International Journal on Digital Libraries manuscript No.
(will be inserted by the editor)

Integrating ontologies and thesauri

for RDF schema creation and metadata querying.*

Bernd Amann[1,2], **Irini Fundulaki****[1,2], **Michel Scholl**[1,2]

[1] Cedric CNAM, 292 Rue St. Martin, 75141 Paris Cedex 03 France
[2] INRIA Rocquencourt, 78153 Le Chesnay Cedex, France
email : bernd.amann@inria.fr, irini.fundulaki@inria.fr, michel.scholl@inria.fr

Received: / Revised version:

Abstract. In this paper we present a new approach for building metadata schemas by integrating existing ontologies and structured vocabularies (thesauri). This integration is based on the specification of inclusion relationships between thesaurus terms and ontology concepts and results in application-specific metadata schemas incorporating the structural views of ontologies and the deep classification schemes provided by thesauri. We will also show how the result of this integration can be used for RDF schema creation and metadata querying. In our context, (metadata) queries exploit the inclusion semantics of term relationships, which introduces some recursion. We will present a fairly simple database-oriented solution for querying such metadata which avoids a (recursive) tree traversal and is based on a linear encoding of thesaurus hierarchies.

Key words: Ontologies, Thesauri, Domain Model, Metadata Querying, Mediation

1 Introduction

In open and evolving environments such as the World Wide Web, discovering, integrating and accessing information are difficult and complex tasks due to the semantic heterogeneities [30] resulting from the different terminologies and conceptualizations employed by the various information providers and consumers.

Providing access to heterogeneous and distributed databases through integrated views has been studied from the early 80's [6]. A large number of papers exist on the integration of distributed databases and [36,46,43] are comprehensive studies on the topic. However such approaches for data integration are not appropriate anymore for new applications based on the integration of a large number of Web resources that are not necessarily strongly structured or have a structure which is not fully available.

New approaches to this issue have been proposed in the past ten years. All of these are based on a three-tier architecture, where applications access *wrapped* information *sources* via *mediators*. In this paper, we focus on mediation models based on the creation and exchange of *semantic metadata* [32] describing the contents of shared Web resources in terms of a common domain specific vocabulary or *metadata schema*.

A metadata schema organizes information within a domain of interest and is defined by a community of people who want to provide tools for describing and querying resources within this domain. More precisely, a metadata schema is comprised of (1) a vocabulary, i.e. a set of element names to be used for the description of information in a domain (e.g. the *creator*, *title* elements of the Dublin Core metadata element set [18]), and (2) a set of semantic relationships for information structuring. We first present a modular approach for the creation of metadata schemas based on the integration of *existing ontologies* and *thesaurus hierarchies* defined according to the ISO 2788 standard for monolingual thesauri [31].

Each new source is added in the system by providing to the mediator its description. More precisely, a *source description* expresses the contents and the semantics of a source in terms of the metadata schema. For describing sources, a *knowledge-base* approach is often advocated. Information Manifold [3], PICSEL [26] are examples of such systems, based on Description Logics to represent the *metadata schema* and the *source descriptions*. In this paper, we propose a database approach with limited expressive power compared to that of the above knowledge-based systems but which is more efficient in the context of large size metadata schemas. We advocate that it is possible to efficiently implement the selection of sources according to their descriptions includ-

* This work was partially supported by the European project C-Web (IST-1999-13479)
** The work of this author was supported by a fellowship from the TMR European Network Chrorochronos.

ing the necessary reasoning mechanisms by using standard database technology.

1.1 Integrating ontologies and thesauri

Ontologies and *thesauri* can be considered as orthogonal ways for describing information. Ontologies are declarative specifications of the *concepts* and *roles* in a domain of discourse, and provide structural, sharable views of information. Thesauri are structured vocabularies (collections of terms), with rich semantics but restricted structural relationships. For example, although the *Art & Architecture Thesaurus (AAT)*[1], one of the largest thesauri in the field of western art terminology, includes extended taxonomies of cultural artifacts and styles, there is no explicit relationship denoting the fact that artifacts have a style. In our context, ontologies are perceived to have a dual role : provide a generic view of information and a structural interface over thesauri.

We follow a two-step approach to the construction of metadata schemas. In a first step, we specify for each ontology concept a set of thesaurus terms. The result is a *connection relation* between terms and concepts carrying *inclusion semantics*. In a second step, a *concept thesaurus* is extracted automatically for each concept. This thesaurus contains the terms connected to the concept in the connection relation, along with *hierarchical term relationships* derived from the initial thesaurus. The integration of these thesauri with the ontology produces a metadata schema consisting of (1) a *structural view* provided by the ontology, (2) *connection relations* between concepts and terms, and (3) *thesaurus hierarchies*.

The result of this integration is a *conceptual metadata schema* that can be used for several purposes.

1.2 Creating RDF schemas

The first application of our integration process is the creation of RDF [9] metadata schemas. The *Resource Description Framework* (RDF) is a metadata specification language that supports standard mechanisms for the representation of *metadata schemas* as well as *source specific metadata* (source descriptions).

Whereas RDF is very useful for the representation of metadata in the form of XML documents, it does not provide any methodology for the construction of metadata schemas, which is a difficult and time consuming task especially in environments that comprise a large number of information sources. Moreover, RDF offers no mechanism to decide whether a particular metadata schema meets the needs of an application or domain. Our integration model can be considered as a possible methodology for creating complex RDF schemas by using existing semantic components *(ontologies, thesauri)* that describe the organization of information within a domain of discourse.

[1] http://www.ahip.getty.edu/vocabulary/aat_intro.html

1.3 Source description and discovery

The second application of the resulting metadata schema discussed in this paper is (Web) *resource description* and *discovery*. In our context, a Web resource can be anything that is identified by a URL, i.e. a site containing a collection of documents with homogeneous or heterogeneous structure, a single document or a fragment of a document, an image. In this paper, we propose an efficient solution where a set of source descriptions is viewed as a *database* that can be queried for source addresses. In our context, efficiency is important because of the huge size of our metadata schemas (resulting from the large number of terms in the integrated thesauri) compared to traditional metadata schemas used in mediator based systems.

To illustrate our approach, we take examples from the cultural application domain. Thesaurus examples are taken from the *Art & Architecture Thesaurus* (AAT), one of the Getty Information Institute's[2] ongoing projects and known as one of the largest thesauri in the area of western art historical terminology. Ontology examples are inspired from the *ICOM/CIDOC Reference Model* [19] which is the result of one of the most significant efforts for a formal representation of the basic notions of the cultural application domain.

This paper is organized as follows. After having discussed related work (Section 2), we successively present in Section 3, the notions of ontology and thesaurus and our approach to the automatic construction of metadata schemas by integrating those semantic components. In the same section, we will also describe a straightforward translation of the resulting metadata schema into an RDF schema. Section 4 defines a resource description model for describing and querying Web resources based on our integrated metadata schema. An implementation of this description model with a standard object-oriented database system is presented in Section 5. Future work is discussed in Section 6.

2 Related Work

2.1 Search Engines

Within open and evolving environments such as the Web, *full text* and *keyword based search engines* implement an easy and rather efficient access to the underlying data sources. Search engines such as Altavista[3] and Yahoo[4] provide enhanced search capabilities through the classification of Web resources (URLs) into hierarchically organized information categories. Users can restrict the search space of a keyword query by choosing some specific domain of interest in form of a category from the hierarchy, but cannot represent more sophisticated (e.g. structured) queries.

[2] http://www.gii.getty.edu/

[3] http://www.altavista.com

[4] http://www.yahoo.com

2.2 Mediation systems

Mediation systems [50] provide more flexible and powerful tools for integrating sources. The idea is to establish appropriate *mappings* between the *source structure* and the *mediator's view*. These mappings are used to select sources and rewrite user queries into several source queries. Up to date, there exist two main approaches used in mediation based systems. In the first category, source integration is based on the *global as view* approach which relies on a flexible (semi-structured) data model to define global mediator views on heterogeneous sources. The data model and mapping language are able to represent and integrate data coming from heterogeneous structured and semi-structured sources. Representative systems in this category are TSIMMIS [11,24], YAT [13,16] and MIX [5]. The second is based on the *local as view* approach where each source is described independently as a local view on the *mediator domain model* which captures the basic vocabulary of a certain domain, expressed in some *database* or *knowledge-base* formalism (e.g. Description Logics [7]). Information Manifold [3], SIMS [12], PICSEL [26], Infomaster [25] and DISCO [49] are basic examples of such systems.

An important part of the research has been directed towards the use of *ontologies* as mediator domain models. One of the first systems to follow this approach was Carnot [17] that uses the CYC [35] knowledge base for describing source contents. The CYC ontology contains about 10^5 general concepts and 10^6 assertions on these concepts. An information source is integrated in Carnot by providing *mapping rules* between the source structures and CYC structures in the form of *articulation axioms*. User queries are formulated using the CYC structures, which are then translated into local structures through the established mapping rules.

Similar to Carnot, OBSERVER [37] and its later version *InfoQuilt*[5] use a knowledge-base approach based on the CLASSIC description logics [8] to provide access to heterogeneous sources. Each information source is described by a source-specific ontology and interoperation is achieved through inter-ontology relationships.

