elpub2002 Proceedings — J. A. Carvalho et. al. (Eds.) — VWF Berlin, 2002

Library::⋆ — a Toolkit for Digital Libraries

Alberto Manuel Simões and José João Almeida

Departamento de Informática
Universidade do Minho
{albie@alfarrabio., jj@}di.uminho.pt

Abstract. In last years the amount of digital documents has increased dramatically. Unfortunately the same did not occur with the structure and organization of the information. Traditionally we built a digital library using a catalog with documents' meta-information including a conceptual classification and an ontology of concepts.
In this document we present a set of modules to help in the task of building and maintaining a digital library. It includes a module to work with ontologies, a set of modules to handle specific catalog formats (like BibTEX), a module to define new catalog formats and a tool to integrate ontologies and multi-format catalogs in a web browse-able knowledge-base.

1 Introduction

In this document we define a digital library as

$$DigLib \overset{\triangle}{=} ontology \times (document^{\star} \times catalog)^{\star}$$

Almost all authors agree that the main point of a digital library are documents, no matter what type they are. In addition to those documents, we need a catalog which contains meta-information, building a common view over them.

The ontology (or ontologies) will be used to define the active set of classification terms and their semantic structure. It will be used to build semantic relations between documents.

In next sections we will present `Library::⋆` toolkit: a set of Perl modules for digital library management with some real digital library examples. All the referred modules can be downloaded from CPAN [1].

1.1 Catalogs

There are different kind of catalogs: you can add documents meta-information in a XML catalog, on a database or a text file. In fact, many other formats can be used as catalogs. An example is BibTEX which can be used as a catalog of articles or books.

We will define a catalog as a sequence of entries, each one containing at least a title, url and meta-data.

$$catalog \overset{\triangle}{=} (entry)^{\star} \equiv (title \times url \times metadata)^{\star}$$

In order to integrate all these kind of catalogs in a digital library, we need a common view over them. In section 2 we will present some tools to help maintaining different catalog types, and on 4, some ways to integrate them in the same digital library.

1.2 Ontology

Ontologies are semantic networks, where terms (representing concepts) are related with each other.

$$term \rightharpoonup (relation \rightharpoonup term^{\star})$$

As each document is cataloged, there are terms related to each document. This way, ontologies provide a conceptual navigation level, building relations between terms and documents, and between documents itself.

On section 3 we present an ontology tool.

2 Catalog tools

As said, catalogs maintain meta-information regarding each document, and there is no need to have all catalogs in the same format,from simple text or XML files to full relational databases. This diversity of formats makes it essential to have a common way to see them. This lead to the definition of a common interface (API) to use and extract information from this variety of catalogs.

2.1 API for defining new catalog types

`Library::Catalog` module provides a set of common catalog interfaces. These include support for generic XML based catalog, BibTEX as a catalog of publications or news as a catalog. Although these are built-in, the module can handle new catalog types defining the following functions:

- **Entry extraction** — given some catalog (e.g. a filename) return the sequence of catalog entries:

$$asList : catalog \longrightarrow entry^{\star}$$

- **Entry as HTML** — to build a world wide web view for each entry, there should be some way to convert it to HTML:

$$asHTML : entry \longrightarrow title \times url \times HTMLbody$$

  We use these three items separately to make the tool able to construct better looking pages, and to generate correct links.

- **Entry as LaTeX** — to create printed versions of the library catalog:

$$asLaTeX : entry \longrightarrow \text{LaTeX}$$

- **Entry as Text** — to remove all non-textual contents, and return text to be used in full text search over the catalog;

$$asText : entry \longrightarrow text$$

- **Keyword extraction** — this function should return a list of terms related with the document, to be used in conjunction with the ontology.

$$asRelations : entry \longrightarrow term^{\star}$$

- **Identifier extraction** — an unique identifier of the entry in the catalog to be used whenever an entry contents is changed but the cataloged object is the same.

$$asIdentifier : entry \longrightarrow identifier$$

With this set of function all catalog can be used in the same manner. Although we suggest the creation of these function, the module uses some heuristics to replace some of them.

2.2 An example: BibTEX

Publications indexes and search engines are very popular. One of the common way to organize publications is to use BibTEX files with the respective bibliography. If we want to make them available in a digital library, we can see BibTEX as a catalog.

