COPA
A coding schema optimized for the hierarchical access to information

Javier Masa, Diego R. López

RedIRIS

E-mail: {javier.masa,diego.lopez}@rediris.es

1. Introduction

During the past years, RedIRIS has been offering services for the location of persons, groups, and activities related to a certain knowledge area. The main problem in the implementation of these services has been the need for a system to simply and efficiently perform searches in hierarchical or semi-hierarchical structures (such as classifications and ontologies) at different levels. COPA has been designed as a solution to this problem.

COPA is a coding schema that allows effective searches in hierarchical structures, at one level or recursively, and do not impose a significative complication to existing storage and retrieval systems (databases, directories,...) . The system is able to perform searches using metadata that is included in the pre-existing structure.

2. The COPA coding schema

The basic idea is the creation of string identifiers made of the concatenation of special sets of symbols, that incorporate hierarchy position data in them. These identifiers are added to the data available for a certain element (attributes in a directory object, for example).

COPA coding uses a letter (`l`) and several numbers (`n`) per each hierarchy level, so a COPA code has an aspect like this: `lnnlnnlnnlnnlnn`. The letter identifies a certain level within the hierarchy and the numbers conform the identifier for an element at that level. For example, the code `a02b05c01` refers to the second node (`02`) at the first level (`a`), the fifth node (`05`) at the second level (`b`), and the first node (`01`) at the third level (`c`). This regular structure simplifies the definition of both positive and negative filters, making very easy the expression of complex searches inside the virtual hierarchy defined by the codes.

The COPA coding scheme can be applied to any hierarchical naming structure, using the COPA code at the "leaf" part of the name. Let's consider the case of uniform resource names (URNs): the above example can be part of an URN (using the prefixes assigned to RedIRIS) like this:

```
    urn:mace:rediris.es:copa:classificationID:a02b05c01
```

3. What can be achieved using COPA

Although COPA was initially created to ease recursive searches at different levels in a hierarchical structure, it can also be used to enable intelligent navigation systems that use different views of this kind of structures, to maintain ontologies, to identify and manage groups, etc. The COPA codes have to be added as metadata to the elements defining the information structure and its components.

3.1. Virtual views. Ontologies

One of the main features of COPA is that it allows the decoupling of the actual representation of the information from the view that is offered at a given time. A hierarchical classification in knowledge areas can be maintained on a flat structure or on a structure based on organizational criteria. The knowledge area hierarchy is kept by the metainformation using COPA codes. This way, the (possibly multiple) affiliation of an object to a certain area is represented in the object itself, not by means of its position in the hierarchy (as it is the common approach up to now).

A flat structure in the information tree is key for maintaining the independence of entries with respect to the position of the objects related to them, so their position(s) within the structure may be changed without actually moving them in the supporting system. A change in the corresponding attribute(s) is enough.

Let's assume a COPA-based hierarchical structure of knowledge areas, established in terms of different levels, more specific as the depth in the structure increases. The following examples shows the codes corresponding to a path from a node at the top level in the structure (`a01`) to the branch at `a01b01c08d06`

Code	Name
a01	Sciences
a01b01	Biology
a02b01c08	Entomology
a02b01c08d06	Insect Taxonomy

Let's assume that this structure is stored in an LDAP server where all the entries corresponding to the area definitions are under `dc=areas,dc=org,dc=es`. Each entry has two attributes: the COPA code and the name of the knowledge area.

It is simple to build a program able to navigate the directory showing the hierarchical structure defined by the COPA codes: it just has to search for the areas at the same level and show them.

The first level can be covered by an LDAP filter including those entries with a code attribute starting with `a` but without `b`:

```
    (&(code=a*)(!(code=a*b*)))
```

Once the entries at the first level are shown, if the user selects Sciences (`code=a01`), the filter to be applied must include those entries with a code starting with `a01b` but without a `c`, that is:

```
    (&(code=a01b*)(!(code=a01b*c*)))
```

It is easy to imagine the process for the rest of levels.

It is also possible to build different (and orthogonal) virtual views by means of other COPA-coded attributes representing different hierarchies.

Building ontologies, in which the relationships among areas are richer than the simple specialization/generalization provided by the hierarchy, can be accomplished by adding other attributes to the entries, each one modeling a different relationship. These attributes will hold the code of the areas related to the one represented by the entry. And this can, of course, be mixed with the multiple virtual views...

3.2. Searching at different levels in the hierarchy

Let's consider now a corporate directory where all the objects (persons, departments,...) in the directory system have attributes with COPA-coded metadata. Each of these attributes classifies the object in a certain area, or puts it at a certain position in the organizational structure.

To find all objects related to Entomology we must search for those objects with `code=a01b01c08`. To find objects under Biology implies a search of `code=a01b01*`.