2.3 Annotation systems

Document annotation systems constitute a different way to information integration and querying on the Web. Data integration is obtained by *annotating* Web documents with semantic tags originating from *ontologies* that explicitly capture the semantics of the document contents.

SHOE [29] (Simple HTML Ontological Extensions) extends the HTML element set with new element types derived from application specific ontologies. In order to be able to query Web documents according to their annotations, the SHOE crawler gathers and stores the annotations of HTML pages in a knowledge base (Parka) that can be queried via first order conjunctive queries.

[5] http://lsdis.cs.uga.edu/proj/iq/iq.html

Ontobroker [21] consists of tools that enhance query access and inference services for Web documents. Similar to SHOE, HTML pages are annotated by element tags derived from application specific ontologies defined by using a logical object model with inference rules based on F-Logic. As in SHOE, annotated HTML pages are gathered, annotations are extracted and stored in the Ontobroker knowledge base that can be queried with a powerful logical query language.

OntoSeek [28] supports content-based access to the Web, designed for information retrieval from on-line yellow pages and product catalogs. OntoSeek combines an ontology driven content matching facility with a moderately expressive representation formalism. In OntoSeek the ontology incorporated is Sensus [33] based on WordNet [40] linguistic ontologies. Resource descriptions are arbitrary linguistic expressions (sentences) which are encoded as *linguistic conceptual graphs* using the previously mentioned ontologies. The bottleneck of this approach is that the semantics obtained by a linguistic analysis of a description might not correspond to the initial semantics defined by its author. Moreover, Sensus contents do not often correspond to real world relationships between classes of entities in the world, making difficult the precise encoding of information.

Quest [4] was designed and implemented for querying and manipulating documents written in the OHTML [34] markup language. OHTML supports fine granularity semantic tagging of HTML pages. In contrast to the previous systems that require extensions of the HTML tags, in Quest annotations are encoded as HTML comments with no incidence on the actual structure of the HTML page. OHTML annotations are viewed as OEM [42] objects, expressed in textual form within an HTML page. Quest uses the W3Lorel query language, based on the Lorel [2] language to query the OEM objects (semantic view), as well as the hypertext view (HTML tags) of the document. Following this semi-structured approach, Quest allows for arbitrary tagging of HTML pages, offering a flexibility to the user on the choice of semantic tags. On the other hand, it introduces a certain degree of *semantic ambiguity* allowing users within the same community to annotate, using different terminologies, closely related document contents.

2.4 Metadata vocabularies

Over the past years a great amount of effort has been invested in the development of metadata vocabularies for the exchange of information across different applications and domains. Dublin Core[6] contributes to semantic interoperability by promoting a common set of elements which can be used to describe in a consistent manner information concerning the contents of electronic documents, such as their *title*, *creator*, or *subject*. USMARC[7] defines a set of descriptive elements for the representation and exchange of bibliographic data. In the cultural domain, the Aquarelle Project [39] uses the XML

[6] http://purl.oclc.org/dc/

[7] http://lcweb.loc.gov/marc/marc.html

CI DTD (Document Type Definition) of the French Ministry of Culture [8] to describe a set of element names, dedicated to territory inventory making. All the above metadata element sets are the result of the collaboration of a number of user communities and other authorities in the corresponding fields.

2.5 Contribution

Our contribution is twofold. First the proposed integration of ontologies and thesauri can be considered as a methodology to construct annotation and mediation schemas by (re-)using existing semantic components in a domain of interest. This leads to efficient and scalable mediator designs for integrating and accessing sources within a domain of interest. The second contribution is a fairly simple database-oriented solution for querying metadata, which avoids recursive tree traversals on thesauri based on a linear encoding of thesaurus hierarchies. Whereas the expressive power of source descriptions is limited compared to more powerful knowledge-based mediator systems, metadata query evaluation can be implemented in a very efficient way.

3 Integrating Ontologies and Thesauri

3.1 Ontologies

The term *ontology* has been used in several disciplines, from philosophy, to knowledge engineering, where an ontology is comprised of *concepts*, *concept properties*, *relationships* between concepts and *constraints*. Ontologies are defined independently from the actual data [27], reflect a common understanding of the semantics of the domain of discourse and are used to share and exchange information between sources. They are declarative specifications of the basic notions in a domain. In the fields of information systems and database systems, an ontology would be represented by a conceptual schema. In our context, we consider ontologies with *inheritance* relations (isa) and *typed roles* between *concepts*, sufficient to model a large class of applications that can be easily represented as RDF schemas (Section 3.4).

Definition 1. An ontology is a triple $\mathcal{O} = (C,\ R,\ isa)$ defined as follows :

1. $C = \{c_1, c_2, \ldots, c_n\}$ is a set of concepts, where each concept c_i refers to a set of objects *(concept instances)*,
2. $R = \{r_1, r_2, \ldots, r_m\}$ is a set of binary typed roles between concepts,
3. isa is a set of inheritance relationships defined between concepts. Inheritance relationships define a partial order over concepts and carry *subset* semantics.

Ontologies can be represented as labeled directed graphs where nodes correspond to concepts and arcs correspond to roles and isa relationships. Figure 1 illustrates an example ontology, inspired from the ICOM/CIDOC Reference Model [19] which is used to describe cultural information. Concept Physical Object collects all physical objects, the latter *composed of* other physical objects. Activities (concept Activity) are *associated with* physical objects, the former *performed by* persons, institutions and organizations (concept Actor). Concepts Biological Object and Man-Made Object are *sub-concepts* of Physical Object and inherit all roles defined in their superconcept. Instances of Man-Made Object have a title (role *has-title*) instance of concept Title and have been created in a specific period (role *of-period*, instance of concept Period). Iconographic Object is a sub-concept of Man-Made Object. Iconographic objects have a style (role *style*), instance of concept Style.

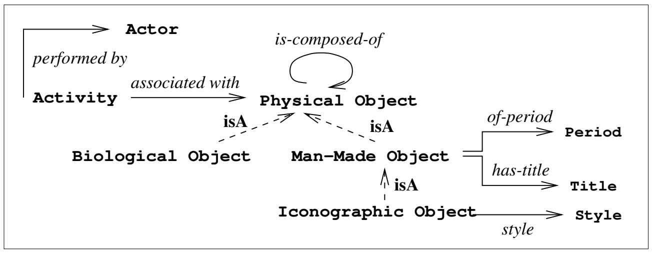

Fig. 1. A simple cultural ontology.

3.2 Thesauri : Structured Vocabularies

A vocabulary is a collection of *terms* that describe information in a domain of interest. Examples of such vocabularies are the *ACM Computing Classification System*[9], the Library of Congress Subject Headings[10], the *Unified Medical Language System*[11] and the *Art & Architecture Thesaurus*[12] in the cultural domain. Thesauri are structured vocabularies of thousands of *terms* which are used as efficient means for *consistent indexing* and *retrieval* of information.

Thesaurus terms are considered as the representation of concepts in the form of a noun or a noun phrase. Concepts are perceived by thesaurus developers as referring collectively to a set of objects (*concept instances*) [38] that are considered as such not with respect to a formal classification process but through a common agreement. Under this perspective, the interpretation of a thesaurus term is a set of objects, which we call the *extension* of the term. Thesauri are said to be structured since they include a fixed set of semantic term relationships. Due to the set theoretic definition of terms, these semantic relationships are interpreted as relations between sets [20,47].

[8] http://aquarelle.inria.fr/Inventaire

[9] http://www.iicm.edu:8080/jucs_classification

[10] http://www.grci.com/services/library/libcongress/index.shtml

[11] http://gmedserv.nlm.nih.gov/research/umls

[12] http://www.ahip.getty.edu/vocabulary/aat_intro.html

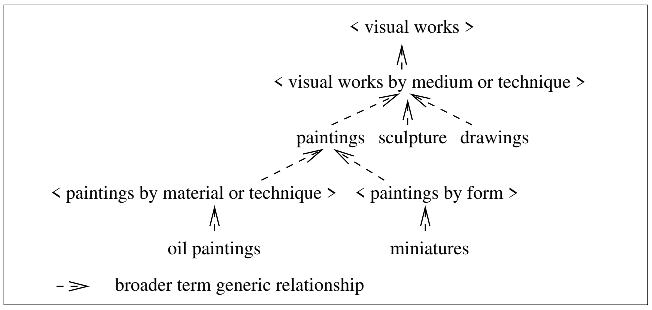

Fig. 2. Part of the *Art & Architecture Thesaurus* hierarchy *Visual Works* which collects all artifacts that are used for visual communication (paintings, sculptures, photos).

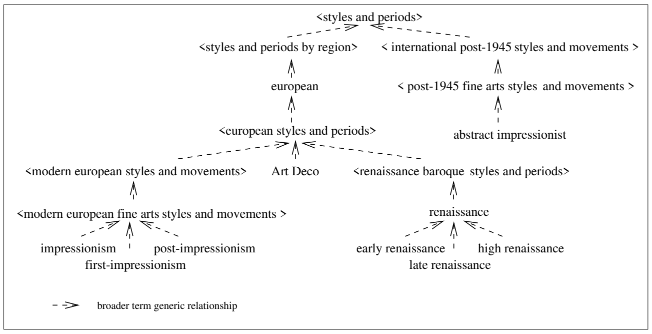

Fig. 3. Part of the *Art & Architecture Thesaurus* hierarchy *Styles & Periods* which collects all styles, periods and movements of Art in the western world.