Using `Library::Catalog::BibTeX` to process the file, the function `asList` would break the catalog into entries, very similar with:

```
@Book{WebCompanion,
   author = "Michel Goossens and Sebastian Rahtz",
   title = "The \LaTeX{} Web Companion: integrating \TeX{}, HTML \& XML",
   publisher ={Addison-Wesley},
   keywords = "latex, xml, html, web",
   year = 1999}
```

First, notice that we added a new field named **keywords** in order to associate terms with the document. Some other fields are often needed and suggested, as an URL one. Then, the module will use the **asText** function to convert the entry to search-able text:

```
Michel Goossens and Sebastian Rahtz The LaTeX Web
Companion: integrating TeX, HTML XML Addison-Wesley
latex xml html web 1999
```

The conversion to LaTeX will generate:

The LaTeX Web Companion: integrating TeX, HTML & XML
Michel Goossens and Sebastian Rahtz
Addison-Wesley 1999

The HTML conversion would return something similar for a browser. The relations extractor can extract the keywords, {latex, xml, html, web} and the identifier could be `WebCompanion`.

3 Ontology tools

To maintain an ontology, or graph of concepts, we chose to implement a thesaurus tool. In fact, although our tool is based on ISO monolingual and multilingual thesaurus [3, 4], it contains more information than a simple thesaurus making it sit near topic-maps [5]. This graph contains not only information about terms relations but also thesaurus meta information like defined relations, their ranges and domains as some other relations properties. All these information belongs to the nodes of the graph (as normal terms).

3.1 Defining a thesaurus

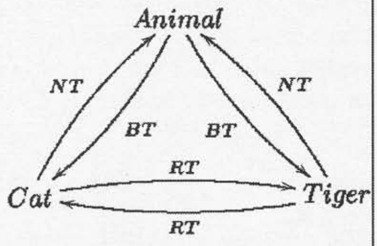
A thesaurus is a graph where nodes represent terms and arcs are relations between them. Standard relations are shown in table 1.

Using thesaurus meta-information, `Library::Thesaurus` is able to complete itself adding arcs based on mathematical properties of the relations (inverse, sym-

Table 1. ISO-2788 Standard Thesaurus Relations

Relation	Symmetric	Semantic
BT	NT	*A* is broader than *B* if it represents a bigger concept than *B*
RT	RT	This relation means that term concepts are related between them
USE	UF	We say *A* USE *B* when *A* and *B* represent the same concept and *B* was chosen as the preferred term
SN	—	Scope note relates a term with a brief description of the intended usage of the term/concept

Table 2. An example of an ISO-like thesaurus, and respective complete graph

`Cat`
`BT Animal`
`RT Tiger`
 	
`Tiger`
`BT Animal`
	

metric, reflexive). Table 2 shows an example, already in `Library::Thesaurus` notation, with relations between three terms: *Animal*, *Cat* and *Tiger* and it's internal representation.

3.2 Using `Library::Thesaurus`: defining meta-information

Thesaurus files are very close to ISO multi-lingue thesaurus syntax. The module can import and export this format, and has the capability to generate several views like LaTeX or a navigation-enabled HTML.

A typical thesaurus file for `Library::Thesaurus` contains information about terms and properties about relations.

Relations properties can be described using a meta-data command section in the top of the thesaurus file, or using meta-terms. In this article we will use the meta-data command syntax as it is slightly shorter.

- **Feedback Information** — whenever a relation needs to be printed, a description for the relation is searched. It can be defined in the top of the document using a command like `%desc BT Broader Term`;
- **Mathematical Properties** — it is possible to define mathematical properties like inversion of relations (`%inv POF HAS`) as well as symmetric, transitive, reflexive or anti-symmetric properties. They will make the thesaurus auto-completion active. This ability makes it easier to define the thesaurus without taking too much time checking relation coherence (because it can be done by software).
- **Range and Domains** — some relations (like the scope note or URL) map terms into other data type. The command `%external URL` defines it to have a different range than normal terms.
- **Multi-lingue support** — commands like `%lang EN PT FR` define active languages.

When you use an unknown relations in a meta-data command, this relation is created and can be used as any built-in relation.

