in order to keep this document as short as possible. For more information on the Backoffice area, check pages 45 ff.

CAUTION: the XML specific modules and functions are not described in this part of the document, but are condensed in the chapter XML management.

MySQL Database connection (general)

Description

This include-file establishes a generic connection with the MySQL database server and the database called “echo”. The connection script is stored in the file db_connect.inc.php.
Author: Laurent Dousset
Copyright: no copyright, this is a well known connection script

Limitations

No particular limitations. You need to modify your local settings in this file.

PHP structure

```
<?php
/*
FILE:           db_connect.inc.php
AUTHOR:         Laurent Dousset
PROJECT:        ECHO
```

```
DATE CREATED:   20 November 2002
DESCRIPTION:    Opens a connection to a MySQL database.
*/

// Modify the below variables in accordance to
// MySQL access privileges and names

    $host       = "localhost";
    $user       = "your_username";
    $password   = "your_password";
    $database_n = "echo";    // this is the name of your database

// Don't change anything below, unless you want to handle
// connection errors specifically

    $db = mysql_connect($host, $user, $password)
        or die ("Could not connect to database");
    mysql_select_db ($database_n, $db);
?>
```

To invoke the connection, simply include at the top of your webfiles the following line:

```
include(“db_connect.inc.php”);
```

Facts sheet display – main module

Description

The specification of facts sheet is a set of “rigid” database entries for each society. It does answer specifically simple questions such as “demography”, “country of residence”, “language name” etc. Each such answer is or can be linked to a more explicit article (see Article display module below).

Database table structure and SQL create commands

The specification sheet takes its resources from three tables. The first contains the names for the various fields/rows—and hence permits that a set of distinct terms can be associated to distinct societies—; the second table contains the “answers” to the questions/terms defined in the first table; the third table is a reference to links, internal and external, that shall explain or provide more details to the answers given in table 2.
Be aware that these are preliminary database table structures.

Table 1: echo_facts_terms

This table contains the term-definitions, and is created as follows.

```
CREATE TABLE echo_facts_terms (
  id int(11) NOT NULL auto_increment,
  author varchar(250) NOT NULL default '',
  A1 varchar(250) NOT NULL default '',
  A2 varchar(250) NOT NULL default '',
  A3 varchar(250) NOT NULL default '',
  A4 varchar(250) NOT NULL default '',
  A5 varchar(250) NOT NULL default '',
  A6 varchar(250) NOT NULL default '',
  A7 varchar(250) NOT NULL default '',
  A8 varchar(250) NOT NULL default '',
  B1 varchar(250) NOT NULL default '',
  B2 varchar(250) NOT NULL default '',
  B3 varchar(250) NOT NULL default '',
  B4 varchar(250) NOT NULL default '',
  B5 varchar(250) NOT NULL default '',
  B6 varchar(250) NOT NULL default '',
  B7 varchar(250) NOT NULL default '',
  B8 varchar(250) NOT NULL default '',
  B9 varchar(250) NOT NULL default '',
  B10 varchar(250) NOT NULL default '',
  C1 varchar(250) NOT NULL default '',
  C2 varchar(250) NOT NULL default '',
  C3 varchar(250) NOT NULL default '',
  C4 varchar(250) NOT NULL default '',
  C5 varchar(250) NOT NULL default '',
  C6 varchar(250) NOT NULL default '',
  C7 varchar(250) NOT NULL default '',
  C8 varchar(250) NOT NULL default '',
  C9 varchar(250) NOT NULL default '',
  C10 varchar(250) NOT NULL default '',
  C11 varchar(250) NOT NULL default '',
  C12 varchar(250) NOT NULL default '',
```

```
C13 varchar(250) NOT NULL default '',
C14 varchar(250) NOT NULL default '',
C15 varchar(250) NOT NULL default '',
C16 varchar(250) NOT NULL default '',
C17 varchar(250) NOT NULL default '',
D1 varchar(250) NOT NULL default '',
D2 varchar(250) NOT NULL default '',
D3 varchar(250) NOT NULL default '',
D4 varchar(250) NOT NULL default '',
D5 varchar(250) NOT NULL default '',
D6 varchar(250) NOT NULL default '',
D7 varchar(250) NOT NULL default '',
E1 varchar(250) NOT NULL default '',
E2 varchar(250) NOT NULL default '',
E3 varchar(250) NOT NULL default '',
E4 varchar(250) NOT NULL default '',
E5 varchar(250) NOT NULL default '',
E6 varchar(250) NOT NULL default '',
E7 varchar(250) NOT NULL default '',
F1 varchar(250) NOT NULL default '',
F2 varchar(250) NOT NULL default '',
F3 varchar(250) NOT NULL default '',
F4 varchar(250) NOT NULL default '',
F5 varchar(250) NOT NULL default '',
F6 varchar(250) NOT NULL default '',
F7 varchar(250) NOT NULL default '',
F8 varchar(250) NOT NULL default '',
F9 varchar(250) NOT NULL default '',
F10 varchar(250) NOT NULL default '',
F11 varchar(250) NOT NULL default '',
F12 varchar(250) NOT NULL default '',
F13 varchar(250) NOT NULL default '',
F14 varchar(250) NOT NULL default '',
F15 varchar(250) NOT NULL default '',
F16 varchar(250) NOT NULL default '',
F17 varchar(250) NOT NULL default '',
F18 varchar(250) NOT NULL default '',
F19 varchar(250) NOT NULL default '',
F20 varchar(250) NOT NULL default '',
F21 varchar(250) NOT NULL default '',
F22 varchar(250) NOT NULL default '',
F23 varchar(250) NOT NULL default '',
F24 varchar(250) NOT NULL default '',
F25 varchar(250) NOT NULL default '',
F26 varchar(250) NOT NULL default '',
F27 varchar(250) NOT NULL default '',
F28 varchar(250) NOT NULL default '',
G1 varchar(250) NOT NULL default '',
G2 varchar(250) NOT NULL default '',
G3 varchar(250) NOT NULL default '',
G4 varchar(250) NOT NULL default '',
G5 varchar(250) NOT NULL default '',
G6 varchar(250) NOT NULL default '',
G7 varchar(250) NOT NULL default '',
G8 varchar(250) NOT NULL default '',
G9 varchar(250) NOT NULL default '',
G10 varchar(250) NOT NULL default '',
G11 varchar(250) NOT NULL default '',
H1 varchar(250) NOT NULL default '',
H2 varchar(250) NOT NULL default '',
H3 varchar(250) NOT NULL default '',
H4 varchar(250) NOT NULL default '',
H5 varchar(250) NOT NULL default '',
H6 varchar(250) NOT NULL default '',
H7 varchar(250) NOT NULL default '',
H8 varchar(250) NOT NULL default '',
H9 varchar(250) NOT NULL default '',
```

```
  H10 varchar(250) NOT NULL default '',
  H11 varchar(250) NOT NULL default '',
  H12 varchar(250) NOT NULL default '',
  H13 varchar(250) NOT NULL default '',
  H14 varchar(250) NOT NULL default '',
  I1 varchar(250) NOT NULL default '',
  I2 varchar(250) NOT NULL default '',
  I3 varchar(250) NOT NULL default '',
  I4 varchar(250) NOT NULL default '',
  I5 varchar(250) NOT NULL default '',
  I6 varchar(250) NOT NULL default '',
  I7 varchar(250) NOT NULL default '',
  J1 varchar(250) NOT NULL default '',
  J2 varchar(250) NOT NULL default '',
  J3 varchar(250) NOT NULL default '',
  J4 varchar(250) NOT NULL default '',
  J5 varchar(250) NOT NULL default '',
  J6 varchar(250) NOT NULL default '',
  J7 varchar(250) NOT NULL default '',
  J8 varchar(250) NOT NULL default '',
  J9 varchar(250) NOT NULL default '',
  J10 varchar(250) NOT NULL default '',
  J11 varchar(250) NOT NULL default '',
  J12 varchar(250) NOT NULL default '',
  J13 varchar(250) NOT NULL default '',
  J14 varchar(250) NOT NULL default '',
  J15 varchar(250) NOT NULL default '',
  PRIMARY KEY  (id)
) TYPE=MyISAM COMMENT='Description of field names to be used for echo_facts';
```

As an example, following is the insert for a standard record:

```
INSERT INTO echo_facts_terms VALUES (1, '', 'Usual anthropological
designation', 'Related files in database', 'Self-designation', 'Self-
designation (man)', 'Self-designation (woman)', 'Designation of human being',
'Designations in written sources (alternative designations)', 'Official name
in countries of residence', 'Countries of residence', 'Disaspora', 'Official
ethnic regions', 'Status of ethnic group', 'Civil status of the individual',
'Representation in state institutions', 'State protection of culture',
'Cultural and political associations', 'International representation (UNO)',
'Type of international aid programs targeting this population', 'Language
name', 'Linguistic group', 'Official standard dialect', 'Used written
language, contemporary', 'Written language historical', 'UNESCO status of
language', 'Number of people speaking the language of their own ethnic group',
'Ratio of people speaking the language of their ethnic group', 'Number of
people who are native speakers of the language of their ethnic group', 'Ration
of people who are native speakers of the language of their ethnic group',
'Predominant language in use', 'Ratio of people literate in own language',
'Schooling in own language', 'Language courses in teaching institutions',
'Medias in own language', 'Own contemporary literature (in own language)',
'Own contemporary literature (in other language)', 'Population', 'Birth-rate',
'Mortality', 'Life expectancy', 'Fertility', 'Proportion of native pupulation
in residential areas', '', 'First European contact', 'First western
ethnographic description', 'First other ethnographic description', 'First
major ethnographic collection', 'Colonisation', '', '', 'Seasonal variations
in social morphology', 'Main social subdivisions', 'Residential mobility
type', 'Basic dwelling unit', 'Basic production unit', 'Basic consumption
unit', 'Basic appropriation unit', 'Kinship terminology type', 'Descent rule',
'Genealogical memory (in generations)', 'Marriage rule', 'Exogamous unit',
'Relative status of marriage partners', 'Extension of marriage prohibition',
'Polygamy', 'Residence', 'Relation between descent and residence', 'Pre-
marital contact between marriage partners', 'Marriage arrangements', 'Kinship
degree or category of contribution to marriage annangement', 'Divorce',
'Social stratification', 'Relevance of age to social status', 'Specialists',
```

```
'Political leadership', 'Political unit', 'Armed conflict - solidarity unit',
'Morality and law agencies', 'Principal religious system', 'Religious
organisations', 'Principal opportunities for collective rituals', 'Extent of
ritual cycle', 'Types of ritualists', 'Main mode of access to ritual
functions', 'Main type of ritual places', 'Basic components of the person',
'Representations of the after-life', 'Cosmological conceptions', 'Conceptions
of temporality', 'Type of ritual prescription for the mother, at birth',
'Ritual prescription for the father, at birth', 'Type of ritual treatment of
the new-born child', 'Type of ritual treatment of the remains at birth',
'Delivery place', 'Puberty initiation (male)', 'Puberty initiation (female)',
'Key ritual expression of wedding', 'Average age at marriage (male)', 'Average
age at marriage (female)', 'Rites of passage between age classes', 'Funeral
body disposal', 'Funeral body position', 'Disposal palce', '', 'Usual
resting/sitting position', 'Delivery position', 'Permanent bodily
alterations', 'Homosexuality', 'Pre-marital sexuality (male)', 'Pre-marital
sexuality (female)', 'Subsistence type', 'Biogeographical landscape', 'Rough
population density', 'Dwelling type', 'Stapple food', 'Gender specialisation
of tasks', 'Money', 'Mutually exclusive exhange spheres', 'Means of
transportation', 'Material of most efficient cutting tool', 'Vital imports',
'Specialisation in interethnic relations', 'Principal types of musical
instruments', 'Main type of dance performance', 'Main type of dance
performers');
```