The ISO 2788 Standard [31] for the documentation and establishment of monolingual thesauri defines the following four kinds of term relationships which distinguish structured thesauri from arbitrary collections of terms :

1. *generalization (broader term generic - btg)*,
2. *instance (broader term - bt)*,
3. *partitive* or *part-of (broader term partitive - btp)*,
4. *associative (related term - rt)* and
5. *equivalence (used for term - uf)*.

Term relationships btg and btp are called *hierarchical*. In this paper we are only concerned with btg relationships[13] which carry subset semantics and are the most frequently used hierarchical relationships. Nevertheless, our approach can easily be extended with equivalence relationships : equivalent terms generally play the role of synonyms. In our approach a term (or descriptor) can be considered as a canonical name for the equivalence class defined by the equivalence relationship.

btg-relationships are *transitive* and organize terms with similar semantics into *directed acyclic graphs* (DAG), referred to as *hierarchies*, or *classification schemes*. Two examples of btg-hierarchies are shown in Figures 2 and 3. For example in Figure 2, term *paintings* is broader than *oil paintings*, with the interpretation that oil-paintings are paintings. A thesaurus hierarchy is defined by its *root term*, a term with no broader term (e.g. *<visual works>* in Figure 2). We assume *mono-hierarchical* thesauri which can be represented by a forest of hierarchies where each term has exactly one broader term.

Although the definition we give is not complete w.r.t. all possible term relationships existing in real thesauri it is sufficient for creating rich metadata schemata.

Definition 2. A *thesaurus* is a couple $\mathcal{T} = (D, btg)$ such that

1. $D = \{t_1, t_2, \dots t_n\}$ is a set of terms,
2. btg defines a partial ordering in D where each term has at most one predecessor.

[13] The interested reader can refer to ISO 2788 [31] for a deeper presentation of the remaining term relationships.

3.3 Integrating Ontologies and Thesauri

In this section, we present a methodology for the construction of metadata schemas based on the integration of existing ontologies and thesaurus hierarchies. The construction is done in two steps. In the first step, we specify for each ontology concept c, a set of terms, considered as *sub-concepts* of c. This step is similar to establishing *inter-schema assertions* [10] for database schema integration and cannot be a completely automated process since it requires the knowledge of the thesaurus and the ontology semantics. In the second step, a *concept thesaurus* is extracted *automatically* for each concept. This thesaurus contains the terms connected to the concept in the first step, along with *broader-generic* (btg) relationships derived from the initial thesaurus, which can be done automatically. The integration of each extracted concept thesaurus with the ontology produces a *metadata schema* consisting of a *structural view* (provided by the ontology) and a *semantic view* (provided by thesaurus hierarchies).

Step 1 : Specialization of Concepts with Terms

In the first step of the integration process, thesaurus terms are "connected" to ontology concepts. These connections belong to a binary *connection relation* $Con \subseteq D \times C$ over a set of thesaurus terms D and a set of ontology concepts C. An example of a connection relation is presented in Figure 4. Terms *impressionism*, *post-impressionism* and *abstract impressionism* of the *Art & Architecture Thesaurus* hierarchy *Styles & Periods* (Figure 3) describe specific styles (ontology concept Style in Figure 1). Term *first-impressionism* of the same hierarchy describes both a style and a period (concepts Style and Period respectively). Similarly, term *renaissance* and its narrower terms of hierarchy *Styles & Periods* describe different types of styles and periods (ontology concepts Style and Period respectively). Finally, terms *paintings*, *oil paintings* and *sculpture* of the *AAT* hierarchy *Visual Works* (Figure 2) describe different kinds of iconographic objects (ontology concept Iconographic Object).

The way the user actually chooses the concepts to be connected to a given term will not be discussed in this paper. Briefly speaking, either one connects t with some concept c

Term	Concept	*Term*	Concept
impressionism	Style	*paintings*	Iconographic Object
post-impressionism	Style	*oil paintings*	Iconographic Object
abstract impressionism	Style	*sculpture*	Iconographic Object
renaissance	Style	*early renaissance*	Style
renaissance	Period	*early renaissance*	Period
late renaissance	Style	*high renaissance*	Style
late renaissance	Period	*high renaissance*	Period
first-impressionism	Period	*first-impressionism*	Style

Fig. 4. A connection relation Con for *AAT* hierarchies *Styles & Periods*, *Visual Works* and ontology concepts Style, Period and Iconographic Object.

with the assumption that all descendants of t are connected to c or one chooses explicitly among the descendants of t.

In the previous example, we do not connect the whole thesaurus hierarchy *Styles & Periods* to concepts Style and Period. We adopt this *selective approach*, i.e. relating thesaurus terms to ontology concepts explicitly, for several reasons. An obvious reason is that some terms could be out of the scope of the application that has to be described by the resulting metadata schema. For example, if some application is only concerned with paintings, then terms referring to artifacts other than paintings (e.g. *sculpture*, *drawings*) need not be considered in the resulting schema. Another reason is that some terms (e.g. *guide terms* in [31,48]) are used to organize thesaurus hierarchies (e.g. <*visual works by medium or technique*>) and might have no use for describing information. Finally, another important reason is that thesaurus hierarchies might contain terms which can be connected to different concepts. For example, terms of the *AAT* hierarchy *Styles & Periods* (Figure 3) describe styles (e.g. *impressionism*), periods (e.g. *art deco*), or both styles and periods (e.g. *renaissance*). Connecting terms to concepts in a selective manner allows users to clarify between the multiple semantics of a term (e.g. as in the case of *homonyms*) and consequently resolve semantic ambiguities at the thesaurus level.

Observe that a term can be connected to several concepts, For example, in the connection relation illustrated in Figure 4, term *first-impressionism* is connected to both Style and Period concepts.

Step 2 : Thesaurus Extraction

After having defined the connection relation between terms and concepts, we extract for each concept in the connection relation, a thesaurus, called *concept thesaurus*.

Let $\mathcal{T}$ be a thesaurus connected to a subset of concepts $S \subseteq C$ by a connection relation Con_S. Then, $extract(S, \mathcal{T})$ is a new thesaurus that contains (1) the subset of all terms in $\mathcal{T}$ connected to concepts in S and (2) all btg relations between these terms, induced by the btg relations in the initial thesaurus. More precisely :

Definition 3. Denote by $Con_S(t, c)$ the relational table where each tuple represents the connection between a term t in thesaurus $\mathcal{T} = (D, btg)$ and a concept c in S. Then $extract(S, \mathcal{T}) = (D', btg')$ is a new thesaurus where

- $D' = \pi_t(Con_S)$ is the subset of terms in the connection relation.
- btg' is induced by the partial order defined by btg on terms in D' : for all pairs of terms t and t' in D', if $btg(t, t')$, i.e. t' is a broader term of t in the initial thesaurus $\mathcal{T}$, then $btg'(t, t')$, i.e. t' is a broader term of t in the new thesaurus $extract(S, \mathcal{T})$.

It is possible to define a *concept thesaurus* $\mathcal{T}_c$ for each concept c in the set of ontology concepts S as follows :

Definition 4. Let $\mathcal{T}$ be a thesaurus, S_c be the set of sub-concepts of c including c and Con_{S_c} be a connection relation defined for $\mathcal{T}$ and S_c. Then the *concept thesaurus* associated with c is defined as $\mathcal{T}_c = extract(S_c, \mathcal{T})$.

For the definition of a concept thesaurus we exploit not only the btg relations between the terms but also the isa relationships at the ontology level. Consider the example in Figure 5. Terms t, v and w are connected to concepts c, d and e respectively. The extraction operation on concept c will construct the thesaurus $\mathcal{T}_c$ that contains besides term t, terms v and w that are connected to its sub-concepts. Observe also that a term can appear in multiple concept thesauri, and terms that are not connected to any concept such as, for example, term u in Figure 5, have disappeared from the concept thesauri. Moreover, the selection operation on concept c created a btg relation between terms w and t which were not directly related in the original thesaurus.

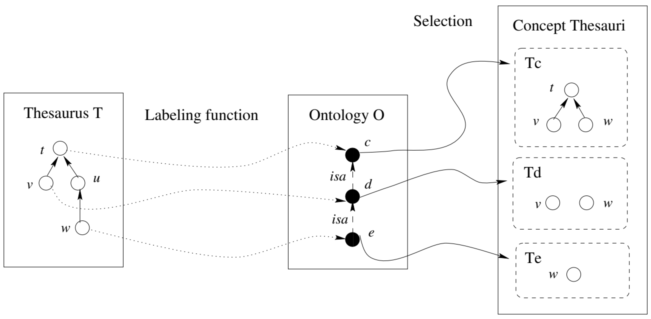

Fig. 5. Extracted Thesaurus Examples.