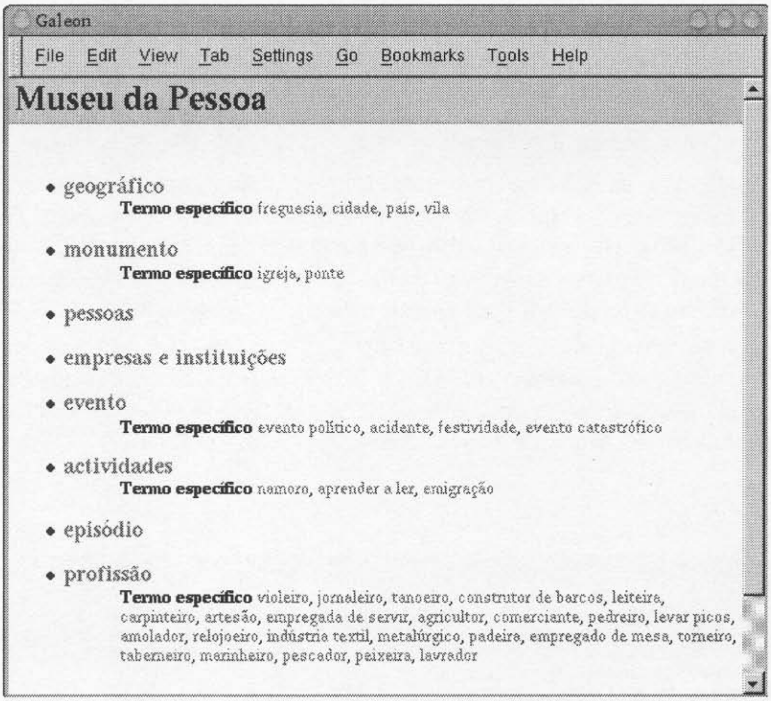
The definition of new relations is very important. In fact, looking at topic maps and some other ontology tools and formalisms, we see the appearance of new kind of relations like "about", "instance of" or "part of".

3.3 Writing programs using `Library::Thesaurus`

This module provides an API for managing terms or relations, and an API for structure based processing tools.

In the first set of tools we can see functions to add or remove terms, or define relation properties. These functions are useful inside the module but also makes possible for external programs to change a thesaurus without user interaction. In the structure based processing tools we can see:

- **Down Translate** — a generic user defined structure oriented processing tools for the thesaurus. The user defines what to do with terms, relations, special relation types, and down translates will traverse the thesaurus tree and return the processing result;
- **Transitive Closure** — calculates the transitive closure using the set of relation specified.
- **Conversion Methods** — consists of three methods to convert the thesaurus to HTML, XML or LaTeX. They are based on down translation function.
- **Navigation** — can be done using the `navigate` method which can be incorporated into a CGI, and turn a simple thesaurus on an interactive [6] HTML based conceptual navigation. Figure 1 shows a thesaurus being navigated.



Museu da Pessoa

- geográfico
  Termo específico freguesia, cidade, país, vila
- monumento
  Termo específico igreja, ponte
- pessoas
- empresas e instituições
- evento
  Termo específico evento político, acidente, festividade, evento catastrófico
- actividades
  Termo específico namoro, aprender a ler, emigração
- episódio
- profissão
  Termo específico violeiro, jornaleiro, tanoeiro, construtor de barcos, leiteira, carpinteiro, artesão, empregada de servir, agricultor, comerciante, pedreiro, levar picos, amolador, relojoeiro, indústria textil, metalúrgico, padeira, empregado de mesa, torneiro, taberneiro, marinheiro, pescador, peixeira, lavrador