Table 2: echo_facts

This table contains the answers for table 1, and is created as follows (displayed into two columns)

```
CREATE TABLE echo_facts (
  id int(11) NOT NULL
auto_increment,
  id_soc int(11) NOT NULL default
'0',
  author varchar(250) NOT NULL
default '',
  A1 text NOT NULL,
  A2 text NOT NULL,
  A3 text NOT NULL,
  A4 text NOT NULL,
  A5 text NOT NULL,
  A6 text NOT NULL,
  A7 text NOT NULL,
  A8 text NOT NULL,
  B1 text NOT NULL,
  B2 text NOT NULL,
  B3 text NOT NULL,
  B4 text NOT NULL,
  B5 text NOT NULL,
  B6 text NOT NULL,
  B7 text NOT NULL,
  B8 text NOT NULL,
  B9 text NOT NULL,
  B10 text NOT NULL,
  C1 text NOT NULL,
  C2 text NOT NULL,
  C3 text NOT NULL,
  C4 text NOT NULL,
  C5 text NOT NULL,
  C6 text NOT NULL,
  C7 text NOT NULL,
  C8 text NOT NULL,
  C9 text NOT NULL,
  C10 text NOT NULL,
  C11 text NOT NULL,
  C12 text NOT NULL,
  C13 text NOT NULL,
  C14 text NOT NULL,
  C15 text NOT NULL,
  C16 text NOT NULL,
  C17 text NOT NULL,
  D1 text NOT NULL,
  D2 text NOT NULL,
  D3 text NOT NULL,
  D4 text NOT NULL,
  D5 text NOT NULL,
  D6 text NOT NULL,
  D7 text NOT NULL,
  E1 text NOT NULL,
  E2 text NOT NULL,
  E3 text NOT NULL,
  E4 text NOT NULL,
  E5 text NOT NULL,
  E6 text NOT NULL,
  E7 text NOT NULL,
  F1 text NOT NULL,
  F2 text NOT NULL,
  F3 text NOT NULL,
  F4 text NOT NULL,
  F5 text NOT NULL,
  F6 text NOT NULL,
  F7 text NOT NULL,
  F8 text NOT NULL,
  F9 text NOT NULL,
  F10 text NOT NULL,
  F11 text NOT NULL,
  F12 text NOT NULL,
```

```
  F13 text NOT NULL,
  F14 text NOT NULL,
  F15 text NOT NULL,
  F16 text NOT NULL,
  F17 text NOT NULL,
  F18 text NOT NULL,
  F19 text NOT NULL,
  F20 text NOT NULL,
  F21 text NOT NULL,
  F22 text NOT NULL,
  F23 text NOT NULL,
  F24 text NOT NULL,
  F25 text NOT NULL,
  F26 text NOT NULL,
  F27 text NOT NULL,
  F28 text NOT NULL,
  G1 text NOT NULL,
  G2 text NOT NULL,
  G3 text NOT NULL,
  G4 text NOT NULL,
  G5 text NOT NULL,
  G6 text NOT NULL,
  G7 text NOT NULL,
  G8 text NOT NULL,
  G9 text NOT NULL,
  G10 text NOT NULL,
  G11 text NOT NULL,
  H1 text NOT NULL,
  H2 text NOT NULL,
  H3 text NOT NULL,
  H4 text NOT NULL,
  H5 text NOT NULL,
  H6 text NOT NULL,
  H7 text NOT NULL,
  H8 text NOT NULL,
  H9 text NOT NULL,
  H10 text NOT NULL,
  H11 text NOT NULL,
  H12 text NOT NULL,
  H13 text NOT NULL,
  H14 text NOT NULL,
  I1 text NOT NULL,
  I2 text NOT NULL,
  I3 text NOT NULL,
  I4 text NOT NULL,
  I5 text NOT NULL,
  I6 text NOT NULL,
  I7 text NOT NULL,
  J1 text NOT NULL,
  J2 text NOT NULL,
  J3 text NOT NULL,
  J4 text NOT NULL,
  J5 text NOT NULL,
  J6 text NOT NULL,
  J7 text NOT NULL,
  J8 text NOT NULL,
  J9 text NOT NULL,
  J10 text NOT NULL,
  J11 text NOT NULL,
  J12 text NOT NULL,
  J13 text NOT NULL,
  J14 text NOT NULL,
  J15 text NOT NULL,
  PRIMARY KEY  (id)
) TYPE=MyISAM COMMENT='Facts TABLE
for each society';
```

NOTE: the row identifications such A1, A2 etc. must be equal to the row identifications of table 1.

Identification Letters of tables echo_facts and echo_facts_terms

Currently, the letters of these identifications have the following meaning:

A Appellations
B Official status
C Language
D Demography
E Ethnological history
F Social organization
G Religion
H Life cycle
I Body
J Material culture

Table 3: echo_facts_links

This table contains eventual links to elements of table 2, and is created as follows:

```
CREATE TABLE echo_facts_links (
  id int(11) NOT NULL auto_increment,
```

```
  id_soc int(11) NOT NULL default '0',
  id_field varchar(5) NOT NULL default '',
  id_article_link int(11) default '0',
  external_link text,
  PRIMARY KEY  (id)
) TYPE=MyISAM COMMENT='Links to internal and external pages for echo_facts
fields';
```

PHP functions and structure

The specification sheet is called by the same module as other articles, however, when the article id is not known or is set to zero, then the specification sheet is displayed. The extract of this submodule is as follows:

```
<?php
/*
FILE:           display_content.php
AUTHOR:         Laurent Dousset
PROJECT:        ECHO
DATE CREATED:   04 February 2003
DATE UPDATED:   27 May 2003
DESCRIPTION:    This file contains the various functions used to display the
article content
*/

[... other functions described elswhere in this document ...]

//***** DISPLAYS THE ARTICLE

function display_article($id_soc, $id_article, $font_face, $font_size,
$action)
{
   if ($action == "links")
   {
      [....]
   }

[ and other handling of $action events ]

      if ($id_article == 0)
   {
      include "scripts/display_facts.php";
      print_facts_table($id_soc);
   }
?>
```

The `display_facts.php` and the specified `print_facts_table()` function are as follows:

```
<?php
/*
FILE:           display_facts.php
AUTHOR:         Laurent dousset
PROJECT:        ECHO
DATE CREATED:   05 April 2003
DATE UPDATED:   20 Mai 2003
DESCRIPTION:    To display general facts sheet
*/


//*****************************************
function print_facts_table($soc_id)
```

```
{

   $fields = get_field_codes($soc_id);          // get the codes A1, A2 etc.
   $field_number = count($fields);              // number of fields
   $field_names = get_field_names($soc_id);     // get the names for A1, A2
etc.
   $soc_facts = get_soc_facts($soc_id);         // get the actual content for
soc_id

   echo "<p><font face=\"Courier New,Courier,Monaco\" size=\"1\"><b>GENERAL
FACTS MENU: </b></font><font face=\"Arial\" size=\"1\">
   <a href=\"#appellations\">appellations</a> | <a
href=\"#official_status\">official status</a> |
   <a href=\"#language\">language</a> | <a href=\"#demography\">demography</a>
| <a href=\"#history\">ethnological history</a> | <a
href=\"#socialorg\">social organization</a>
    | <a href=\"#religion\">religion</a> | <a href=\"#life\">life cycle</a> |
<a href=\"#body\">body</a> | <a href=\"#material\">material culture</a>
   </font></p><br>";

// search for A(n) fieldnames
   echo "<table width=\"100%\" border=\"0\" cellpadding=\"5\" cellspacing=\"0\"
border=\"0\"><tr><td bgcolor=\"#778899\"><a name=\"appellations\"</a>
   <b>APPELLATIONS</b></td><td width=\"110\" bgcolor=\"#778899\">Further
details</td></tr></table>";

   list_tables("A",$soc_id,$field_number,$fields,$field_names,$soc_facts);


// search for B(n) fieldnames
   echo "<br><table width=\"100%\" border=\"0\" cellpadding=\"5\"
cellspacing=\"0\" border=\"0\"><tr><td bgcolor=\"#778899\"><a
name=\"official_status\"</a>
   <b>OFFICIAL STATUS</b></td><td width=\"110\" bgcolor=\"#778899\">Further
details</td></tr></table>";

   list_tables("B",$soc_id,$field_number,$fields,$field_names,$soc_facts);


// search for C(n) fieldnames
   echo "<br><table width=\"100%\" border=\"0\" cellpadding=\"5\"
cellspacing=\"0\" border=\"0\"><tr><td bgcolor=\"#778899\"><a
name=\"language\"</a>
   <b>LANGUAGE</b></td><td width=\"110\" bgcolor=\"#778899\">Further
details</td></tr></table>";

   list_tables("C",$soc_id,$field_number,$fields,$field_names,$soc_facts);

// search for D(n) fieldnames
   echo "<br><table width=\"100%\" border=\"0\" cellpadding=\"5\"
cellspacing=\"0\" border=\"0\"><tr><td bgcolor=\"#778899\"><a
name=\"demography\"</a>
   <b>DEMOGRAPHY</b></td><td width=\"110\" bgcolor=\"#778899\">Further
details</td></tr></table>";

   list_tables("D",$soc_id,$field_number,$fields,$field_names,$soc_facts);

// search for E(n) fieldnames
   echo "<br><table width=\"100%\" border=\"0\" cellpadding=\"5\"
cellspacing=\"0\" border=\"0\"><tr><td bgcolor=\"#778899\"><a
name=\"history\"</a>
   <b>ETHNOLOGICAL HISTORY</b></td><td width=\"110\"
bgcolor=\"#778899\">Further details</td></tr></table>";

   list_tables("E",$soc_id,$field_number,$fields,$field_names,$soc_facts);

// search for F(n) fieldnames
```

```
   echo "<br><table width=\"100%\" border=\"0\" cellpadding=\"5\"
cellspacing=\"0\" border=\"0\"><tr><td bgcolor=\"#778899\"><a
name=\"socialorg\"</a>
   <b>SOCIAL ORGANISATION</b></td><td width=\"110\" bgcolor=\"#778899\">Further
details</td></tr></table>";

   list_tables("F",$soc_id,$field_number,$fields,$field_names,$soc_facts);

// search for G(n) fieldnames
   echo "<br><table width=\"100%\" border=\"0\" cellpadding=\"5\"
cellspacing=\"0\" border=\"0\"><tr><td bgcolor=\"#778899\"><a
name=\"religion\"</a>
   <b>RELIGION</b></td><td width=\"110\" bgcolor=\"#778899\">Further
details</td></tr></table>";

   list_tables("G",$soc_id,$field_number,$fields,$field_names,$soc_facts);


// search for H(n) fieldnames
   echo "<br><table width=\"100%\" border=\"0\" cellpadding=\"5\"
cellspacing=\"0\" border=\"0\"><tr><td bgcolor=\"#778899\"><a
name=\"life\"</a>
   <b>LIFE CYCLE</b></td><td width=\"110\" bgcolor=\"#778899\">Further
details</td></tr></table>";

   list_tables("H",$soc_id,$field_number,$fields,$field_names,$soc_facts);

// search for I(n) fieldnames
   echo "<br><table width=\"100%\" border=\"0\" cellpadding=\"5\"
cellspacing=\"0\" border=\"0\"><tr><td bgcolor=\"#778899\"><a
name=\"body\"</a>
   <b>BODY</b></td><td width=\"110\" bgcolor=\"#778899\">Further
details</td></tr></table>";

   list_tables("I",$soc_id,$field_number,$fields,$field_names,$soc_facts);

// search for J(n) fieldnames
   echo "<br><table width=\"100%\" border=\"0\" cellpadding=\"5\"
cellspacing=\"0\" border=\"0\"><tr><td bgcolor=\"#778899\"><a
name=\"material\"</a>
   <b>MATERIAL CULTURE</b></td><td width=\"110\" bgcolor=\"#778899\">Further
details</td></tr></table>";

   list_tables("J",$soc_id,$field_number,$fields,$field_names,$soc_facts);

}

/**************************
//**************************
//
//   HELP FUNCTIONS BELOW
//
//**************************
/**************************/

function
list_tables($letter,$soc_id,$field_number,$fields,$field_names,$soc_facts)
{

   $ff = "Arial";
   $fs = 1;
   echo "<table width=\"100%\" cellpadding=\"5\" cellspacing=\"0\">";
   $i = 0;
   for ($x = 0; $x < $field_number; $x++)
   {
      $pos = strpos($fields[$x],$letter);
      if (is_integer($pos))
```

```
     {
//      if ($soc_facts["$fields[$x]"] != "")
//      {
           $i++;
           if ($i % 2 == 0)
           { $color = "#E0FFFF"; }
              else { $color = "#B0C4DE"; }

           echo "<tr bgcolor=\"$color\" valign=\"top\"><td width=\"20\"><font
face=\"$ff\" size=\"$fs\" color=\"#696969\">$fields[$x]</font></td>
           <td width=\"40%\"><font face=\"$ff\"
size=\"$fs\">$field_names[$x]</font></td>
           <td><font face=\"$ff\" size=\"$fs\"><b>";
           $print_string = nl2br($soc_facts["$fields[$x]"]);
           echo $print_string;
           echo "</b></font></td><td width=\"110\">";

           search_links($fields[$x],$soc_id);

           echo "</td></tr>";
//      }
     }
   }
   echo "</table>";

}

function search_links($field,$soc_id)
{

   $search_string = "SELECT * FROM `echo_facts_links` WHERE id_soc = '$soc_id'
AND id_field like '".$field."'";
   $facts_result = mysql_query ("$search_string")
      or die ("Sorry, there was a problem connecting to the database. <br>
Please hit the back button of your browser and try again later.");
   $rows_number = mysql_numrows($facts_result);
    if ($rows_number > 0)
    {
     while ($row = mysql_fetch_row($facts_result))
     {

        $article_result = mysql_query ("Select title from echo_articles where
id = '$row[3]'")
           or die ("Sorry, there was a problem connecting to the database. <br>
Please hit the back button of your browser and try again later.");
        $rows_number = mysql_numrows($article_result);
        if ($rows_number > 0)
        {
           $row_article = mysql_fetch_row($article_result);
           echo "<font face=\"Arial\" size=\"1\"><a
href=\"articles.php?id_soc=$soc_id&id_article=$row[3]\">$row_article[0]</a></f
ont><br>";
        }
     }
   }
   mysql_free_result($facts_result);
}


//*****************************************
function get_field_codes($soc_id)
{
   $soc_fact_terms = mysql_query ("SELECT echo_facts_terms_id FROM
`echo_societies` where id = '$soc_id'")
     or die ("Sorry, there was a problem connecting to the database. <br>
Please hit the back button of your browser and try again later.");
   $rows_number = mysql_numrows($soc_fact_terms);
```

```
   if ($rows_number > 0)
   {
      $row = mysql_fetch_row($soc_fact_terms);
      $terms_id = $row[0];
   } else { $terms_id = 1; }
   if ($terms_id == 0) {$terms_id = 1;}

   $facts_result = mysql_query ("SELECT * FROM `echo_facts_terms` where id =
'$terms_id'")
      or die ("Sorry, there was a problem connecting to the database. <br>
Please hit the back button of your browser and try again later.");

   $i = 0;
   while ($i < mysql_num_fields($facts_result))
   {
      $meta = mysql_fetch_field($facts_result);
      if (!$meta) {
            echo "No information available<br />\n";   }

      $fields[]=$meta->name;
      $i++;
   }
   mysql_free_result($facts_result);
   return $fields;
}

//*****************************************
function get_field_names($soc_id)
{
   $soc_fact_terms = mysql_query ("SELECT echo_facts_terms_id FROM
`echo_societies` where id = '$soc_id'")
      or die ("Sorry, there was a problem connecting to the database. <br>
Please hit the back button of your browser and try again later.");
   $rows_number = mysql_numrows($soc_fact_terms);
   if ($rows_number > 0)
   {
      $row = mysql_fetch_row($soc_fact_terms);
      $terms_id = $row[0];
   } else { $terms_id = 1; }
   if ($terms_id == 0) {$terms_id = 1;}

   $facts_result = mysql_query ("SELECT * FROM `echo_facts_terms` where id =
'$terms_id'")
      or die ("Sorry, there was a problem connecting to the database. <br>
Please hit the back button of your browser and try again later.");

   $field_names = mysql_fetch_row($facts_result);
   mysql_free_result($facts_result);

   return $field_names;
}
//*****************************************
function get_soc_facts($soc_id)
{
   $facts_result = mysql_query ("SELECT * FROM `echo_facts` where id_soc =
'$soc_id'")
      or die ("Sorry, there was a problem connecting to the database. <br>
Please hit the back button of your browser and try again later.");

   $rows_number = mysql_numrows($facts_result);
    if ($rows_number > 0)
    {
     $soc_facts = mysql_fetch_array($facts_result,MYSQL_ASSOC);
   }
   mysql_free_result($facts_result);
   return $soc_facts;
}
```

