Integrating Scientific Data through External, Concept-based Annotations

Michael Gertz Kai-Uwe Sattler
Department of Computer Science, University of California, Davis
One Shields Avenue, Davis, CA 95616-8562, USA
{gertz|sattler}@cs.ucdavis.edu

Abstract

In several scientific application domains, such as the computational sciences, the transparent and integrated access to distributed and heterogeneous data sources is key to leveraging the knowledge and findings of researchers. Standard database integration approaches, however, are either not applicable or insufficient because of lack of local and global schema structures. In these application domains, data integration often occurs manually in that researchers collect data and categorize them using "semantic indexing", in the most simple case through local bookmarking, which leaves them without appropriate data query, sharing, and management mechanisms.

In this paper, we present a data integration technique suitable for such application domains. This technique is based on the notion of controlled data annotations, resembling the idea of associating semantic rich metadata with diverse types of data, including images and text-based documents. Using concept like structures defined by scientists, data annotations allows scientists to link such Web-accessible data at different levels of granularity to concepts. Annotated data describing instances of such concepts then provides for sophisticated query schemes that researchers can employ to query the distributed data in an integrated and transparent fashion. We present our data annotation framework in the context of the Neurosciences where researchers employ concepts and annotations to integrate and query diverse types of data managed and distributed among individual research groups.

1 Introduction

The past few years have witnessed a steady improvement of standard database integration techniques to deal and scale with data on the Web [12, 2]. Data wrapper and mediator techniques are just a few of such approaches that basically investigate the schemas of Web-accessible data sources and either materialize (wrapped) data into a globally accessible data warehouse or provide for global queries mediated among the sources.

While these approaches are suitable for data sources that exhibit database like structures, they are not applicable or suited for integrating data from sources where such schema-like structures are almost non-existent. The most prominent example of such scenarios are the computational sciences, such as computational biology, Neurosciences, astrophysics or medicine. In these domains, the data typically range from database like sources over Web documents to high-resolution images. For example, while an image may exhibit the same information content as a few rows of a relation or a paragraph in a text document, it is almost impossible to automatically integrate and represent these "data" or rather regions of interest in one globally accessible structure. The typical way in which users integrate such data is that they categorize them using their domain knowledge and expertise and then build structures that either cluster the data in a materialized fashion or simply keep link structures to the Web data.

In this paper, we present a coherent framework that allows scientist to utilize *concepts*, representing well-defined, agreed upon domain specific features of interest, to semantically enrich Web-accessible data and to link such data to concepts as instances of such concepts. While concepts and relationships among concepts then can be considered as a global schema, the data, which can be regions of interest (ROIs) in a document and are further described by scientists in terms of their properties, represent instances of such concepts. Relationships between concepts can be inherited

to relationships among data and thus can be considered as typed (Web-)links among data that are external from the data. In other terms, annotations represent metadata that follow the structures specified by concepts. Concepts, annotations and Web data then build a graph structure for which we present a flexible query language and its realization as basis for different services that can be build on annotations, such as a search engine or concept browser.

The following gives an example from a project currently conducted at the Center for Neuroscience at UC Davis in the context of the Human Brain Project [17, 22, 31]. In this project, Neuroscientists at different sites generate various types of data related to the human brain, including high resolution images of anatomical structures, bio-physiological or chemical properties of cells, nuclei, and their connections in form of graphs or charts, and text data describing findings regarding specific features of interest, such as the processing of information by cell structures. With none of theses data a schema is associated but they are all accessible through the Web where, e.g., centralized image or chart registries are employed by researchers to publish the data they generate.