Fig. 1. *A thesaurus being navigated*

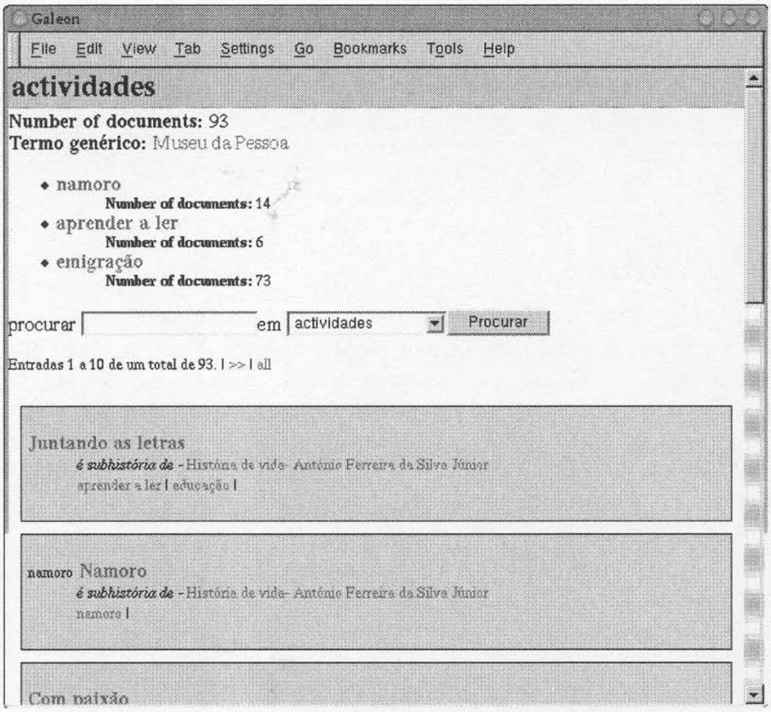
4 `Library::Simple`

Having catalog interfaces and an ontology, `Library::Simple` to construct a digital library. This process is a full catalog traversal where each entry found is processed to extract information: HTML appearance, LaTeX appearance, full entry text and term relations. This information will be stored in a set of index databases. The process of creation of the digital library databases consists of:

- for each catalog, extract a list of entries;
- create an index mapping entries to catalogs;
- convert each entry to HTML and LaTeX, saving the results on two Berkeley Database files;
- for each entry, extract plain text contents, and relations;
- count the number of documents fitting on each concept;
- create a log with terms found as relations which are not in the ontology;
- create a log with non-reachable entries from the ontology;

To create all this the user must supply a configuration file. This file contains information about the ontology used, an identifier for the digital library and a set of catalogs configuration. For each catalog the user must set it's type (in case it is a known type) or must supply the functions shown in section 2. After the digital library has been created, it can be used to do:

- **Navigation** — as with the thesaurus tool, it is possible to navigate the digital library using a CGI. This navigation (see figure 2) joins the thesaurus oriented navigation with the presentation of the documents related to all terms the current term transitive closure. To this concept oriented navigation, we added an HTML form to do full text search on catalog/ontology;
- **Report Generation** — using the ontology to guide the report structure, we created a tool to generate HTML or LaTeX reports of cataloged documents. Reports are organized mapping a part of ontology tree to the document tree and including document entries in the correspondent areas.



actividades

Number of documents: 93
Termo genérico: Museu da Pessoa

- namoro
  Number of documents: 14
- aprender a ler
  Number of documents: 6
- emigração
  Number of documents: 73

procurar em actividades

Entradas 1 a 10 de um total de 93. | >> | all

Juntando as letras
é subhistória de - História de vida- António Ferreira da Silva Júnior
aprender a ler | educação |

namoro Namoro
é subhistória de - História de vida- António Ferreira da Silva Júnior
namoro |

Com paixão

Fig. 2. *A digital library being navigated*

5 Conclusion

This set of modules proven to be crucial to maintain digital libraries. In special, the catalog format independence makes possible the construction of digital libraries from almost all kind of information.

We have been using it in several projects, since software, photographic and heterogeneous digital libraries. The most heterogeneous one joins catalogs from structured XML to chaotic HTML indexes pages. It contains 4000 documents spread by 20 catalogs of about 10 different syntaxes. In fact, this project has a very diverse contributor list from informatics to biologists, all them using quite different software platforms. Currently, we are working on:

- integration between concept navigation, catalog text searching and `ht://Dig` [2] tool;
- meta-thesaurus tool to use in conjunction the ontology and the catalog related terms to enrich automatically the thesaurus;
- base the report creation tool on a document template to make possible custom document organization;
- Use the process of digital library recalculation to extract news, which can be used to create a column on a digital library portal;

References

1. Comprehensive perl archive network. `http://www.cpan.org`.
2. `ht://dig` - www search engine software. `http://www.htdig.org`.
3. *ISO 2788 – Guidelines for the establishment & development of monolingual thesauri.* International Organization for Standardization.
4. *ISO 5964 – Guidelines for the establishment & development of multilingual thesauri.* International Organization for Standardization.
5. Steve Pepper. The tao of the topic maps. In *XMLEurope*, pages 229–235, 2000.
6. Pollard, Richard. *Hypertext presentation of thesauri used in online searching.* Graduate School of Library & Information Science — University of Tennessee.