?>

Backoffice management

The backoffice management of this module is fully implemented.
it is possible to

1. Fill in the answers to the specification sheet for each society
2. Link such an answer to an explanatory article included elsewhere in the database
3. Modify the specification sheet labels

Article display – main module (image inserts, text inserts, article list inserts)

Description

This is the main module of the PHP interface. It contains two pages, one is the caller (articles.html), the other the handler (display_content.php). The handler never appears as such to the user, but inserts results of database queries into the page articles.html.
The distinction between caller and handler was a necessary one in the work process. It does, indeed, enable designers and programmers to work simultaneously on the same page but without interfering into each other's principles. All that is need is an include call in the caller page. Subsequently, whenever the designer wants to process or display certain data, the function name is called. Variables are handled transparently so that designers do not have to worry about the user queries themselves.

The module depends on various database tables and handles various aspects of it. It creates one of the principal screens in the website. In particular, it handles:

- insert of society name following a user query
- insert of society details " " (general facts)
- insert the list of articles for this society from the articles database table
- display the article the user has decided to read from the articles table
- insert illustrations that accompany the viewed article from the images table as thumbnails
- create full-view pages for each thumbnail (includes some simple javascript)
- insert user_comments module
- insert glossary module for this article
- insert navigational principles such as next article to read and previous article to read

Some tables and sub-scripts are described elsewhere in this document. Here only the kernel of the engine is presented.

Benefits

See above.

Limitations

No particular limitations

Database table structure

Various dependencies in tables are used in this module. These tables are:

echo_societies
echo_articles
echo_images
echo_glossary_root (described elsewhere)
echo_glossary_terms (described elsewhere)
echo_articles_comments (described elsewhere)

Following are some extracts of the principles of these tables.

echo_societies

id	name	etc.
numerical this is the reference to the society. In other tables, this reference is called id_soc	*varchar* this is the name of the social group that has been decided to be used as a screen name	*various* following are various columns that inform the sub-module general facts in the article display module

echo_articles

id	id_soc	order	title
numerical this is the reference to the article.	*numerical* this is the reference to the society (see table above)	*numerical* this number is used to set the sequence of the articles for a society. If there are 2 articles for a society where the first has order 2 and the second order 1, then the second article will be displayed before the first article	*varchar* this is the name of the article that will be used to construct the navigational menu within a society.

echo_images

This table does not contain the image, photo or any other illustration itself, but only stores the exact location of its file within the filesystem. It is therefore a light table. If one image relates to two or more societies, or to two or more articles, then it is possible to duplicate rows without slowing down the system.

id	filename	x_size and y_size	caption	id_soc	id_article	etc
numerical id of image	*varchar* contains the name of the image file. the path to this file can be set in the file global.inc.php Currently, the files are in the folder /media/images/	*numerical* these values are used to create the thumbnails but, more importantly, to determine the size of the pop-up window.	*text* is displayed in the pop-up window.	*numeric* reference to echo_societies above	*numeric* reference to echo_articles above	others

echo_sounds

This table does not contain the sounds or music-files themselves, but only stores the exact location of its file within the filesystem. It is therefore a light table. If one file relates to two or more societies, or to two or more articles, then it is possible to duplicate rows without slowing down the system.

id	filename	caption	id_soc	id_article	sound_title
numerical id of image	*varchar* contains the name of the sound file. The path to this file can be set in the file global.inc.php Currently, the files are in the folder /media/sounds/	*text* is displayed in the pop-up window.	*numeric* reference to echo_societies above	*numeric* reference to echo_articles above	*varchar* some descriptive text for the resource

echo_ext_doc_to_articles

This table contains external links/resources providers want to specify for a specific article. The resources to which the links point can be of any type: image, html, txt etc.

id	id_soc	id_article	url_text	url
numerical id of image	*numeric* reference to echo_societies above	*numeric* reference to echo_articles above	*varchar* Some short text illustrating/identifying the external resource	*varchar* the actual URI/URL of the external resource

echo_tables

This table contains table definitions providers want to add for a specific article. This tool is provided for illustrative purposes only

id	id_soc	id_article	title	commets	others
numerical id of image	*numeric* reference to echo_societies above	*numeric* reference to echo_articles above	*text* the title of the table	*text* some comment the provider wants to add	*the various fields of the tables are defined after this (see below for mysql table definition)*