If a user wants to find any information in the system pertaining a "Brazilian red spider" related to Taxonomy, the search can be expressed as the following LDAP filter:

```
(&(description="Brazilian red spider")(area="a01b01c08d06*"))
```

The only information requested to the user is the string about the spider. The code area is provided automatically by the search interface, since the user has navigated to that position in the virtual hierarchy.

4. Practical considerations

There is a series of questions to be considered before starting to work with a COPA-based virtual structure. The discussion of these questions is focused on LDAP, since this has been the main supporting infrastructure for COPA-enabled services so far.

4.1. Actual structure of the information

It is necessary to start from a set of data objects able to be structured in a (semi)hierarchical way: corporate structures, classifications and ontologies are good examples. The objective is to create a virtual structure, orthogonal to that defined for objects by their DNs in the LDAP servers. The virtual structure and the objects will be connected by incorporating COPA-based metadata into the appropriate object attributes.

4.2. Virtual structure(s) of the information

Is the base to allow virtual navigation and the classification of the objects. This structure should be accessible through LDAP, using a completely flat structure. The structure is provided by a COPA code that identifies the position of a structure element within the structure. The selection of the appropriate COPA format (see below) will be determined by the maximum number of entries that are expected to be necessary at each level of the structure.

4.3. Navigational support

For a certain navigation interface, the virtual view to be used must be defined. This definition determines the way in which data will be available and what will be the structure perceived by users accessing the system.
Virtual views can be defined using the class `copaSpecObject`, that incorporates the following mandatory attributes:

copaN Defines the format of the COPA code to be used. For example, `copaN=3` indicates that the code uses groups of 3 characters (1 letter and 2 numbers) for each level in the virtual structure, thus allowing for up to different elements as children of each non-leaf node.

copaBase Defines the location where the information about the virtual structure is.

copaCodeAttr Holds the name of the attribute where COPA codes will be stored. This is the attribute that the navigation interface will search to implement the virtual view.

copaPrintAttr Defines attributes which will be used to show information about the objects in the virtual view.

It is also possible to indicate the default virtual view to be used for a certain object or branch in the directory tree by means of the attribute `copaMainNav`. This attribute is part of the class `copaObject`, that can be used to model any COPA-aware object.

5. A short example

In this section we are going to show a simple example of an university information service, using two orthogonal virtual structures. The first one is based on a knowledge area ontology called CATRE (derived from a direct import of UNESCO codes into COPA). The other structure will offer a view of the university's organizational structure. We will concentrate on person entries: faculty, staff, students,...

5.1. Infraestructure

Actual structure

All entries are stored in a flat structure under **`dc=people,dc=univ,dc=es`**
The general format of a DN in this tree is:

```
uid=userIdentifier,dc=people,dc=univ,dc=es
```

Each entry has COPA-based metadata to show where is located in both virtual structures. For example, entries describing professors have the `teachingCode` attribute indicating the area(s) of teaching activities, and the `researchCode` attribute holding research areas. Affiliation to the organizational structure is modeled by the `deptCode` attribute.

Virtual structure

`dc=catre,dc=copa,dc=univ,dc=es` holds the knowledge area ontology, using entries of the type:

```
catreCode=a01,dc=catre,dc=copa,dc=univ,dc=es
```

`dc=org,dc=copa,dc=univ,dc=es` holds the organizational structure: schools, departments, groups, etc. by means of entries like:

```
orgCode=a01,dc=org,dc=copa,dc=univ,dc=es
```

It is even possible to relate the organizational structure to the above ontology: Just including a `catreCode` attribute in the objects describing the organizational structure would be enough.

Navigational support

A couple of navigation definition objects must be defined, one for each virtual structure:
`dc=catrenav,dc=copa,dc=univ,dc=es` will support the navigation based on CATRE. The entry will look like:

```
dn: dc=catrenav,dc=copa,dc=univ,dc=es
objectClass: copaSpecObject
copaN: 3
copaCodeAttr: catreCode
copaPrintAttr: catreName
copaBase: dc=catre,dc=copa,dc=univ,dc=es
```

`dc=orgnav,dc=copa,dc=univ,dc=es` will support the navigation based on the organizational structure of the university. The entry will look like:

```
dn: dc=orgnav,dc=copa,dc=univ,dc=es
objectClass: copaSpecObject
copaN: 3
copaCodeAttr: orgCode
copaPrintAttr: cn
copaBase: dc=org,dc=copa,dc=univ,dc=es
```

5.2. Services

A few examples of services that can be offered with this infrastructure are:

1. Two different navigation interfaces, one for each virtual structure. These interfaces will simply use the COPA metadata in the entries of the directory to build the views they offer. As mentioned above, both interfaces could be interconnected if a relation of the organizational structure and the ontology has been created.
2. An interface to information about (an possibly management of) virtual research groups. Each virtual group is composed of all the entries with a given `researchCode` .
3. By adding COPA codes into webpage metadata (if Dublin Core is in use, the element `dc.subject` can hold these codes) and harvesting them, a thematic search engine can be built, possibly connected to any of the other services above.