At this point, we should mention that a concept thesaurus can be induced by those of its super-concepts. For example, if d is a sub-concept of c, and $\mathcal{T}_c$ is the concept thesaurus of c, then the concept thesaurus of d can be extracted from thesaurus T_c as follows : $\mathcal{T}_d = extract(S_d, \mathcal{T}_c)$. In the previous example, only concept thesaurus $\mathcal{T}_c$ of concept c has to be extracted from the original thesaurus $\mathcal{T}$. All thesauri corresponding to sub-concepts of c might then be created on demand during the creation of the metadata schema (Section 3.4).

3.4 RDF schema construction

The first application of the resulting metadata schema is straightforward and consists in the creation of an *RDF schema*. Ontology concepts and thesaurus terms in the metadata schema are modeled as RDF classes and ontology roles correspond to RDF properties. Ontology isa relationships, *connection relations* between terms and concepts and btg relations between terms all carry *inclusion* semantics and are modeled with the RDF subclassOf property.

Resource Description Framework

The *Resource Description Framework* (RDF) is a *foundation for processing metadata* [9][14] which supports standard mechanisms for the representation of metadata schemas as well as source specific metadata. It relies on a simple, graph-based data model and uses XML (eXtensible Markup Language)[15] to communicate and process metadata in a machine readable and human understandable format. Similar to the separation of schema and instance in traditional databases, we can distinguish between RDF *schemas* and RDF *descriptions* (instances of an RDF schema).

RDF descriptions

RDF can be used to describe any kind of *resource*, identified by a URI (Uniform Resource Identifier), such as a Web server, an XML document or an element of an HTML page (e.g. an image). RDF supports the definition of resource *properties* whose values can be other resources or literals (strings, integers). A collection of *property/value* pairs that refers to a specific resource is called an RDF *description* and can be represented as a *labeled directed graph* where nodes correspond to resources or literals (values) and edges to resource properties.

Figure 6 shows an RDF description for a Web page that describes a painting of the French painter Claude Monet. The XML *namespace* mechanism allows the specification of different RDF schemas. For example, lines 2 and 3 define two XML namespaces : xlmns:web-page and xlmns:artifact. The first one contains general properties of HTML pages (title, presents, creator) and the second one specifies properties of cultural artifacts (title, style, type, period). This mechanism is very useful since it permits the reuse of existing, distinct RDF schemas within the same RDF description, without creating name conflicts (e.g. web-page:title, artifact:title). Line 4 tells us that the description that follows concerns the HTML page which can be accessed by the URL http://metalab.unc.edu/louvre/paint/monet/first-/impression/. The title of this page is *"Web Museum: Monet, Claude : Impression : soleil levant "* (line 5) and has been created by Nicolas Pioch (line 14). To describe properties of the painting, it is necessary to define a local resource which is identified by URI soleil_levant that refers to the painting. The painting's properties are its type (oil painting, line 8), title (Impression : soleil levant, line 9), style (impressionism, line 10) and period (first-impressionism, line 11).

RDF schemas

The *RDF Schema Specification Language* [9] is a declarative language used for the definition of RDF schemas[16] incorporating aspects from knowledge representation models (e.g. semantic nets), database schema definition languages and graph models. It is a simple language of restricted expressive power compared to predicate calculus based metadata languages such as CycL [35] and KIF[17].

An RDF schema defines *classes* and *properties* which can be instantiated in RDF descriptions. Classes are organised into hierarchies using the property rdfs:subclassOf which is defined in *RDF Schema* (namespace rdfs) which has the standard semantics of class inheritance in object-oriented data models. For example, the RDF schema illustrated in Figure 7 defines class Man Made Object (line 4) and its subclass Iconographic Object (lines 5,6). It also defines classes Style (line 7), Period (line 8). RDF Schema allows both typed and untyped properties. Properties in our example are typed (i.e. they have a restricted domain and range). In Figure 7, property period (line 15) is defined between classes Man Made Object (line 16) and Period (line 17), using the RDF Schema properties rdfs:domain and rdfs:range respectively. Summarizing, RDF offers a rich, comparatively simple graph-based data model and supports the definition of source specific metadata (RDF descriptions) and metadata schemata (RDF schemas). It uses XML for the syntactical representation, exchange, and processing of these metadata.

RDF metadata schema construction

The creation of the RDF schema $\mathcal{S}$ for an ontology $\mathcal{O} = (C, R, isa)$, and a set of concept thesauri $\mathcal{T}_c = (D, btg)$ is straightforward :

1. RDF classes : for each ontology concept c and for each term t in $\mathcal{T}_c$, define RDF classes c and c:t, respectively.
2. RDF properties : for each typed role $r(c, d)$ in R define an RDF property with rdfs:domain RDF class c and rdfs:range RDF class d.

[14] http://www.w3c.org/RDF

[15] http://www.w3.org/XML/

[16] In the following *RDF Schema* will denote the specification language used to define RDF schemas.

[17] http://logic.stanford.edu/kif/kif.html

```
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:web-page ="http://metalab.unc.edu/louvre/namespaces/web-pages"
  xmlns:artifact ="http://metalab.unc.edu/louvre/namespaces/artifacts">
 <rdf:Description
   about="http://metalab.unc.edu/louvre/paint/monet/first/impression">
   <web-page:title>Web Museum: Monet, Claude: Impression: soleil levant
   </web-page:title>
   <web-page:presents>
     <rdf:Description about="soleil_levant">
       <artifact:type>oil painting</artifact:type>
       <artifact:title>Impression : soleil levant</artifact:title>
       <artifact:style>impressionism</artifact:style>
       <artifact:period>first-impressionism</artifact:period>
     </rdf:Description>
   </web-page:presents>
   <web-page:creator>Nicolas Pioch</web-page:creator>
  </rdf:Description>
</rdf:RDF>
```

Fig. 6. An RDF description for resource http://metalab.unc.edu/louvre/paint/monet/first/impression.

```
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
   xmlns:rdfs="http://www.w3.org/TR/1999/PR-rdf-schema-19990303#"
   xmlns:artifact="">
  <rdfs:Class rdf:ID="Man Made Object"></rdfs:Class>
  <rdfs:Class rdf:ID="Iconographic Object">
    <rdfs:subclassOf rdf:resource="#Man Made Object"/></rdfs:Class>
  <rdfs:Class rdf:ID="Style"></rdfs:Class>
  <rdfs:Class rdf:ID="Period"></rdfs:Class>
  <rdf:Property rdf:ID="style">
    <rdfs:domain rdf:resource="#Iconographic Object"/>
    <rdfs:range rdf:resource="#Style"/> </rdf:Property>
  <rdf:Property ID="title">
    <rdfs:domain rdf:resource="#Man Made Object"/>
    <rdfs:range  rdf:resource="#rdfs:Literal"/></rdf:Property>
  <rdf:Property rdf:ID="period">
    <rdfs:domain rdf:resource="#Man Made Object"/>
    <rdfs:range rdf:resource="#Period"/></rdf:Property>
</rdf:RDF>
```

Fig. 7. An RDF schema for describing cultural resources.

3. RDF subclassOf properties : define an rdfs:subclassOf property between the corresponding RDF classes for each isa (and btg) relation between ontology concepts (and thesaurus terms). In addition, for each RDF class c:t, that corresponds to the *root term* of concept thesaurus $\mathcal{T}_c$, add an rdfs:subclassOf property between classes c:t and c.

It is interesting to note that we connect only the *root* term of each concept thesaurus to the corresponding concept. Due to the transitivity of the rdfs:subclassOf property, it can be induced that a term t is a subclassOf another term t or a concept c.

The RDF schema illustrated in Figure 8 has been constructed from the ontology in Figure 1, the thesaurus classification schemes in Figures 2 and 3, and the connection relation in Figure 4.

Ontology concepts Man Made Object, Iconographic Object, Style, Period and terms *oil paintings*, *paintings*, *impressionism* and *first-impressionism* are all represented as RDF classes (lines 5,7,9,11,21,23,25,27). RDF class paintings [18] is defined as a subclass of class Iconographic Object (line 22), since term *paintings* is the root term of the Iconographic Object concept thesaurus. In the same way, classes impressionism and first-impressionism are defined as subclasses of classes Style and Period respectively (lines 26,28). Class oil paintings is a subclass of class paintings (line 24) (defined by the btg-relations between term *oil paintings* and term *paintings*). Ontology role *style*, is defined as an RDF property, its domain being the class Iconographic Object (line 19) and its range class Style (line 20). By definition of the *subclassOf* property, all subclasses of Iconographic Object inherit this property.

[18] For readability purposes, terms are only prefixed with the corresponding concept if they are shared by different concept thesauri.