SQL Commands to create the database tables in MySQL

Following are the SQL commands for creating the article table and related tables.

```
#
# Table structure for table `echo_articles`
#

CREATE TABLE echo_articles (
  id int(11) NOT NULL auto_increment,
  id_soc int(11) NOT NULL default '0',
  order smallint(6) NOT NULL default '0',
  title varchar(250) NOT NULL default '',
  article_text text NOT NULL,
  author varchar(250) NOT NULL default '',
  date_created date NOT NULL default '0000-00-00',
  date_modified date NOT NULL default '0000-00-00',
  category varchar(250) NOT NULL default '',
  PRIMARY KEY  (id),
  FULLTEXT KEY article_text (article_text),
  FULLTEXT KEY title (title)
) TYPE=MyISAM COMMENT='articles of the NECEP database';
# --------------------------------------------------------

#
# Table structure for table `echo_ext_doc_to_articles`
#

CREATE TABLE echo_ext_doc_to_articles (
  id int(11) NOT NULL auto_increment,
  id_soc int(11) NOT NULL default '0',
  id_article int(11) NOT NULL default '0',
  url_text varchar(250) NOT NULL default '',
  url varchar(250) NOT NULL default '',
  PRIMARY KEY  (id)
) TYPE=MyISAM COMMENT='Stores for each article a link to an external image';
# --------------------------------------------------------

#
# Table structure for table `echo_images`
#

CREATE TABLE echo_images (
  id int(11) NOT NULL auto_increment,
  filename varchar(100) NOT NULL default '',
  x_size int(11) NOT NULL default '0',
  y_size int(11) NOT NULL default '0',
  caption text NOT NULL,
  id_soc int(11) NOT NULL default '0',
  id_article int(11) NOT NULL default '0',
  image_title varchar(250) NOT NULL default '',
  class int(11) NOT NULL default '0',
  PRIMARY KEY  (id),
  FULLTEXT KEY caption (caption)
) TYPE=MyISAM COMMENT='image informtion storage';
# --------------------------------------------------------

#
# Table structure for table `echo_sounds`
#

CREATE TABLE echo_sounds (
  id int(11) NOT NULL auto_increment,
```

```
  filename varchar(100) NOT NULL default '',
  caption text NOT NULL,
  id_soc int(11) NOT NULL default '0',
  id_article int(11) NOT NULL default '0',
  sound_title varchar(250) NOT NULL default '',
  class int(11) NOT NULL default '0',
  PRIMARY KEY  (id),
  FULLTEXT KEY caption (caption)
) TYPE=MyISAM COMMENT='sound informtion storage';
# --------------------------------------------------------

#
# Table structure for table `echo_tables`
#

CREATE TABLE echo_tables (
  id int(11) NOT NULL auto_increment,
  id_soc int(11) NOT NULL default '0',
  id_article int(11) NOT NULL default '0',
  title text NOT NULL,
  comments text NOT NULL,
  rc_11 varchar(250) NOT NULL default '',
  rc_12 varchar(250) NOT NULL default '',
  rc_13 varchar(250) NOT NULL default '',
  rc_14 varchar(250) NOT NULL default '',
  rc_15 varchar(250) NOT NULL default '',
  rc_16 varchar(250) NOT NULL default '',
  rc_21 varchar(250) NOT NULL default '',
  rc_22 varchar(250) NOT NULL default '',
  rc_23 varchar(250) NOT NULL default '',
  rc_24 varchar(250) NOT NULL default '',
  rc_25 varchar(250) NOT NULL default '',
  rc_26 varchar(250) NOT NULL default '',
  rc_31 varchar(250) NOT NULL default '',
  rc_32 varchar(250) NOT NULL default '',
  rc_33 varchar(250) NOT NULL default '',
  rc_34 varchar(250) NOT NULL default '',
  rc_35 varchar(250) NOT NULL default '',
  rc_36 varchar(250) NOT NULL default '',
  rc_41 varchar(250) NOT NULL default '',
  rc_42 varchar(250) NOT NULL default '',
  rc_43 varchar(250) NOT NULL default '',
  rc_44 varchar(250) NOT NULL default '',
  rc_45 varchar(250) NOT NULL default '',
  rc_46 varchar(250) NOT NULL default '',
  rc_51 varchar(250) NOT NULL default '',
  rc_52 varchar(250) NOT NULL default '',
  rc_53 varchar(250) NOT NULL default '',
  rc_54 varchar(250) NOT NULL default '',
  rc_55 varchar(250) NOT NULL default '',
  rc_56 varchar(250) NOT NULL default '',
  rc_61 varchar(250) NOT NULL default '',
  rc_62 varchar(250) NOT NULL default '',
  rc_63 varchar(250) NOT NULL default '',
  rc_64 varchar(250) NOT NULL default '',
  rc_65 varchar(250) NOT NULL default '',
  rc_66 varchar(250) NOT NULL default '',
  rc_71 varchar(250) NOT NULL default '',
  rc_72 varchar(250) NOT NULL default '',
  rc_73 varchar(250) NOT NULL default '',
  rc_74 varchar(250) NOT NULL default '',
  rc_75 varchar(250) NOT NULL default '',
  rc_76 varchar(250) NOT NULL default '',
  PRIMARY KEY  (id)
) TYPE=MyISAM COMMENT='Free tables for echo-necep';
```

PHP functions and structure

The caller (articles.html, or any other page)

The caller (articles.html) can be arranged in simple html to the wishes of the designer. To display the various elements of the database into the page, the designer may call various PHP functions. To call these functions, he must however include a number of include files. To do this, he must place at the top of his page, even above the <html> tag, the following PHP commands:

```
<?php
   include ("db_connect.inc.php");
   include ("global.inc.php");
   include ("scripts/display_content.php");
   include ("user_interaction/article_comments.inc.php");
   include ("scripts/glossary_parse.inc.php");
?>
```

These includes do not display anything yet, but declare the various variables and functions the designer may want to use later on. Following is the list functions he may use:

```
<?php
   // THIS FUNCTION INSERTS THE NAME OF THE SOCIETY
   // a more comprehensive function is below
   // PLEASE LEAVE AS IS
   display_name_simple($id_soc);
?>
```

```
<?php
// THIS FUNCTION INSERTS THE NAME OF THE SOCIETY
// PLEASE MODIFY FONT FACE, SIZE AND COLOR.
   $font_face = "Arial";
   $font_size = 5;
   $font_color = "red";
   display_name($id_soc, $font_face, $font_size, $font_color);
?>
```

```
<?php
// THIS FUNCTION INSERTS THE ARTICLE LIST FOR A SOCIETY
// PLEASE MODIFY FONT FACE AND SIZE.
   $font_face = "Arial";
   $font_size = 1;
   display_article_list($id_soc, $font_face, $font_size, $id_article);
?>
```

```
<?php
// THIS FUNCTION ADDS THE GLOSSARY FOR THE DISPLAYED ARTICLE
// PLEASE MODIFY FONT FACE AND SIZE.
   $font_face = "Arial";
   $font_size = 1;
   add_glossary($id_article, $font_face, $font_size);
?>
```

```
<?php
// THIS FUNCTION INSERTS THE ARTICLE ITSELF
// PLEASE MODIFY FONT FACE AND SIZE.
   $font_face = "Arial";
   $font_size = 2;
   display_article($id_soc,$id_article, $font_face, $font_size);
```

```
?>
```

```
<?php
// THIS SCRIPT INSERTS USER COMMENTS LEFT FOR THE ARTICLE
// INCLUDING A FORM FOR LEAVING COMMENTS
   insert_comment($id_article);
?>
```

```
<?php
// THIS FUNCTION INSERTS THE IMAGES FOR AN ARTICLE
// PLEASE MODIFY THE WIDTH OF THE THUMBNAILS/PREVIEWS
   $max_width = 120;
   insert_image($id_article, $max_width);
?>
```

Each single function (single block above), can be called from wherever the designer would like the content to be displayed. For example, if he wishes to include the name of the society in the title of the page, he could write (copy-paste) this as follows (there is no need to worry about the variable $id_soc, the function will handle this; just leave as is).

```
<title>NECEP proto-type: <?php display_name_simple($id_soc); ?></title>
```

In order for pop ups to work, the page must contain the following (or similar) javascript in the head part of the page:

```
<script language="JavaScript">
   function open_popup(page)
   {
      window.open(page,'');
      return false;
   }
</script>
```

The handler (display_content.php)

The file display_content.php handles most functions called by the caller. These functions are reproduced below, but are, in summary:

`function display_name_simple($id_soc)`
Simply display the name of the social group without any formatting

`function display_name($id_soc, $font_face, $font_size, $font_color)`
Displays the name of the social group with formatting as defined in the variables $font_face etc.

`function display_article_list($id_soc, $font_face, $font_size, $id_current_article)`
Displays a list of articles relevant for this society with a link to these articles (via articles.html)

`function display_article($id_soc, $id_article, $font_face, $font_size)`
Displays the article itself, with links to the previous and next article in the sequence via articles.html. If the next pointer reaches the end of the article list, then it is set back to zero, which displays the general facts page for a society. $id_article = 0 is reserved for the general facts page of a society.

`function insert_image($id_article, $max_width)`
Inserts the images as thumbnails that accompany the article. $max_width sets how wide the images have to be. The height is proportionally adapted.

The full script is as follows:

```
<?php

/*
FILE:           display_content.php
AUTHOR:         Laurent Dousset
PROJECT:        ECHO
DATE CREATED:   04 February 2003
DATE UPDATED:   25 October 2003
DESCRIPTION:    This file contains the various functions used to display the
                article content
*/


//***** DISPLAYS THE NAME OF THE SOCIETY (NO FORMATTING)
function display_name_simple($id_soc)
{
   $society = mysql_query ("SELECT name FROM `echo_societies`
      where id = '$id_soc'");
    $rows_soc = mysql_numrows($society);
    if ($rows_soc > 0) {
     while ($row = mysql_fetch_row($society))  {
          echo "$row[0]";
   } }
}

//***** DISPLAYS THE NAME OF THE SOCIETY (WITH FORMATTING)
function display_name($id_soc, $font_face, $font_size, $font_color)
{
   $society = mysql_query ("SELECT name FROM `echo_societies`
      where id = '$id_soc'");
    $rows_soc = mysql_numrows($society);
    if ($rows_soc > 0) {
     while ($row = mysql_fetch_row($society))  {
          echo "<p><font face=\"$font_face\" size=\"$font_size\"
            color=\"$font_color\"><b>$row[0]</b><font></p>";
   } }
}


//***** DISPLAY THE LIST OF ARTICLES FOR THIS SOCIETY
function display_article_list($id_soc, $font_face, $font_size,
$id_current_article)
 {
   if (isset($id_soc)) {
      if ($id_current_article == 0) {
         echo "<font face=\"$font_face\" size=\"$font_size\">>
            General facts<br><br>";
      } else {
         echo "<font face=\"$font_face\" size=\"$font_size\">
            <a href=\"articles.html?id_soc=$id_soc&id_article=0\">
            General facts</a><br><br>";
      }

      $article_list = mysql_query ("SELECT id, title FROM `echo_articles`
         where id_soc = '$id_soc' ORDER by `order`");
      $rows_articles = mysql_numrows($article_list);
      if ($rows_articles > 0) {
         while ($row = mysql_fetch_row($article_list)) {
            if ($id_current_article == $row[0])  {
               echo "> $row[1]<br>";
            } else {
               echo "<a
href=\"articles.html?id_soc=$id_soc&id_article=$row[0]\">
                  $row[1]</a><br>";
      } } }
      echo "</font>";
```

```
   }
}

//***** DISPLAYS THE ARTICLE
function display_article($id_soc, $id_article, $font_face, $font_size)
{
   if ($id_article != 0) {
      $article = mysql_query ("SELECT * FROM `echo_articles`
         where id = '$id_article'");
      $rows_article = mysql_numrows($article);

      if ($rows_article > 0) {
         $font_size++;
         echo "<font face=\"$font_face\"
size=\"$font_size\"><b>$row[3]</b><font>";
         $font_size--;
         echo "<font face=\"$font_face\" size=\"$font_size\"><br><br>
            $row[4]</font></font>";
   } }

   if ($id_article == 0) {
      echo "INSERT general facts page for $id_soc";
      // THIS FUNCTION IS DESCRIBED ELSWHERE
   }

   // check set of articles to determine following and previous article *******
   $sequence[] = 0;
   $articles = mysql_query ("SELECT id FROM `echo_articles`
      where id_soc = '$id_soc' order by `order`");
    $rows_articles = mysql_numrows($articles);
    while ($row = mysql_fetch_row($articles))  {
      $sequence[] = $row[0];
   }
   reset($sequence);
   while (list($key,$val) = each ($sequence)) {
      if ($val == $id_article) { $current_sequence = $key; }
   }

   $prev = $current_sequence-1;
   if ($prev == 0) { $prev = 0; }

   $next = $current_sequence+1;
   end ($sequence);
   $last_sequence = key($sequence);
   if ($next > $last_sequence) { $next = 0; }

   // display previous and next links *******
   echo "<br><br><table width=\"100%\" border=\"0\"><tr>";
   if ($id_article != 0)
   {
      echo "<td><font face=\"Arial\" size=\"1\">
         <a href=\"articles.html?id_soc=$id_soc&id_article=$sequence[$prev]\">
         previous article</a></font></td>";
      echo "<td align=\"right\"><font face=\"Arial\" size=\"1\">
         <a href=\"articles.html?id_soc=$id_soc&id_article=$sequence[$next]\">
         next article</a></font></td>";
   } else {
      echo "<td align=\"right\"><font face=\"Arial\" size=\"1\">
         <a href=\"articles.html?id_soc=$id_soc&id_article=$sequence[$next]\">
         next article</a></font></td>";
   }
   echo "</tr></table>";
}


//***** DISPLAYS THE IMAGES
```

```
function insert_image($id_article, $max_width)
{
   global $image_path, $image_url;
   $image_result = mysql_query ("SELECT id, filename, image_title FROM
`echo_images`
      where id_article = '$id_article'");
   $rows_image = mysql_numrows($image_result);
   if ($rows_image > 0) {
      while ($row = mysql_fetch_row($image_result)) {
         $path = $image_path.$row[1];
         $link = $image_url.$row[1];

         echo "<br><a href=\"scripts/detail.php?id=$row[0]\" alt=\"$row[2]\"
            title=\"$row[2]\" onClick=\"return open_popup
            ('scripts/detail.php?id=$row[0]')\">
            <img src=\"scripts/thumbnail.php?path=$path&max_width=$max_width\"
            alt=\"$row[2]\"></a>";
   } }
}
?>
```

Backffocie management

Bakcoffice provided: YES, fully implemented

Glossary (auxiliary module)

Description

2 database tables and 2 (in fact 3) short PHP functions to choose from constitute this module. The task of this module is to parse (read-compare-change) a text before it is sent to the end user (browser), and check for the occurrence of words for which there is a glossary-entry, and to (1) replace these words with a link to a glossary window, or (2) create a list of words for which there are entries in the glossary. The PHP functions are handy as they are short and can be invoked from anywhere and for any article. Preferably use function number 2 (see below), as this function simply inserts a glossary list wherever it is called.
Author of module: Laurent Dousset
Copyright: waved

Benefits

Using this module allows administrators and content-providers to simply add words to be glossed, or change descriptions, in the database and the Web content will automatically be indexed with links to the glossary.

Limitations

No particular limitations. Note however that function 1 is case-sensitive. Use preferably function 2, as it is faster, visually more pleasant and not case-sensitive.

Database table structure

The glossary structure consists of two tables. The principle is as follows:

For example, *table 1* below has the entry “patrilineal” with id 15.

Table 2 contains derivates from this word, such as “patrilineal” itself, but also “patrilineality”, for example. Both words in table 2 have as reference (id_root) the number 15 (from the word “patrilineal” in table 1) and refer to the same glossary entry.

1)**echo_glossary_root** (this tables stores the definition of the words, it is the glossary properly speaking).

id	root_word	definition	see_also_1 see_also_2 see_also_3
numerical this is there reference identification through which the definition is called	*varchar* this is the title of the definition (similar to the root dictionary entry)	*text* this is the definition itself. There is no limitation in length and formatting	*numerical* is the id for other words that are relevant for the definition

2)**echo_glossary_terms** (this table stores possible derivate forms of the words, later called phenotypes)

id	id_root	glossary_terms
numerical (unused, yet)	*numerical* this is the reference to table 1 (glossary_root) above	*varchar* these are the phenotypes of root_word(s) that can appear in texts.

SQL Commands to create the database tables in MySQL

```
#
# Table structure for table `echo_glossary_root`
#

CREATE TABLE echo_glossary_root (
  id int(11) NOT NULL auto_increment,
  root_word varchar(150) NOT NULL default '',
  definition text NOT NULL,
  see_also_1 smallint(4) default NULL,
  see_also_2 smallint(4) default NULL,
  see_also_3 smallint(4) default NULL,
  PRIMARY KEY  (id)
) TYPE=MyISAM COMMENT='Glossary: root terms with definitions';


#
# Table structure for table `echo_glossary_terms`
#
CREATE TABLE echo_glossary_terms (
  id int(11) NOT NULL auto_increment,
  id_root int(11) NOT NULL default '0',
  glossary_terms varchar(200) NOT NULL default '',
  PRIMARY KEY  (id)
) TYPE=MyISAM COMMENT='Glossary: phenotypes to be searched for in texts';
```

PHP functions to invoke the use of the glossary

There are two functions to choose from, one of which calls an independent third sub-function. They may be called from anywhere in the site provided you send the appropriate include command (see below). It simply needs as argument the $id of an article in the echo_articles database table and will, for function 1: return the text with indexed links to glossary entries from the database; and for function 2: not alter the text but return an array ($gloss_arr) of ids of words that need to be glossed, and write this list to the browser, including links, wherever this function is called. The PHP functions/file look as follows (preferably use function 2):

The major difference between function 1 and function 2 is that function 1 takes as argument a text variable, while function 2 takes as argument the identification of an article in the database.

```
?php

/*

FILE:           glossary_parse.inc.php
AUTHOR:         Laurent dousset
PROJECT:        ECHO
DATE CREATED:   20 November 2002

DESCRIPTION:    Parses any text $content and creates links to glossary
*/

// *** FUNCTION 1 for glossary ***
// Parses $content agains the content of the glossary_terms table
// Returns $content with incorporated links to glossary.php


function glossary_parse($content)
{
    // Read glossary terms from database and replaces occurences in text
string
    $glossary_result = mysql_query ("SELECT * FROM `echo_glossary_terms`
     ORDER by id");
    $rows_glossary = mysql_numrows($glossary_result);
    if ($rows_glossary > 0)
    {
        while ($row = mysql_fetch_row($glossary_result))
        {
            $temp_replace = "<a href=\"glossary.php?id=$row[1]\"
                target=\"_blank\">$row[2]</a>$speacial_char[$x]";
            $content = str_replace($row[2],$temp_replace,$content);
    } }
    return($content);
}
// *** END function glossary_parse($content) ***



// *** FUNCTION 2 for glossary ***
// This function checks the text and returns the list of words for
// which there is a glossary entry in an array. It does not change $content

function add_glossary($article_id, $font_face, $font_size)
{
   if ($article_id > 0) {
      $article = mysql_query ("SELECT article_text FROM `echo_articles`
         where id='$article_id'");
    $rows_article = mysql_numrows($article);
    if ($rows_article > 0)  {
        while ($row2 = mysql_fetch_row($article)) {
           $content = $row2[0];
        }
```

```
        $gloss_arr[] = "0"; // Instantiate Array with an impossible Glossary
entry
        $glossary_result = mysql_query ("SELECT * FROM `echo_glossary_terms`
           ORDER by id");
        $rows_glossary = mysql_numrows($glossary_result);
        if ($rows_glossary > 0) {
             while ($row = mysql_fetch_row($glossary_result)) {
                if (stristr($content,$row[2]))    // check is glossary term in
text
                {
                     if (!in_array($row[1],$gloss_arr))  // check if in array
                        { $gloss_arr[] = $row[1]; }     // add id to array
     } } }
     if (count($gloss_arr) > 1) {
        display_gloss($gloss_arr, $font_face, $font_size); }
    }}
}
// *** END function add_glossary($content) ***



// *** FUNCTION display_gloss ***
// This function us called by add_glossary($content) and displays
// the glossary wherever the function is called

function display_gloss($gloss_arr, $font_face, $font_size)
{
    echo "<font face=\"$font_face\" size=\"$font_size\"><b>Glossary:</b><br>";
    foreach ($gloss_arr as $word_id) {
        $gloss_index = mysql_query ("SELECT * FROM `echo_glossary_root`
             where id = $word_id");
        $rows_nr = mysql_numrows($gloss_index);
        if ($rows_nr > 0) {
            while ($row_gloss = mysql_fetch_row($gloss_index))  {
                echo "<a href=\"scripts/glossary.php?id=$word_id\"
                     onClick=\"return open_popup('scripts/glossary.php?
                     id=$word_id')\">$row_gloss[1]</a><br>";
        } }
     }
    echo "</font>";
}
// *** END display_gloss($gloss_arr) ***

?>
```

To call these functions you need to:
1) include in any file the line: **include("glossary.inc.php");**
2) invoke the functions with: $content=glossary_parse($content)where $content is a variable for an actual text;
or preferably: **add_glossary($id_article);** where **$id_article** is the identification number of the article to be parsed. Identification numbers are given in the table **echo_articles.**
$gloss_arr the array (list) of ids of matching words.

Result:
Function 1 will display the text with incorporated links to glossary.php
Function 2 will display the list of glossary-entries wherever it is called, and link to glossary.php

Additionnally, the link to the glossary definition window could be implemented as a Javascript popup-window:

Replace the link to glossary.php with:

<a href=\"glossary.php?id=$word_id\" onClick=\"return
open_popup('glossary.php?id=$word_id')\">$row_gloss[1]

include in the head-part of each file in which a call to glossary function is made the following Javascript:

```
<script language="JavaScript">
function open_popup(page)
{
    window.open(page,'','width=300,height=300');
    return false;
}
</script>
```

Backffocie management

Bakcoffice provided: YES, fully implemented

Media handling

Media handling includes administration of images, sounds, movies and animations that are displayed within the various society records, as well as the metadata that accompanies the media.

Three levels in the logical structure have to be distinguished:

1) media handling in the database: location and description of media files are stored in the database, the media files themselves however are not. They are located inside the media folder of the xml part of the website.

2) media handling in the backoffice (includes database and real media file). The proposed backoffice does simplify the insertion and management of media in the database and in the filesystem at the same time in an “in-transparent” manner. In-transparent means that the user does not have to worry about what is actually inserted into the database and what is physically added to the filesystem.

3) xml cataloguing of database and directory content (see XML part of this document).

8. Search Engines

Search engines are an important part of the prototype. They provide the user and visitor with a means to access information in a filtered mode. The search engines prepared are divided into three parts.

1) Part 1 is a simple browse search engine, which displays indexes to the database content in an organized manner. This part has been prepared for those users that are not sure what they are exactly looking for.

2) Part 2 is a website search engine. This search engine does not search information in the database itself, but, after indexing, established its own database with the phenotypical content of the website. With this search engine, auxiliary information, such as user comments left on the website, are indexed and become searchable as well.

3) Part 3 is a database search engine. It allows users to search for specific expressions and features in the database.

Part 1: Browse search engine

NOT YET IMPLEMENTED

Part 2: Website search engine

Description

The website search engine indexes all pages of the website as they actually appear and inserts these indexes in its own database tables and text files. It's principles are similar to those of the famous httdig:// search engine available in Perl. However, in order to maintain a degree of coherence, I have chosen to adapt the excellent PHP search engine PhpDig 1.4.4c and to incorporate it into the proto-type.

Part 2 of the search engines, the Website search engine, is therefore an adaptation of the scripts created by Antoine Bajolet. The original scripts and documentation is available at: `http://phpdig.toiletoine.net/`

NOTE: I have adapted the original scripts to match the proto-type, it's website and database. Simply replacing my adapted scripts with the original phpdig scripts will not work. Adaptation mostly concerns search specific routines and table creation commands.

Search results are displayed using a template based on `aq` designed by `PhpSysInfo` (`http://phpsysinfo.sourceforge.net/`). This template was available in the original `phpdig` set, but has been adapted as well.

Administration

The entire website search engine is contained in the folder `search`. Within this folder, you'll find the folder `admin`. This is the location from which you can updated the indexes:
`http://<yourhost>/search/admin/`
You need to be a phpdig specified administrator to be able to access the administration area. This is set in the file `/search/includes/config.php`
You also need to modify the MySQL connection parameters in:
/search/includes/connect.php

Limitations

As per 11 February 2003, there is a bug in my adaptation of the search engine to the proto-type. This bug is only reflected in the admin zone when updating the indexes. A mysql command error message is displayed during indexation. I hope to be able to resolve this issue soon. Note, however, that this error message has no influences on the indexing process, nor does it influence the search processes. As such, it can be called a "benign bug".

Permissions and security

NOTE: The following folder must be writable by your webserver (www or nobody):

`/..root../search/text_content/`

Please modify the permissions for this folder to

```
chmod a+w text_content
```

However, for security reasons**, make the folder not-executable with:**

```
chmod a-x text_content
```

Backoffice:

Fully implemented, see Administration above

Part 3: Database search engine

NOT YET IMPLEMENTED

9. Backoffice structure

A handbook for the use of the backoffice has been compiled and is downloadable at:
http://www.ehess.fr/centres/logis/necep/
Content providers are URGED to read the handbook.
(it is added as Appendix to this report)

What is the backoffice?

Backoffice is an area of the website in which only authorized users can proceed. The user's identity is checked against a username and a password, both of which are stored in the `echo database` in the `table users`. Once identified, the visitor can proceed to other parts of the backoffice and manage the content of the database, and thus indirectly the content of the website.