Using this RDF schema, one can provide RDF descriptions about specific Web resources. For example, a new RDF description for the source described in Figure 6 is shown in Figure 9. When comparing this new description with the previous one, one can observe that we have replaced namespace artifact by a new namespace int which corresponds to the RDF schema in Figure 8. In this RDF description, semantic information that was captured as a value in the previous description has been added at the schema level. For example, the fact that the resource described an impressionist painting was encoded in the value of tag <artifact:style>.

```
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:rdfs="http://www.w3.org/TR/1999/PR-rdf-schema-19990303#"
         xmlns:int="">
  <rdfs:Class rdf:ID="Physical Object"></rdfs:Class>
  <rdfs:Class rdf:ID="Man-Made Object">
    <rdfs:subclassOf rdf:resource="#Physical Object"/></rdfs:Class>
  <rdfs:Class rdf:ID="Iconographic Object">
    <rdfs:subclassOf rdf:resource="#Man-Made Object"/></rdfs:Class>
  <rdfs:Class rdf:ID="Period"></rdfs:Class>
  <rdfs:Class rdf:ID="Title"></rdfs:Class>
  <rdfs:Class rdf:ID="Style"></rdfs:Class>
  <rdf:Property rdf:ID="of-period">
    <rdfs:domain rdf:resource="#Man Made Object"/>
    <rdfs:range  rdf:resource="#Period"/></rdf:Property>
  <rdf:Property rdf:ID="title">
    <rdfs:domain rdf:resource="#Man Made Object"/>
    <rdfs:range  rdf:resource="#Title"/></rdf:Property>
  <rdf:Property rdf:ID="style">
    <rdfs:domain rdf:resource="#Iconographic Object"/>
    <rdfs:range  rdf:resource="#Style"/></rdf:Property>
  <rdfs:Class rdf:ID="paintings">
    <rdfs:subclassOf rdf:resource="#Iconographic Object"/></rdfs:Class>
  <rdfs:Class rdf:ID="oil paintings">
    <rdfs:subclassOf rdf:resource="#paintings"/></rdfs:Class>
  <rdfs:Class rdf:ID="impressionism">
    <rdfs:subclassOf rdf:resource="#Style"/></rdfs:Class>
  <rdfs:Class rdf:ID="first-impressionism">
    <rdfs:subclassOf rdf:resource="#Period"/></rdfs:Class>
</rdf:RDF>
```

Fig. 8. The RDF schema resulting from the integration of the ontology and thesaurus.

```
1. <rdf:RDF
2.  xmlns:web-page ="http://metalab.unc.edu/louvre/namespaces/web-pages"
3.  xmlns:int ="http://www.connectit.com/icom/aat">
4.  <rdf:Description
5.   about="http://metalab.unc.edu/louvre/paint/monet/first/highway/">
6.   <web-page:title>Web Museum: Monet, Claude :Impression :
                                    soleil levant</web-page:title>
7.    <web-page:presents>
8.     <int:oil paintings
9.      about="http://metalab.unc.edu/louvre/paintings/monet/impression">
10.     <int:title>Impression : soleil levant</int:title>
11.      <int:style><int:impressionism/></int:style>
12.      <int:of-period><int:first-impressionism/>
13       </int:of-period>
14.    </int:oil paintings>
15.   </web-page:presents>
16.   <web-page:creator>Nicolas</web-page:creator>
17.  </rdf:Description>
18.</rdf:RDF>
```

Fig. 9. RDF description for Claude Monet painting using the integrated schema.

This value corresponds in fact to a term in the AAT and is represented as an instance of class int:impressionism (line 11) in the new schema. The same argument holds for the value *first-impressionism* which is now represented as an instance of RDF class int:first-impressionism (line 12). Observe also (tag <rdf:Description>) (Figure 6, line 7), has been replaced by a typed node tag <int:oil paintings> (line 8) indicating that the described resource is an oil painting.

4 Resource Description Model

A *metadata schema* resulting from the above integration process describes a *domain of interest* and is defined by a community of people that want to provide tools for describing and querying resources in this domain. In particular, those tools allow the sharing of a *dynamic* set of Web resources. A source that is integrated in the system, provides a *description* of its structure, contents or even of some semantics which is not explicit in the data nor in its structure. This description is made in terms of the metadata schema. Once a source has provided its description, it becomes *visible* to the system and it is said to be *published*. At a given time instant, the pair *(schema, set of source descriptions)* is called the *description base* (DB).

The description base (DB) is managed by a *mediator* which provides to the user a uniform view of the published sources and a user must express his/her query to the mediator in terms of the metadata schema.

Transparently to the user, the mediator selects a subset of sources as an answer to a user query. If a selected source s contains a collection of documents, the mediator could forward to s a subquery based on the user query in order to filter the documents (fragments) of interest in s. The mediator can then construct query plans that are sent to the selected sources for evaluation, and gathers the returned results. In the general mediation scenario we have described, we could identify two phases: the first, is the one of identification of the relevant sources to a user query, known as *resource discovery* phase; the second is the construction of query plans, their evaluation by the sources and the fusion of the returned results. In this paper, we do not address this second step. We restrict our attention to the first issue which is the identification of the sources relevant to a user query.

Our approach views the DB as a *database* queried for *source addresses*. A source address, called in the sequel a *url* can be the address of a site containing a collection of documents with homogeneous or heterogeneous structures, the address of a single document or that of a document fragment. Once the user has obtained a set of urls, he/she is able to access each of them. In the following, we first define the *resource description model*. Next, we specify the semantics of a DB for a given schema and a given set of described sources. An implementation of this model is discussed in Section 5. Examples of queries are given to illustrate our approach to resource discovery.

4.1 Description of sources

A source is described in terms of one or several concepts. For each concept c, we define a set of typed properties $P(c)$ as follows:

1. If c is connected to some concept c' through role r, we shall say that r is a *(role) property* of type c'.
2. Each concept c is provided with a specific property of name $term$ and of type T_c, where T_c is the concept thesaurus of c. Note that here a *thesaurus term* t is considered as a value of property *term* while in the RDF schema presented in Section 3.4 a term t that belongs to the concept thesaurus of a concept c was viewed as a sub-concept of c.
3. Let A be a set of atomic values. W.l.g., we assume that A is the set of strings [19]. Then, it is possible to create for a concept c a new property p of type A. Property p is referred to as a *value* property of c. The enrichment of the metadata schema with value properties allows to refine source descriptions, but the price to be paid is an increase in the size of the DB.
4. There is no other property in $P(c)$.

To describe some source, one chooses to specify properties of one or several concepts. More precisely, a *(source) description* is a tuple (u, c, d) where u is the URL of the source to be described and d is a concept descriptor for concept c defined as follows. Let $P(c)$ be the set of properties of some concept c. A *concept descriptor* for concept c is a tuple $d = (p_1 = d_1, ..., p_n = d_n)$, $p_i \in P(c), 0 \leq n \leq |P(c)|$ where

- $p_i \in P(c)$ and $p_i \neq p_j$ for all $i \neq j$, i.e. each property can be used at most once in a descriptor,
- if p_i is the $term$ property then d_i is a term of the thesaurus T_c,
- if p_i is a value property r of type A, then d_i is an element of A.
- if p_i is a role property r of type c', then $d_i = (c', d')$ where d' is a concept descriptor for concept c'.

The following examples of source descriptions rely on the ontology of Figure 1.

1. (url1, Actor)
2. (url2, Actor, name = “Monet”)
3. (url3, Actor, term = “painter”, name = “Monet”)
4. (url4, Activity, performed_by = Actor)
5. (url5, Activity, performed_by = “painter”).
6. (url5, Physical_Object, of_period = “impressionism”,term = “paintings”).

The first description defines no properties on concept Actor (the description is the empty tuple []) and source *url1* is about any kind of actors. The second source *url2* is about actors whose name is Monet (*name* is a value property of concept Actor of type String). The third source *url3* is about painters (term = “painter”) with name “Monet” (*painter* is a term of the thesaurus connected to concept Actor). The fourth description says that source *url4* is about activities performed by any kind of actors (concept Activity associated to concept Actor through role *performed_by*). Finally source *url5* is described by the last two descriptions stating that this source is about activities of painters and about paintings (term=”paintings”, concept Physical Object) of the impressionist period.

The two last descriptions are shortcuts for (url5, Activity, performed_by = (Actor, term = “painter”)) and (url5, Physical_Object, term = “paintings”, of_period = (Period, term = “impressionism”)). When there is no ambiguity (single target concept for a role), the target concept may be omitted. Similarly, by convention if no value is specified for the *term* property of a concept c, then as value is considered to be the root term of c’s concept thesaurus.

4.2 Object-Oriented Description Base

A DB is a set of source descriptions, as defined in the previous section. It corresponds to a set of sources published at a given time instant, each of the sources is classified by one or more

[19] We might also add other atomic types such as Integer or Date.

concepts of the metadata schema. We use a classical object-oriented database approach to represent and interpret descriptions : a class is defined for each concept and a description is an object of this class. The *extension* of class c, i.e. the set of objects of class c stored in the DB represents the descriptions classified by c. The DB is the union of the extensions of all classes. The only peculiarity of this object-oriented representation comes from the semantics associated with the term hierarchies in concept thesauri : if o corresponds to the source description $(u, c, d = [term = t, p_2 = d_2, ..., p_n = d_n])$ and belongs to the extension of class c, then for all broader terms t' in the thesaurus connected to c, object o' with description $(u, c, d' = [term = t', p_2 = d_2, ..., p_n = d_n])$ also belongs to the extension of c. In other words, if (u, c, d) is a source description in the DB, then (u, c, d') also belongs to the DB. For example, (i) term $paintings$ and $oil\_paintings$ are terms in the concept thesaurus of Iconographic Object, (ii) term $paintings$ is a broader term of $oil\_paintings$ and (iii), if a source u in the description base is about $oil\_paintings$, then the description base describes also this source as being about $paintings$.