In the descriptions of the various sections of the backoffice below, the file-contents are not listed in order to keep this documents within a reasonable limit. The file headers only are reproduced, including the list of functions that are available in a file. The entire backoffice is available electronically.

How does it work?

The visitor is asked to enter his username and password (both will be given to him by the administrator). A PHP script tests if username and password are indeed corresponding to a registered user. If the result is positive, then a cookie is sent to the browser and the user can proceed in managing the website and database.
The cookie is **time limited**. It is currently set to last 1 (one) hour. After this time, the user is asked to log in again and identify himself. This time limit has been added to avoid misuses. Indeed, imagine you are logged into the ECHO database and, while having lunch, someone maliciously modifies the content of the database.

The Backoffice index page contains the following header (the full script is not reproduced below):

```
/*
FILE:           backoffice/index.php
AUTHOR:         Laurent Dousset
PROJECT:        ECHO
DATE CREATED:   26 November 2002

DESCRIPTION:    index page of backoffice

CONTAINS:
function login_form();
function check_user($user_echo,$password_echo)
function print_menu()
*/
```

The principle is as follows. When calling this page, the script checks if a cookie has been set. If not, it calls the function `login_form()`, where the user types his username and password. These are then sent to the function `check_user(…)`, where the entry is compared with the

users database in echo. If the user exists, then `print_menu()` displays what the user is allowed to do, such as manage the glossary etc., and a time-limited cookie is sent to the browser.

If the visitor tries to access any other page of the backoffice, but has no cookie in his browser record, he is automatically sent to the backoffice/index.php page to log in. This is done through the following test on each page (where `$ECHO_login` is the name of the cookie):

```
if (isset ($ECHO_login))
{
                                        [whatever is done on this page]
} else { header ("Location: index.php"); }
```

Users can, and should, logout when they do not wish to further work in the backoffice area. In this case, the page logout.php is called. This page has the following content (working principle: time limit of the cookie is simply set to a time previous to the current time, which causes the browser to delete the cookie and therefore the authorization):

```
<?php
    setcookie ("ECHO_login", "",time() - 3600);
    header ("Location: index.php");
?>
```

Particular contents of the Backoffice area

Glossary management

Glossary management is contained in the file: glossary_backoffice.php
This file does manage all functions relative to the Glossary, including:

- List all current glossary entries
- add a new definition/glossary entry
- change/edit the definition of an entry
- add a phenotype to a definition. A phenotype is the variation of a glossary entry; for example, totem and totemism are phenotypes of totemism.

The file's header is as follows (no modifications are needed in this file):

```
/*
FILE:           backoffice/glossary_backoffice.php
AUTHOR:         Laurent Dousset
PROJECT:        ECHO
DATE CREATED:   26 November 2002
DESCRIPTION:    Glossary management functions
CONTENTS:
function list_all()
function display_record($id)
function update_record($id,$root_word,$definition)
function new_definition()
function insert($root_word,$definition)
function add_pheno($id,$glossary_terms)
*/
```

User comments management

The management of the comments users and visitors leave for an article on the website is managed in the backoffice page comment_edition.php This file does managae all function relative to user comments including:

- list of all comments
- edition of particular comments
- deletion of particular comments

As with all scripts the updates in the database are immediately reflected on the webpage. The file's structure is as follows (no modifications are needed in this file):

```
if (isset ($ECHO_login))
{
                                    // checks for log in
                                    if (isset($action))
                                    {          if ($action == "delete")
                                               {     // then delete the
specific record/comment
                                                     if ($action == "edit")
                                               {// then display a form to edit the
   specific record/comment
                                                  if ($action == "update")
                                               {// update the new data obtained
   from action = edit
                                  }
// in any case, list here the full set of comments
// this part may have to be changed at a later stage, when the number of
comments becomes too large to be displyed on a simple page.

} else {
// if log in unsuccessfull, then send back to log in page
    header ("Location: index.php");
}


?>
```

Article and related resources management

The management of the articles and related resources in the backoffice is undertaken through the following script pages:

article_edition.php	to add, delete or modify articles
image_ext_add.php	to add, delete or modify links to external resources
image_modif.php	to delete or modify an image resource
image_upload.php	to add/upload an image resource to the server
sound_modif.php	to delete or modify a sound resource
sound_upload.php	to add/upload a sound resource to the server
table_edition.php	to create a table
table_modification.php	to modify a table

As with all scripts the updates in the database are immediately reflected on the webpage.
Following is only the example of article_edition.php

```
<?php

/*

FILE:           article_edition.php
AUTHOR:         Laurent dousset
PROJECT:        ECHO
DATE CREATED:   10 February 2003
DATE UPDATED:

DESCRIPTION:    To edit and delete articles
*/

if (isset ($ECHO_login))
{

    include ("../db_connect.inc.php");         // Establish connection to
database

    echo "<h2>Article management</h2>
    <font face=\"Arial\" size=\"1\">
    [<a href=\"index.php\">Back to Backoffice</a> | <a href=\"logout.php\">Log
out</a>]</font><hr>";

   print_form();

   if (isset($action))
   {


     if ($action == "search")
     {
        if ($search_term_soc != '')
        {  echo "<i>Search for <b>$search_term_soc</b> in Names of social
groups...</i><hr>";
           list_articles_for_society($search_term_soc,0,$soc_name); }
        else
        {
           echo "<i>Search for <b>$search_term_title</b> in article
titles...</i><hr>";
           list_articles_search_title($search_term_title); }
     }

     if ($action == "list_for_soc")
     {
          list_articles_for_society("",$soc_id,$soc_name);
     }

     // ***************************
     if ($action == "delete")
     {

        if ($checked)
        {
           $comment_delete = mysql_query ("DELETE FROM `echo_articles` where id
= '$id_article'")
              or die ("Sorry, there was a problem connecting to the
database.");
           echo "<br><b>The article has successfully been deleted!!!</b><br>";
        } else {

           echo "<script language=\"JavaScript\">
              <!--
              input_box=confirm(\"Are you sure you want to PERMANENTLY DELETE
this article?\");
              if (input_box == true)
```

```
               {
location.replace(\"article_edition.php?id_article=$id_article&action=delete&ch
ecked=true\");}
             else
                { history.back(); }
             -->
             </script>";
        }

     }

     // ***************************
     if ($action == "edit")
     {
        edit_article($id_article);

     }

     // ***************************
     if ($action == "new_article")
     {
        new_article($id_soc,$soc_name);

     }


     // ***************************
     if ($action == "insert")
     {
        $date_created = date("Ymd");
        $date_modified = date("Ymd");
        $article_update = mysql_query ("INSERT INTO `echo_articles` VALUES
('','$id_soc', '$order', '$title', '$article_text',
'$author','$date_created','$date_modified','$category')")
          or die ("Sorry, there was a problem connecting to the database.");
        echo "<br>...unless you got a <i>Sorry message</i> just above, the
article has been inserted!<br>";
        list_articles_for_society("",$id_soc,"");
     }


     // ***************************
     if ($action == "update")
     {
        $date_modified = date("Ymd");
        $article_update = mysql_query ("UPDATE `echo_articles` SET id_soc =
'$id_soc', `order` = '$order', title = '$title', article_text =
'$article_text', author = '$author', date_modified = '$date_modified',
category = '$category' where id = '$id'")
          or die ("Sorry, there was a problem connecting to the database.");
        echo "<br>...unless you got a <i>Sorry message</i> just above, the
article has been updated!<br>";
        list_articles_for_society("",$id_soc,"");
     }




   }



} else {
    header ("Location: index.php");
}
```

```
//*****************************************
function print_form()
{

   $comment_list = mysql_query ("SELECT name, id FROM `echo_societies` order by
name")
   or die ("Sorry, there was a problem connecting to the database. <br> Please
hit the back button of your browser and try again later.");

    $rows_number = mysql_numrows($comment_list);
    if ($rows_number > 0)
    {

     while ($row = mysql_fetch_row($comment_list))
     {

     echo "<b><a
href=\"article_edition.php?action=list_for_soc&soc_id=$row[1]&soc_name=$row[0]
\">$row[0]</a> | </b>";
     }
   } else
   {
     echo "Sorry, could not find any society with your search term. Try
again.";
   }

   echo "
   <br>
     <form name=\"search\" action=\"article_edition.php\" method=\"post\">
     Search all articles for a society. Enter (part of) the name of the
society:<br>
      <input type=\"text\" name=\"search_term_soc\" size=\"35\">
     <br><b>or</b><br>
     Search an article. Enter (part of) the title of an article:<br>
     <input type=\"text\" name=\"search_term_title\" size=\"35\">
     <input type=\"hidden\" value=\"search\" name=\"action\">
     <br><br><input type=\"submit\" name=\"submitButtonName\" value=\"Search
now\">
     </form><hr>
   ";


}


//*****************************************
function list_articles_for_society($search_term,$soc_id,$soc_name)
{

   if ($soc_id > 0)
   {

     $comment_list = mysql_query ("SELECT echo_societies.name,
echo_societies.id, echo_articles.* FROM `echo_societies`, `echo_articles`
WHERE echo_articles.id_soc = '$soc_id' AND echo_societies.id = '$soc_id' order
by `order`")
       or die ("Sorry, there was a problem connecting to the database. <br>
Please hit the back button of your browser and try again later.");
    } else {

     $search_term = "%".$search_term."%";

     $comment_list = mysql_query ("SELECT echo_societies.name,
echo_societies.id, echo_articles.* FROM `echo_societies`, `echo_articles`
WHERE echo_societies.name like '$search_term' AND echo_societies.id =
echo_articles.id_soc order by `order`")
```

```
        or die ("Sorry, there was a problem connecting to the database. <br>
Please hit the back button of your browser and try again later.");
   }

    $rows_number = mysql_numrows($comment_list);
    if ($rows_number > 0)
    {
     print_table($comment_list);
   } else
   {
      echo "Sorry, could not find any society with your search term. Try
again.<br>
      <b>Or would you like to <a
href=article_edition.php?action=new_article&id_soc=$soc_id&soc_name=$soc_name>
add an article for the $soc_name</a></b><br><br>";

   }
}

//*****************************************
function list_articles_search_title($search_term)
{

   $search_term = "%".$search_term."%";

    $comment_list = mysql_query ("SELECT echo_societies.name,
echo_societies.id, echo_articles.* FROM `echo_societies`, `echo_articles`
WHERE echo_articles.title like '$search_term' AND echo_articles.id_soc =
echo_societies.id order by `id_soc`, `order`")
   or die ("Sorry, there was a problem connecting to the database. <br> Please
hit the back button of your browser and try again later.");

    $rows_number = mysql_numrows($comment_list);
    if ($rows_number > 0)
    {
     print_table($comment_list);
   } else
   {
      echo "Sorry, could not find any title with your search term. Try again.";
   }
}

//*****************************************
function print_table($comment_list)
{


   echo "<table border=\"1\" cellpadding=\"4\" cellspacing=\"0\" width=\100%\"
bgcolor=\"#FFFF99\"><tr>
           <td><b>Society (id)</b></td>
           <td><b>id article</b></td>
           <td><b>id _society</b></td>
           <td><b>order</b></td>
           <td><b>title</b></td>
           <td colspan=\"2\" align=\"center\"><b><i>Action</i></b></td>

        </tr>";

      $number = 0;
      while ($row = mysql_fetch_row($comment_list))
      {

           $soc_name = $row[0];
           $soc_id = $row[1];

           $number++;
           if ($number % 2 == 0)
```

```
            { $color = "#E0FFFF"; }
          else { $color = "#E6E6FA"; }

          echo "<tr bgcolor=\"$color\">
                <td>$row[0] ($row[1])</td>
                <td>$row[2]</td>
                <td>$row[3]</td>
                <td>$row[4]</td>
                <td>$row[5]</td>
                <td><a
href=\"article_edition.php?action=edit&id_article=$row[2]\"><b>edit</b> this
article</a></td>
                <td><a
href=\"article_edition.php?action=delete&id_article=$row[2]\"><b>delete</b>
this article</a></td>
           </tr>";
     }

     echo "</table><p><b><a
href=article_edition.php?action=new_article&id_soc=$soc_id&soc_name=$soc_name>
add a new article for the $soc_name</a></b></p>";
}

//*****************************************
function edit_article($id_article)
{
   $article_edit = mysql_query ("Select * from `echo_articles` where id =
'$id_article'")
     or die ("Sorry, there was a problem connecting to the database.");
   $rows_edit = mysql_numrows($article_edit);
   if ($rows_edit > 0)
   {
     while ($row = mysql_fetch_row($article_edit))
     {

     echo "

            <form name=\"update_article\" action=\"article_edition.php\"
method=\"post\">
                <p>
                <h3>You may modify the article content here. <br>Please hit
the Update button below once you're finished.</h3>
                If you are not sure, don't hit the button but simply search
for a new term/title/society in the form above.<p></p>
                <table border=\"1\" cellpadding=\"5\" cellspacing=\"0\">

                <tr valign=\"top\" bgcolor=\"#ccff99\">
                   <tr valign=\"top\" bgcolor=\"#ccff99\"><td>id:
<b>$row[0]</b></td></tr>
                   <tr valign=\"top\" bgcolor=\"#ccff99\"><td>id society:
<input type=\"text\" value=\"$row[1]\" name=\"id_soc\" size=\"4\">
                   <br><b>(only change this if your are absolutely sure that
this article is attributed to the wrong society!!!)</b></td></tr>
                   <tr valign=\"top\" bgcolor=\"#ccff99\"><td>order of
appearance: <input type=\"text\" value=\"$row[2]\" name=\"order\" size=\"11\">
                   <br><b>(this determines in which order the articles should
be read. 1 is the first article, 2 the second etc.)</b></td></tr>
                   <tr valign=\"top\" bgcolor=\"#ccff99\"><td>title:
<br><textarea name=\"title\" cols=\"80\"
rows=\"2\">$row[3]</textarea></td></tr>
                   <tr valign=\"top\" bgcolor=\"#ccff99\"><td>article text:
<br><textarea name=\"article_text\" cols=\"80\"
rows=\"20\">$row[4]</textarea></td></tr>

                   <tr valign=\"top\" bgcolor=\"#ccff99\"><td>Author: <input
type=\"text\" value=\"$row[5]\" name=\"author\" size=\"50\">
```

```
                  <tr valign=\"top\" bgcolor=\"#ccff99\"><td>

                  Category:
                  Please choose one from the pop-up menu
                          <select name=\"category\" size=\"1\">
                          <option value=\"$row[8]\">$row[8]
                  <option value=\"Introduction\">Introduction
                  <option value=\"History\">History
                  <option value=\"Language\">Language
                  <option value=\"Social Organization\">Social Organization
                  <option value=\"Political Organization\">Political
Organization
                  <option value=\"Economic System\">Economic System
                  <option value=\"Religious System\">Religious System
                  <option value=\"Material Culture\">Material Culture
                  <option value=\"Ecology, Habitat, Subsistence\">Ecology,
Habitat, Subsistence
                  </select>




                     </td></tr>
                  </tr>
                  <tr bgcolor=\"#ffffcc\">
                     <input type=\"hidden\" value=\"$row[0]\" name=\"id\">
                     <input type=\"hidden\" value=\"update\" name=\"action\">
                     <td colspan=\"6\"><input type=\"submit\" value=\"Ok, update
this article\" name=\"submitButtonName\"></td></tr>
                  </tr>
                  </table>


               </form>
               <hr>";
            }
         }


}
//*****************************************

function new_article($soc_id,$soc_name)
{


     echo "

               <form name=\"insert_article\" action=\"article_edition.php\"
method=\"post\">
                  <p>
                  <h3>You may add a new article for the $soc_name here.
<br>Please hit the Insert button below once you're finished.</h3>
                  <p></p>
                  <table border=\"1\" cellpadding=\"5\" cellspacing=\"0\">

                  <tr valign=\"top\" bgcolor=\"#ccff99\">
                     <tr valign=\"top\" bgcolor=\"#ccff99\"><td>id:
<b>$row[0]</b></td></tr>
                     <tr valign=\"top\" bgcolor=\"#ccff99\"><td>id society:
<input type=\"text\" value=\"$soc_id\" name=\"id_soc\" size=\"4\">
                     <br><b>(only change this if your are absolutely sure about
what you are doing!!!)</b></td></tr>
                     <tr valign=\"top\" bgcolor=\"#ccff99\"><td>order of
appearance: <input type=\"text\" value=\"$row[2]\" name=\"order\" size=\"11\">
```

```
                  <br><b>(this determines in which order the articles should
be read. 1 is the first article, 2 the second etc.)</b></td></tr>
                  <tr valign=\"top\" bgcolor=\"#ccff99\"><td>title:
<br><textarea name=\"title\" cols=\"80\"
rows=\"2\">$row[3]</textarea></td></tr>
                  <tr valign=\"top\" bgcolor=\"#ccff99\"><td>article text:
<br><textarea name=\"article_text\" cols=\"80\"
rows=\"20\">$row[4]</textarea></td></tr>

                  <tr valign=\"top\" bgcolor=\"#ccff99\"><td>Author: <input
type=\"text\" value=\"$row[5]\" name=\"author\" size=\"50\">

                  <tr valign=\"top\" bgcolor=\"#ccff99\"><td>Category:
                  Please choose one from the pop-up menu
                          <select name=\"category\" size=\"1\">
                  <option value=\"Introduction\">Introduction
                  <option value=\"History\">History
                  <option value=\"Language\">Language
                  <option value=\"Social Organization\">Social Organization
                  <option value=\"Political Organization\">Political
Organization
                  <option value=\"Economic System\">Economic System
                  <option value=\"Religious System\">Religious System
                  <option value=\"Material Culture\">Material Culture
                  <option value=\"Ecology, Habitat, Subsistence\">Ecology,
Habitat, Subsistence
                  </select>

               </td></tr>
               </tr>
               <tr bgcolor=\"#ffffcc\">
                  <input type=\"hidden\" value=\"$row[0]\" name=\"id\">
                  <input type=\"hidden\" value=\"insert\" name=\"action\">
                  <td colspan=\"6\"><input type=\"submit\" value=\"Ok, INSERT
this article\" name=\"submitButtonName\"></td></tr>
               </tr>
               </table>

            </form>
            <hr>";
}

?>
```

10. XML management

The entire content of the database is accessible through the PHP web interface for “direct access”. However, it is also duplicated into xml files with predefined and **private** DTDs. The motive behind this duplication is the possibility to have updated xml files whenever the database itself has been modified. Modifying data in a database is a simple process, while modification of xml files is more complicated and time-consuming. The duplication of the database content into xml files is called **xml management** in this document.

The central piece of the XML management are PHP scripts that export the parsed database content into XML files following a pre-specified DTD.

The PHP scripts for creating such files are rather simple:

1)select all elements of database table.
2)For each of these elements, write the *content* into a text file with extension .xml.
3) *Content* is defined as a line per line write-to-file with, for example:

```
$line = "<tag>".$database_element."</tag";
```

4)Additionally, write an index file containing the list of all xml files of a group of elements and their respective filenames (as in point 2 above).

XML directory: how to proceed

In all examples below, $base_url is as follows:

http://www.ehess.fr/centres/logis/necep/

Note that the variable `$base_url` is not mentioned anymore below, but is implicit whenever the starting directory is `/xml/`. Hence, if you see an url such as
`/xml/societies/society_id.xml`
you should read:
`$base_url/xml/societies/society_id.xml`

and, following the current hosting of NECEP:
http://www.ehess.fr/centres/logis/necep/xml/societies/society_id.xml

---

How to proceed?

Case A: you know the society’s id

If you know the society’s identification number (**id**), you may go straight to
`/xml/society_id.xml`

In `society_id.xml`, you will find the list of tables and articles for this society, as well as other elements such as identification elements and facts sheet elements (see below).
You may fetch from this file the dynamic or xml file of the articles and tables.

Case B: you don't know the society's id

Either use `map_names.xml` at `/xml/` to find the canonical name and the id of a society, in case you have an alternative name or spelling for this society which you can use for a search in `map_names.xml`

Or use any other method to match elements (country, language, continent etc.) which are located in each `/xml/societies/society_id.xml` file within the tag **<identification>**.

To know how many and which are the files (societies) located in `/xml/society/` simply fetch `/xml/societies/index.xml` which contains the name and the url of the xml file for each listed file (society).

---

filename: **map_names.xml**

location: $base_url/xml/

Idea: This is a mapping file for alternative names. If you have a name/word that looks like the name of a society, you may try to see if this name/word does match one of the <altname> tags in this file. If it does, you'll get the canonical name, the id and the xml file location for the correct society.

Use: Once you have found an alternative name, you know the id of a society (item attribute id), the canonical name of the society (item attribute canonical), the url to the dynamic webpage for this society (item attribute dynamic_url), and the url to the xml file for this society (item attribute xml_url).

Structure:

```
<names_map>
   <item id="id" canonical="name of society" dynamic_url="php url"
                              xml_url="xml url of society">
      <altname>alternative name 1</altname>
      <altname>alternative name 2</altname>
      ...etc...
   </item>
   <item id="id2" ... etc ...
```

---

filename: **society_*n*.xml**

location: $base_url/xml/societies/

Idea: This is the central xml file for a society. It contains elements of identification and indexes of articles, tables etc. for each society.

Use: Once you have found the identification number of a society, simply replace the ***n*** in the **society_*n*.xml** filename to retrieve the file. Within the societies folder, you'll find an index.xml file which lists all available **society_*n*.xml** files.

Structure (only the most important tags are reproduced):

```
<society id="n" name="the canonical name">
    <dynamic_url>url of dynamic homepage for this society</dynamic_url>

    <identification>
        <all_names>comma-separated list of names</all_names>
        <language>language name</language>
        <continent>continent from official list (MPI)</continent>
        <country>country from official list (MPI)</country>
    </identification>

    <article_list>
        <article id="n">
            <title>Title of the article</title>
            <dynamic_url>dynamic url of the article</dynamic_url>
            <xml_url>xml file url of the article</xml_url>
        </article>
        <article id="n2">
            ... etc...
        </article>
    </article_list>

    <table_list>
        <table id="n">
        ...exactly same elements as articles above...
        </table>
    </table_list>

</society>
```

---

filename: **article_*n*.xml**

location: $base_url/xml/articles/

Idea: This file contains a prose article for a society. There may be various articles for one society, but each article relates only to one society.

Use: Once you have found the identification number of an article, simply replace the ***n*** in the **article_*n*.xml** filename to retrieve the file. Within the articles folder, you'll find an index.xml file which lists all available **article_*n*.xml** files. To find all article files related to a society, you need to retrieve the **society_*n*.xml** file first and check the <**article_list**> tag (see above)

Structure:

```
<article id="n">
    <title>Its title</title>
    <dynamic_url>the url of the dynamic homepage</dynamic_url>
    <society_id>gives the identification number of the society</society_id>
    <society_name>canonical name of the society</society_name>
    <society_xml>the xml file name for the society</society_xml>
    <article_text>Here is the content of the article itself</article_text>
    <author>Its author</author>
    <date_created>format is yyyy-mm-dd</date_created>
    <date_modified> format is yyyy-mm-dd</date_modified>
</article>
```

Directory structure of the xml version

The xml version of the database content is stored in a predefined and stable file system with the following structure.