Observe that:

- if no URL is described by concept c, the extension of c is empty (c has no description instance),
- if (u, c, d) is in the extension of c and c' is a *super-concept* of c, then (u, c', d) is in the extension of c', i.e. all resources that are described by a concept description (u, c, d) are also described by a concept descriptor (u, c', d), where c is a *sub-concept* of c' (inclusion semantics of isa).

Note also that except for the (thesaurus) *term* attribute defined on a tree-structured domain, our description model can be expressed by any object-oriented database model (see for example the model of [1]), i.e. can be supported by any object-oriented system. In the following section, we shall show that by an appropriate coding of thesaurus terms, we do not need anymore an explicit tree structure (the tree structure being hidden in the value of the term), which renders the description model fully compatible with an object-oriented database representation.

5 Implementation

In this section we describe a prototype implementation of the DB with the object-oriented database system (ODBMS) O_2 [22] and its querying with the OQL query language [15]. Although perhaps too complex as an end user language, OQL supports rather complex queries and allows a powerful mediation with information sources. We look at two possible implementations of the DB. The first subsection describes an implementation based on a representation of thesauri in form of trees. However, the performance of queries involving an extensive use of thesauri traversals might suffer of the somehow naive thesaurus internal representation. This is why we suggest in the second subsection a linearized representation of thesauri which should lead to significant performance gains.

5.1 A Naive Object Oriented Implementation of the DB

Metadata schema concepts are implemented as O_2 classes. A source description (u, c, d) is an object of class c. Its value is a *tuple* whose attributes are (i) a mandatory attribute with name url storing the url of the source described by the object, (ii) a mandatory attribute term that corresponds to *term* property and has for a value a thesaurus term and (iii) as many attributes as the properties of the corresponding concept. A role property r is implemented as an attribute r referring to an object of class c', where *c'* corresponds to the target concept of the role. Value properties p, i.e. properties for which the target is an atomic type A are not referring to an object but have a value of type A, where A is an atomic type of O_2 [20]. Terms are represented as objects (instances) of class Term. The value of each object o, instance of class Term, is a tuple of type $[t, f]$ where t is the name of the term (of type String) and f refers to its broader term. For each class c we define a persistent root cs (database entry point) of type set(c) which contains all objects of class c (all descriptions of concept c). Similarly, for each class c we define a persistent root tc of type set(Term) which contains all terms in the concept thesaurus of c. For evaluating queries requiring thesaurus traversals, we define in each class c the method $tree(t : string) : set(Term)$ which returns the set of narrower terms of t in the concept thesaurus of c. As mentioned previously, the query language used to query the DB is OQL. The user specifies a path in the ontology (by specifying role properties) as well as term and/or value attributes. A query always gives as a result a set of urls. We show below a few examples of queries on the schema of Figure 1 which has been integrated with the thesaurus hierarchies of Figures 2 and 3.

1. Sources about actors?

   ```
   select d.url
   from d in Actors
   ```

   This query selects source descriptions, instances of class Actor.
2. Sources about (any kind of) painters ?

   ```
   select d.url
   from d in Actors
   where d.term in d.tree(``painter'')
   ```

   This query selects all source descriptions, instances of class Actor for which the term used in the description is *painter* or one of its narrower terms.
3. Sources about activities concerning man-made objects of the renaissance period?

[20] The O_2 object model is an hybrid model which includes objects encapsulated in classes and typed values which are not encapsulated into any class.

```
select d.url
from d in Activities
where d.associated_with.of_period.term in
d.associated_with.of_period.tree
                          (``renaissance'')
```

This query selects all activities (instances of class Activity) associated with an object (instance of class Man-Made Object) of the *renaissance* period (note that of-period is a property of type Period).

4. Sources about Picasso as a sculptor?

```
select d.url
  from d in Actors
 where d.name = ``Picasso''
   and d.term in d.tree( ``sculptor'' )
```

This query selects all objects, instances of class Actor, whose name is "Picasso" and the term used to describe the actor is *sculptor* or one of its narrower terms. In this query, property name is a value property of type String.

5. Sources about painters of sculptures?

```
select a.performed_by.url
  from a in Activities
 where a.performed_by.term in
     a.performed_by.tree(``painter'')
 and a.associated_with.term in
   a.associated_with.tree(``sculpture'')
```

This query selects all sources, instances of class Actor, (note that role property performed_by of class Activity is of type Actor), where the term associated with the actor is *painter* or one of its narrower terms and the term associated with a man-made object (note that role property associated_with of class Activity is of type Man Made Object) is *sculpture* or one of its narrower terms.

5.2 Another Thesaurus Implementation

The above implementation takes advantage of the efficient optimization of OQL except in the presence of method *tree* in the where clause which allows to deduce all descriptions in the DB that use a narrower term of the one present in the query. This thesaurus traversal not only is costly but may also lead to non optimal query execution plans. This is particularly true for complex queries such as the last one in the above examples.

Therefore even though querying in such a model only requires a limited form of deduction on the *btg* relation in concept thesauri, this facility becomes a central issue when querying description bases including thesauri with thousands of terms. The idea is then to transform thesaurus traversal queries into equivalent interval queries on a linear domain, which queries could then be efficiently answered by standard DBMS query languages without deduction mechanisms.

To achieve this, thesaurus terms are replaced by labels for which a convenient total order exists. The thesaurus tree is said to be *linearized*, as explained below.

The maximal degree or fan-out in any thesaurus is denoted by max, which means that each thesaurus term has at most max direct sons in the hierarchy. Assume for the time being that $max = 9$. At each level sibling sons are ranked from left to right, with rank in $[1, max]$. Let n be a node in the tree. We label n with a string over the integers i, $1 \leq i \leq max$. The string length is d if n is at depth d in the tree. The i-th character in the label of n is equal to m if the i-th node in the path from root to n has for a rank m (its is the mth of its siblings from left to right). Figure 10 gives an example of labeled thesaurus with $max = 9$ where each term is only represented by its rank.

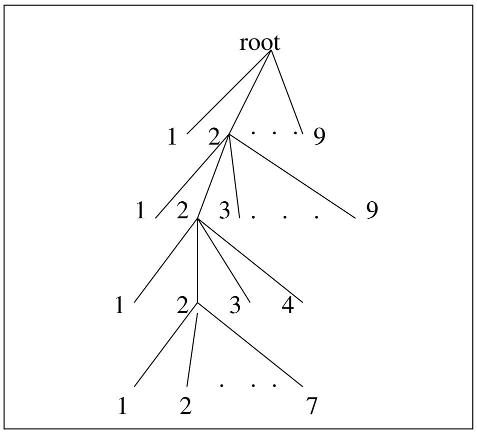

Fig. 10. Linearized Representation of a Thesaurus

Let $label(t)$ denote the label of term t. Then the ascending lexical order on term labels is a total order with the following property : all labels in the sub-tree with root t are larger than $label(t)$ and smaller than $label(t')$ where t' is the next sibling of t on the right in the thesaurus. As an example (max=9), the next sibling of node n with label 22 has for a label 23 and the descendants of n are labeled by 221, 2221, 2222,..., 2227, 223 and 224.

Then in a description stored in the DB, a thesaurus term is represented by a string label and descriptions can be indexed (with the system B-tree) on the term label as any regular O_2 object. Further a typical query such as descendants of term t (obtained by $tree(t)$ in the above implementation), becomes an interval query on node labels : if the term n is labeled by 222, then $tree(n)$ becomes the interval query $[222, 223[$. More generally, let $next(n)$ be the label of the next sibling of n on the right. Query "t in $tree(n)$" becomes the interval query "$label(t)$ between $[label(n), label(next(n))[$". As an example, the above OQL query for "Painters of sculptures" is rewritten as:

```
select a.performed_by.url
  from a in Activities
       lb in LActors,
       lm in LManMadeObjects
 where lb.term = ``painter''
 and lm.term = ``sculture''
 and a.performed_by.label
              between (lb.label, lb.nextlabel)
```

```
and a.associated-with.label
          between (lm.label, lm.nextlabel)
```

We suppose that with each class c is associated a class Lc defining the labels of the concept thesaurus for c, i.e. defining for each term t, its label and its next sibling label. With each class `Lc` is associated the entry point `Lcs` which is a collection of objects of class `Lc`. For example, the range of `lb` is the collection (labels) `LActors`. It corresponds to the term *painter* in the concept thesaurus of Actor and `lb.nextlabel` is the label of the next sibling of this term in the same thesaurus.

Compared to the same query with the tree implementation of the thesaurus, the two last clauses (tree traversal) are replaced by two selections and two interval queries on labels of the collections `LActors` and `LManMadeObjects`.

Three remarks are noteworthy. First, the price to be paid is the two supplementary selection clauses on labels. Of course `LActors` and `LManMadeObjects` must be indexed on the *term* attribute. Second, the descriptions must be preprocessed in order to replace terms by labels. Any change in the label of a term must be propagated to the descriptions. However, the advantage of this solution is two-fold : it allows to process tree traversals by standard database optimization techniques on interval queries and the performance gain to be obtained should be significant for queries involving large thesauri and several criteria on thesaurus terms.

Last, if the query involves several thesauri, the query can be transformed into a hyperrectangle query on a multidimensional space of labels. As an example, take the query above "Painters who painted sculptures". Any s_1 description having for a term a member in $tree(\text{“}sculptures'')$ and any s_2 description having for a term, a member in $tree(\text{“}painter'')$ is a candidate for the answer. Such a pair is a point in the two dimensional space with coordinates $a.performed\_by.label$ and $a.associated\_with.label$. Then the two last *and* clauses can be replaced by a window query. This is illustrated by Figure 11: all points contained in the rectangle represent couples of descriptions on Actor and Man-Made Object (urls) candidates for the query answer. If queries involving both a thesaurus traversal on the concept thesaurus of Actor and Man-Made Object are frequent, it is worth creating a *2-dimensional index* on Actor labels and Man-Made Object labels.

2-dimensional (Spatial) indexing methods[23] as well as *spatial query processing strategies*[41] are not yet fully integrated in the kernel of off the shelf DBMS. However a current trend of these DBMS is to provide simple and fairly efficient extensions to handle spatial data. As an example, [45] describes the spatial extension of the relational DBMS Oracle 8i.

We end up this presentation with a discussion on the choice of the maximal value of the fan-out max. Most thesauri have a maximal fan-out of the order of 100. As an example, the *AAT* thesaurus fan-out is 70. Then a reasonable value for max is 128 allowing to code a node label by a string of characters, each character being chosen in a vocabulary of 128 characters. If a node n with label l has a degree that happens to be larger than max[21] the following simple splitting strategy can be followed (see Figure 12 which shows the tree of Figure 10 after the split resulting from the insertion of a son to the node with label 2, $max = 9$):

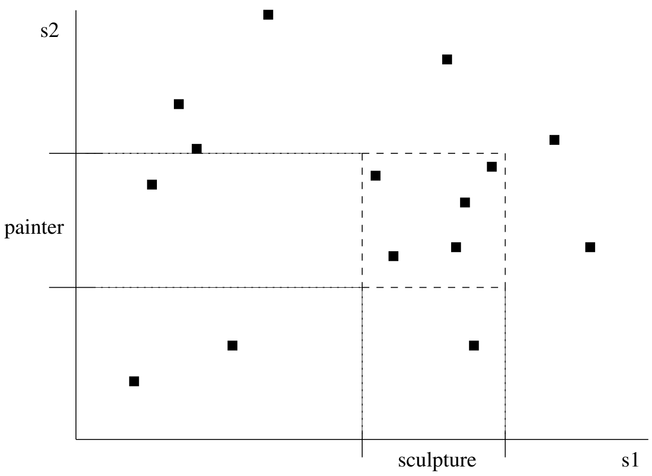

Fig. 11. Two-dimensional Thesaurus Index

1. Create two sons for n with label 1 and 2. These two sons n_1 and n_2 are not associated with any term.
2. Assign to n_1 the $(max + 1)/2$ left sons of n and to n_2 the $(max + 1)/2$ right sons. Let $l.i$ be the label of the ith son of n prior to the split. After the split if $i \leq max/2$, its label becomes $l.1.i$, else it becomes $l.2.i$. This update has to be propagated recursively down the tree.

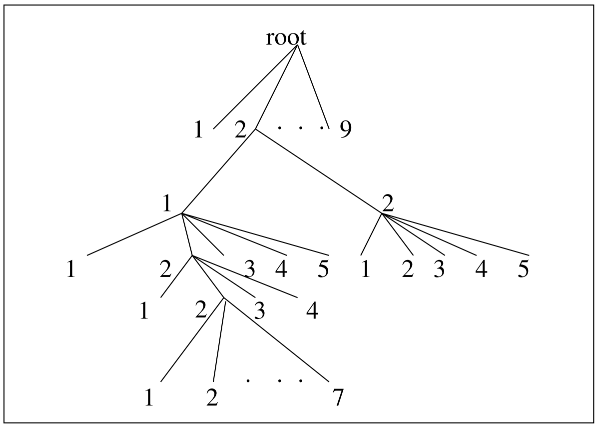

Fig. 12. Splitting a Linearized Thesaurus

6 Future Work

In this paper, we have presented a modular, component based approach to the construction of metadata schemas based on

21 Either upon the thesaurus creation or upon a term insertion posterior to the creation.

the integration of ontologies and thesaurus hierarchies. A prototype under development provides tools to support the creation of a metadata schema resulting from (a) the *specification* of connection relations between a thesaurus $\mathcal{T}$ and an ontology $\mathcal{O}$, and (b) the *automatic creation* of concept thesauri. We have also shown two applications using such metadata schema : (i) the automatic creation of *RDF schema* and (ii) the description and querying of source descriptions . Thesauri are represented as tree-structured attribute values, and although not supported by off the shelf DBMS, the use of tree-structured domains is useful in several application areas, see for example [44, 14].

Our examples were taken from the cultural application domain which disposes a large number of thesauri. RCHME [22] (Royal Commission of the Historical Monuments of England), ICONCLASS [23], ULAN[24] (United List of Artist Names), Thesaurus de l'Architecture of the French Ministry of Culture[25] are some of these thesauri. Nevertheless, the presented approach can also be applied to other scientific domains (e.g. medicine, biology or chemistry) or electronic commerce (e.g. electronic catalogue) applications that dispose rich classification structures.

An interesting issue concerns the specification of the connection relation. Whereas, this relation can be specified manually for a limited number of terms and concepts, its creation gets cumbersome when the number of connected terms and concepts increases. There are two possible solutions to this problem. First, thesaurus terms are often used for indexing documents. The existence of these terms at the data level might then be exploited for the automatic creation of the connection relation by using data mining techniques. The second solution consists in the definition of simple declarative queries (e.g. path expressions) for extracting sets of thesaurus terms.

As described in Section 4, the resulting metadata schema can be perceived as the domain model in mediation based systems and it plays an essential role in achieving *semantic interoperability* between the sources. We are currently studying the use of the metadata schema to describe the structure of XML sources by establishing appropriate *mapping rules* between the XML DTDs and the metadata schemas in the spirit of the Xyleme Project[26]. We intend to implement a mediator that takes advantage of both the semantic source descriptions and the structural descriptions to mediate user queries to underlying information sources.

We are currently applying this approach to the definition of semantically rich domain models in the context of the *C-Web* project[27].

C-Web is primarily a resource discovery and information integration system for specific user communities on the Web, with a mediator-based architecture. As a resource discovery system, it allows to describe existing information resources with structured metadata, acting as surrogates of the actual documents. A Web resource is viewed by the mediator as a valid XML document or a collection of valid XML documents. C-Web aims at selecting documents according to their description in a first step, and to send in a second step to selected sources a query in a standard XML query language and to integrate the result. In this context we are extending the description model of Section 4 so that the mediator can automatically generate XML queries from (i) the user query and (ii) the information found in the source description of sources being selected.

[22] http://www.rchme.gov.uk/thesaurus/mon_types/default.htm
[23] http://iconclass.let.ruu.nl/home.html
[24] http://shiva.pub.getty.edu/ulan_browser/
[25] http://www.culture.gouv.fr/documentation/thesarch/pres.htm
[26] http://www.xyleme.com
[27] http://cweb.inria.fr/cwebproj.html

Acknowledgments We would like to thank Vassilis Christophides for his precious comments on preliminary versions of this paper. The authors also thank Alain Michard and Marie-Christine Rousset for all the fruitful discussions, as well as Martin Doerr for his help in the understanding of the ICOM/CIDOC Reference Model. The authors are also indebted to the anonymous referees for their comments that helped improving the final version of this paper.

References

1. S. Abiteboul and P. Kanellakis. Object identity as a query language primitive. In *Proc. of ACM SIGMOD Int. Conference on Management of Data*, Portland, Oregon, June 1989.
2. S. Abiteboul, D. Quass, J. McHugh, J. Widom, and J. Wiener. The Lorel Query Language for Semistructured Data. *International Journal on Digital Libraries (JODL 97)*, 1996. URL: ftp://ftp.inria.fr/INRIA/Projects/verso/VersoReport-101.ps.gz.
3. J. Ordille A.Y. Levy, A. Rajaraman. Querying Heterogeneous Information Sources Using Source Descriptions. In *Proc. of the Int. Conference on Very Large Databases*, pages 251–262, Bombay, India, September 1996. Morgan Kaufmann.
4. Z. Bar-Yossef, Y. Kanza, Y. Kogan, W. Nutt, and Y. Sagiv. Querying Semantically Tagged Documents on the World-Wide Web. In *Proc. of the 4th Workshop on Next Generation Information Technologies and Systems, NGITS'99*, Zikhron-Yaakov (Israel), July 1999.
5. C. Baru, A. Gupta, B. Ludaescher, R. Marciano, Y. Papakonstantinou, and P. Velikhov. XML-Based Information Mediation with MIX. In Exhibitions Program of ACM SIGMOD Int. Conference on Management of Data, June 1999.
6. C. Batini, M. Lenzerini, and S. B. Navathe. A comparative analysis of methodologies for database schema integration. *ACM Computing Surveys*, 18(4):323–364, December 1986.
7. A. Borgida. Description Logics in Data Management. *IEEE Transactions on Knowledge and Data Engineering*, 7(5):671–682, 1995.
8. A. Borgida, R. Brachman, D. McGuiness, and L. Resnick. CLASSIC: A Structural Data Model for Objects. In *Proceedings of the 1989 ACM SIGMOD International Conference on the Management of Data*, pages 58–66, Portland, Oregon, June 1989.
9. D. Brickley and R.V. Guha. Resource Description Framework (RDF) Schema Specification 1.0. Technical Report CR-rdf-schema-20000327, W3C, March 2000. W3C Candidate Recommendation.