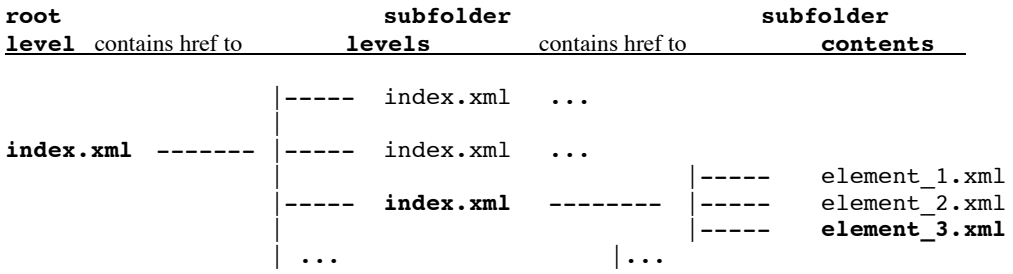
In each subfolder, there is a file named `index.xml` which has links to each file of this subfolder and can be interpreted as its table of contents. All `index.xml` of all subfolders have the same structure and refer to the same DTD which is located in `/xml/index.dtd` (see below).

Root elements

`/xml`	contains the entire xml content. In the PHP scripts, the url to this folder is stored in the variable `$base_url` in an absolute mode; for example:
	`$base_url = "http://www.echo.net/xml/"`
	
	The equivalent path for PHP output functions is stored in `$base_path`
	
	All subsequent links are expressed as a concatenation with `$base_url`.
	For example, the file …`/xml/glossary/index.xml` is expressed as:
	`$url = $base_url."glossary/index.xml"`
	
	In the xml files themselves, links to other xml files are expressed using the entire http path in an absolute mode. For example as:
	
	`http://www.echo.net/xml/glossary/index.xml`
	(idem for DTD declarations)
	
	To **adapt** `$base_url` and `$base_path` **to your local settings**, modify the variables in `backoffice/xml/config.inc.php`

`/xml/map_names.xml` `/xml/map_names.dtd`	This a mapping file. It does give alternative names of societies following the canonical name and the url pointing to the correct society record. This file can be used for search engines as a society-thesaurus (see details below).

/xml/index.dtd	contains the document type declaration of all index.xml in each subsequent folder
/xml/index.xml	contains the index and links to all index.xml of subsequent folders. **This is the most important file of the entire xml folder. Indeed, from the content of this …/xml/index.xml file, the entire tree system is named and can be accessed without any previous knowledge of the xml directory structure. See the following figure for a schema with in bold the example of path to element_3.xml**



Subfolders of the XML directory system

Glossary management

xml/glossary/	contains an xml file for each glossary entry and their DTD.
xml/glossary/index.xml	contains the reference of each glossary_n.xml. It's DTD is /xml/index.dtd
xml/glossary/glossary_n.xml	n stands for the identification number of the glossary entry. There is one file for each glossary entry. The index is in index.xml

Article management

xml/articles/	contains an xml file for each article entry and their DTD.
xml/glossary/index.xml	contains the reference of each article_n.xml. It's DTD is /xml/index.dtd
xml/glossary/article_n.xml	n stands for the identification number of the article entry. There is one file for each article. The index is in index.xml

Media management (multi-media)

xml/media/	Contains actual media files (movies, sounds and images).

xml	This is a shared folder between the php version and version of the website. It is the container for all non-text type files located in the project. Metadata for these files is located in `/xml/media_metadata/`. The index is in `index.xml`
`xml/media_metadata/`	Contains descriptions and of media files located in /xml/media/ — The index is in `index.xml`

Structure of xml files and corresponding DTDs

index.xml and index.dtd

There is an `index.xml` in the xml root folder, as well as in each subfolders, even when there is no direct call to such an `index.xml`. For example, in the images folder, there is an `index.xml` which is the table of contents of this folder, even if there is no need for such a table in the web interface.
All `index.xml` of all subfolders, including that of the xml root folder, have an identical structure and identical tag names (of course, the DOCTYPE declaration depends on the `$base_url` variable you have set in `/backoffice/xml/config.inc.php`)

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE index SYSTEM "http://127.0.0.1/echo/xml/index.dtd">
<index>
   <index_element>
      <url name="[url to xml file]">
         <Hyperlink>[name of link/element]</Hyperlink>
      </url>
   </index_element>
   <index_element>
      <url name="[url to xml file]">
         <Hyperlink>[name of link/element]</Hyperlink>
      </url>
   </index_element>
[etc... depending on the number of files in the folder]
</index>
```

All `index.xml` files make reference to one and the same private `index.dtd`, which is located in the root of the xml folder. This dtd has the following content:

```
<!ELEMENT url (#PCDATA | Hyperlink)*>
<!ATTLIST url name CDATA #IMPLIED>
<!ELEMENT index (#PCDATA | index_element)*>
<!ELEMENT index_element (#PCDATA | url)*>
<!ELEMENT Hyperlink (#PCDATA)>
```

map_names.xml and map_names.dtd

Definition :

This xml file is a repository of all alternative names used for all societies listed in the NECEP database.

Reason :

It is not true to say that it is easy and unambiguous to identify a society through its name. First of all there are various spellings found in the literature, but there also may be completely distinct appellations for one and the same group. This file shall help to overcome this problem in mapping the alternative names used in the literature to the NECEP canonical name of the society for further mapping to other databases.

Location:

`$base_url/xml/` where $base_url currently is at:
`http://www.ehess.fr/centres/logis/necep/`

Structure with two examples:

```
<names_map>
    <item id="4" canonical="Evenk"
          dynamic_url="http://127.0.0.1/ECHO/articles.php?id_soc=4"
          xml_url="http://127.0.0.1/ECHO/xml//societies/society_4.xml">
          <altname>Tungus</altname>
          <altname>Evenki</altname>
          <altname>Evengkil</altname>
    </item>

    <item id="1" canonical="Ngaatjatjarra"
          dynamic_url="http://127.0.0.1/ECHO/articles.php?id_soc=1"
          xml_url="http://127.0.0.1/ECHO/xml//societies/society_1.xml">
          <altname>Ngadajara</altname>
          <altname>Wenamba</altname>
          <altname>Ngaatjatjarra</altname>
    </item>
  </names_map>
```

Note that the attribute **id** is the identification number of the society in the NECEP database; that **canonical** is the name we have chosen to be “canonical” for this society; and that **url** points to the dynamic URL for this society.

DTD: (map_names.dtd)

```
<!ELEMENT names_map (#PCDATA | item)*>
<!ELEMENT item (#PCDATA | altname)*>
<!ATTLIST item id CDATA #IMPLIED>
<!ATTLIST item canonical CDATA #IMPLIED>
<!ATTLIST item dynamic_url CDATA #IMPLIED>
<!ATTLIST item xml_url CDATA #IMPLIED>
<!ELEMENT altname (#PCDATA)>
```

Glossary XML files

The glossary xml files are located in /xml/glossary/ . This folder also contains the `index.xml` of its content (see above), and the `glossary.dtd`
For each glossary entry in the database, the PHP export script creates an xml file. The filename of these xml files are reflecting the id number of the glossary entry in the database. For example, glossary entry *anthropology* with id=12 will be saved into `glossary_12.xml`

The `glossary_n.xml` files have the following structure:

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE glossary SYSTEM "http://127.0.0.1/echo/xml/glossary/glossary.dtd">

<glossary id="[id number of database entry]">
   <word>[name of element/word]</word>
   <definition>[definition of word as in database]</definition>
</glossary>
```

The corresponding DTD in `glossary.dtd` is as follows:

```
<!ELEMENT glossary (#PCDATA | word | definition)*>
<!ATTLIST glossary id CDATA #IMPLIED>
<!ELEMENT definition (#PCDATA)>
<!ELEMENT word (#PCDATA)>
```

PHP script elements

The PHP script for the glossary to xml export is located in the file:
`/backoffice/xml/export_glossary.php`

Herewith I only reproduce the particular lines that write the database record content into xml format:

Header and content of index.xml -- into variable $index_output

```
$index_output = "<?xml version=\"1.0\" encoding=\"UTF-8\" ?>\r<!DOCTYPE index
SYSTEM \"".$base_url."index.dtd\">\r<index>\r";
```

```
$index_output = $index_output."<index_element><url
name=\"".$base_url."glossary/".$gloss_href."\"><Hyperlink>".$gloss_word."</Hyp
erlink></url></index_element>\r".[etc...]</index>;
```

Header and content of glossary_n.xml -- into variable $out_put

```
$out_put = "<?xml version=\"1.0\" encoding=\"UTF-8\" ?>\r<!DOCTYPE glossary
SYSTEM \"".$base_url."glossary/glossary.dtd\">\r\r<glossary
id=\"".$row[0]."\">\r<word>".$row[1]."</word>\r<definition>".$row[2]."
</definition>\r</glossary>\r";
```

The remaining elements of the script are basically SQL `select` commands from the database and file writing functions (including log-in/cookie verifications).

Articles XML files

The article xml files are located in /xml/articles/ . This folder also contains the `index.xml` of its content (see above), and the `article.dtd`
For each glossary entry in the database, the PHP export script creates an xml file. The filename of these xml files are reflecting the id number of the article entry in the database. For example, article entry with id=12 will be saved into `article_12.xml`

The `article_n.xml` files have the following structure:

```
<?xml version="1.0" encoding="iso-8859-1" ?>
  <!DOCTYPE article (View Source for full doctype...)>
<!--
- NECEP
- Non European Components of European Patrimony
- ECHO project
- http://127.0.0.1/ECHO/
- For queries, please contact:
- Laurent Dousset <dousset@ehess.fr>
  -->
<article id="9">
  <title>Test for Tongouse</title>
   <url>http://127.0.0.1/ECHO/articles.php?id_soc=5&id_article=9</url>
  <society_id>5</society_id>
  <society_name>Tongouse</society_name>
  <society_xml>http://127.0.0.1/ECHO/xml/societies/society_5.xml
      </society_xml>
  <article_text>Test for Tongouse</article_text>
  <author>Laurent Dousset</author>
  <date_created>2003-02-15</date_created>
  <date_modified>2003-04-01</date_modified>
</article>
```

The corresponding DTD in `glossary.dtd` is as follows:

```
<!ELEMENT date_modified (#PCDATA)>
<!ELEMENT article (#PCDATA | title | id_society | society_url | article_text |
author | date_created | date_modified)*>
<!ATTLIST article id CDATA #IMPLIED>
<!ELEMENT title (#PCDATA)>
<!ELEMENT article_text (#PCDATA)>
```

```
<!ELEMENT id_society (#PCDATA)>
<!ELEMENT date_created (#PCDATA)>
<!ELEMENT author (#PCDATA)>
<!ELEMENT society_url (#PCDATA)>
```

PHP script elements

The PHP script for the glossary to xml export is located in the file:
`/backoffice/xml/export_articles.php`

Herewith I only reproduce the particular lines that write the database record content into xml format:

Header and content of index.xml -- into variable $index_output

```
$index_output = "<?xml version=\"1.0\" encoding=\"iso-8859-1\" ?>\r<!DOCTYPE
index SYSTEM \"".$base_url."index.dtd\">\r<index>\r";

$index_output = $index_output."<index_element><url
name=\"".$base_url."articles/".$article_href."\"><Hyperlink>".$article_title."
</Hyperlink></url></index_element>\r";
```

Header and content of glossary_n.xml -- into variable $out_put

```
$out_put = "<?xml version=\"1.0\" encoding=\"iso-8859-1\" ?>\r<!DOCTYPE
article SYSTEM \"".$base_url."articles/article.dtd\">\r<article
id=\"".$row[0]."\">\r<title>".$row[3]."</title>\r<id_society>$row[1]</id_socie
ty>\r<society_url>$society_href</society_url>\r<article_text>".$row[4]."</arti
cle_text>\r<author>$row[5]</author>\r<date_created>$row[6]</date_created>\r<da
te_modified>$row[7]</date_modified></article>\r";
```

The remaining elements of the script are basically SQL `select` commands from the database and file writing functions (including log-in/cookie verifications).