10. Diego Calvanese, Giuseppe De Giacomo, Maurizio Lenzerini, Daniele Nardi, and Riccardo Rosati. Description logic framework for information integration. In *Proc. of the 6th Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR'98)*, 2-13 1998.
11. S. Chawathe, H. Garcia-Molina, J. Hammer, K. Ireland, Y. Papakonstantinou, J. Ullman, and J. Widom. The TSIMMIS project: Integration of heterogeneous information sources. In *Proceedings of IPSJ Conference*, Tokyo, Japan, October 1994. TSIMMIS project: http://www-db.stanford.edu/tsimmis.
12. C. Y. Chee, Y. Arens, C. A. Knoblock, and C. N. Hsu. Retrieving and Integrating Data from Multiple Information Sources. *International Journal of Intelligent and Cooperative Information Systems*, 2(2):127–158, 1993.
13. V. Christophides, S. Cluet, and J. Siméon. On Wrapping Query Languages and Efficient XML Integration. In *Proc. of ACM SIGMOD Int. Conference on Management of Data*, Dallas,USA, May 2000.
14. W. W. Chu, M. A. Merzbacher, and L. Berkovich. The design and implementation of CoBase. *SIGMOD Record (ACM Special Interest Group on Management of Data)*, 22(2):517–522, June 1993.
15. S. Cluet. Designing OQL: Allowing Objects to be Queried. *Information Systems*, 23(5), 1998.
16. S. Cluet, C. Delobel, J. Simeon, and K. Smaga. Your Mediators Need Data Conversion. In *Proceedings ACM SIGMOD Int. Conference on Management of Data*, Seattle, Washington, June 1998.
17. C. Collet, M. Huhns, and W. Shem. Resource Integration Using a Large Knowledge Base in Carnot. *IEEE Computer*, pages 55–62, December 1991.
18. Dublin Core Metadata Initiative. http://purl.oclc.org/dc/.
19. M. Doerr and N. Crofts. Electronic organization on diverse data - the role of an object oriented reference model. In *Proceedings of 1998 CIDOC Conference*, Melbourne, Australia, October 1998.
20. M. Doerr and I. Fundulaki. A proposal on extended links semantics. Technical Report TR-215, Institute of Computer Science-FORTH, March 1998.
21. Dieter Fensel, Stefan Decker, Michael Erdmann, and Rudi Studer. Ontobroker: Or How to Enable Intelligent Access to the WWW. In *Proceedings of the 11th Workshop on Knowledge Acquisition, Modeling, and Management*, Banff, Canada, April 1998.
22. Bancilhon Francois, Delobel Claude, and Kanellakis Paris. *Building an Object-Oriented Database System : The Story of O_2*. Morgan Kaufman, 1992.
23. V. Gaede and O. Günther. Multidimensional Access Methods. *ACM Computing Surveys*, 30(2):170–231, 1998.
24. H. Garcia-Molina, J. Hammer, K. Ireland, Y. Papakonstantinou, J. Ullman, and J. Widom. Integrating and Accessing Heterogeneous Information Sources in TSIMMIS. In *Proc. of the AAAI Symposium on Information Gathering*, pages 61–64, Stanford, California, March 1995.
25. M. R. Genesereth, Arthur M. Keller, and O. Duschka. Infomaster: An Information Integration System. In J. Peckman, editor, *Proc. ACM SIGMOD International Conference on Management of Data*, Tuscon Arizona, USA, May 1997.
26. F. Goasdoué, V. Lattés, and M-C Rousset. The use of CARIN language and algorithms for information integration: The PICSEL System. To appear in the International Journal of Cooperative Information Systems.
27. N. Guarino. Understanding, Building, and Using Ontologies. *International Journal of Human and Computer Studies*, 46(2/3):293–310, 1997.
28. N. Guarino, C. Masolo, and G. Vetere. OntoSeek: Content-Based Access to the Web. *IEEE Intelligent Systems*, pages 70–79, May/June 1999.
29. J. Heflin, J. Hendler, and S. Luke. Reading Between the Lines: Using SHOE to Discover Implicit Knowledge from the Web. In *In Proc. of AAAI Workshop on Artificial Intelligence and Information Integration*, 1998.
30. Richard Hull. Managing semantic heterogeneity in databases: a theoretical prospective. In *PODS '97. Proceedings of the 16th ACM SIGMOD Int. Conference in Management of Data*, pages 51–61, New York, USA, 1997.
31. Documentation - Guidelines for the establishment and development of monolingual thesauri. International Organization for Standardization, 11 1986. Ref. No ISO 2788-1986.
32. V. Kashyap and A. Sheth. *Cooperative Information Systems, Trends and Directions*, chapter Semantic Heterogeneity in Global Information Systems: the Role of Metadata, Context and Ontologies. Academic Press, 1998.
33. K. Knight and S. Luk. Building a Large-Scale Knowedge Base for Machine Translation. In *Proc. of the American Association of Artificial Intelligence AAAI-94*, Seattle, WA, 1994.
34. Y. Kogan, D. Michaeli, Y. Sagiv, and O.Shmueli. Utilizing the Multiple Facets of WWW Contents. In *Proc. of the 3rd Workshop on Next Generation Information Technologies and Systems, NGITS'97*, Neve Ilan (Jerusalem), Israel, 1997.
35. D. B. Lenat. CYC: A Large-Scale Investment in Knowledge Infrastructure. *Communications of the ACM*, 38(11):33–38, 1995.
36. W. Litwin, L. Mark, and N. Roussopoulos. Interoperability of multiple autonomous databases. *ACM Computing Surveys*, 22(1):267–293, March 1990.
37. E. Mena, V. Kashyap, A. Sheth, and A. Illarramendi. OBSERVER An Approach for Query Processing in Global Information Systems based on Interoperation across Pre-existing Ontologies. In IEEE Computer Society Press, editor, *Proceedings First IFCIS International Conference on Cooperative Information Systems (CoopIS'96)*, pages 14–25, Brussels, June 1996.
38. R.S. Michalski. *Categories and Concepts, Theoretical Views and Inductive Data Analysis*, chapter Beyond Prototypes and Frames: The Two-Tiered Concept Representation. Academic Press, 1993.
39. A. Michard, V. Christophides, M. Scholl, M. Stapleton, D. Sutcliffe, and A-M. Vercoustre. The Aquarelle Resource Discovery System. *Journal of Computer Networks and ISDN Systems*, 30(13):1185–1200, August 1998.
40. G. A. Miller. WordNet: A Lexical Database for English. *Communications of the ACM*, 38(11):39–41, 1995.
41. A. Papadopoulos, P. Rigaux, and M. Scholl. A performance evaluation of spatial join processing strategies. In F. Lochovsky R. H. Gueting, D. Papadias, editor, *Proc. of SSD'99*, Springer, LNCS 1651, pages 286–307, Hong-Kong, July 1999.
42. Yannis Papakonstantinou, Hector Garcia-Molina, and Jennifer Widom. Object Exchange Across Heterogeneous Information Sources. In *Proc. ICDE Conf.*, March 1995. TSIMMIS project: http://www-db.stanford.edu/tsimmis.
43. E. Pitoura, O.A. Bukhres, and A.K. Elmagarmid. Object orientation in multidatabase systems. *ACM Computing Surveys*, 27(2):141–195, June 1995.

44. P. Rigaux and M. Scholl. Multi-scale partitions: Application to spatial and statistical database. In J.R. Herring M.J. Egenhofer, editor, *Proc. SSD'95*, volume 951 of *LNCS*, pages 170–183, Portland, August 1995.
45. J. Sharma. Oracle8ispatial: Experiences with extensible databases. An Oracle Technical White Paper, May 1999.
46. A.P. Sheth and J.A. Larson. Federated database systems for managing distributed, heterogeneous, and autonomous databases. *ACM Computing Surveys*, 22(1):183–236, March 1990.
47. M. Sintichakis and P. Constantopoulos. A Method for Monolingual Thesauri Merging. In *Proc. 20th International Conference on Research and Development in Information Retrieval, ACM SIGIR*, Philadeplphia PA, USA, July 1997.
48. D. Soergel. The Art and Architecture Thesaurus, AAT. A critical appraisal. Technical report, College of Library and Information Sciences, University of Maryland, 1995.
49. A. Tomasic, R. Amouroux, P. Bonnet, O. Kapitskaia, H. Naacke, and L. Raschid. The Distributed Information Search Component (DISCO) and the World Wide Web. In *Proc. of ACM SIGMOD Int. Conference in Management of Data*, pages 546–548, Tuscon, Arizona, USA, June 1997.
50. G. Wiederhold. Mediaton in information systems. *ACM Computing Surveys*, 27(2):265–267, June 1995.