Assume a researcher browsing a newly generated set of images published by a research group (left browser window in Figure 1). In one of these images, the researcher identifies a region containing a cell structure of a known type A. Ideally, she would like to annotate this region and "link" it to a conceptual structure that has been defined in the specific research context. This structure defines a concept (similar to a metadata schema) that details general, agreed upon properties of the cell type A such as a definition, terms typically used to refer to the cell type, and other general properties of that concept. Through this linkage, the identified region in the image then can be understood as an *instance* of the concept and which then is filled out by the researcher (e.g., specific values for that instance in this image). Assume the researcher knows about another group that deals with similar images, generated in the context of a different experiment. She pulls up one such Web document which contains some images as well as some descriptive text (right browser window). The image in this document exhibits a region containing the cell type A. Again, she would like to link this region to the same concept used for the first image, thus specifying another instance of this concept with perhaps different values for the properties. The aspect of linking (annotating) these regions to a concept and instantiating concept properties is indicated through the lines and boxes at the marked regions in Figure 1. Linking different regions of interest, independent of whether these regions are

Figure 1: Scenario of Concept-based Annotations for Scientific Data

within an image or text document then corresponds to a data integration process where the researcher manually classifies data and associates them with respective structures. It should be noted that none of the aspects described in the above scenarios can be accomplished by applying known integration methods for Web data, unless sophisticated feature detection mechanisms can be provided for the automatic discovering of concept instances.

In the following, we present the conceptual framework that supports the modeling and querying of scenarios such as the above. The annotation graph model described in Section 2 provides expressive means to specify diverse types of relationships among concepts, documents, and regions of interest, and it also provides query mechanisms to traverse such graph structures representing integrated data. In Section 3, we then outline the integration process for scientific data in the context of the annotation graph model. Section 4 discusses the realization of an integration platform implementing the model and services based thereon. After a presentation of related work in Section 5, we conclude the paper in Section 6 with a summary and some future research.

2 Integration Model

Integrating data from different sources requires correspondences among schemas to some extent. Typically, the sources to be integrated offer some kind of schema, which represents a sub-/superset of a global schema. However, application domains as those described in the introduction in general

lack such kind of data source schemas. Instead, correspondences among structures and data exist on a semantic level only where regions of interest in documents can be understood as instances (data) of certain concepts (schemas). In this respect, integrating documents and the data they represent means to enable queries on automatically extracted or manually assigned features of the documents. Such features can be represented through annotations. An annotation of a document basically can be understood as a typed link between an object (so-called *region of interest* or *ROI*) in the document and a domain specific concept representing a metadata template. Associated with such an annotation are property values that describe the feature according to the concept. In order to specify, represent, and in particular query concepts, annotations and documents in a uniform fashion, these types of information need to be not only modeled appropriately, but a respective model should also be easy to implement and use in different data management and retrieval tasks.

The model should be extensible with respect to different conceptual structures, such as Dublin Core [7] in the most simple case, Neuronames [21] or UMLS [5] as examples for specific Neuroscience or medical vocabularies, as well as complex ontology-like structures. Such conceptual structures are typically developed in a collaborative fashion and represent various domain specific aspects. Note that conceptual models are not the primary focus of our work. They are mainly used to provide common semantic "anchors" for annotations associated with documents. Thus, we rely on work in the context of knowledge models as developed, e.g., by the Semantic Web community. In the following, we detail our annotation graph model that is based on a simple conceptual structure underlying our approach to annotate regions of interest in documents for data integration purposes.

2.1 Annotation Graph Model

The annotation graph model represents concepts, annotations and (portions of) documents as nodes in a graph and are linked by typed relationships. The model serves as an integration model defining connections among a global metadata schema and distributed and heterogeneous document collections.

Assume a set $T = \{\mathit{String}, \mathit{Int}, \mathit{Date}, \dots\}$ describing a set of simple data types. Let $\mathrm{dom}(T)$ denote the set of all possible values for T. In the annotation graph model, a property of a node is defined as an identifier and a type $\mathit{PDef} = \mathit{String} \times T$. An instantiation of a property consists of an identifier and a value $\mathit{PVal} = \mathit{String} \times \mathrm{dom}(T)$.

As outlined above, concepts provide templates for annotations that are associated with documents. In our model, concept are represented by a simple yet extensible form of *concept nodes*. We assume that each concept node has (1) a concept identifier (typically the preferred name for the concept), (2) some descriptive terms or phrases typically used for the concept in data retrieval tasks ($\mathbb{P}\,\mathit{Term}, \mathit{Term} \subseteq \mathit{String}$), (3) a natural language definition ($\mathit{Def} \subseteq \mathit{String}$) that associates an agreed upon, well-defined meaning with the concept, and (4) a set of concept property specifications $\mathbb{P}\,\mathit{PDef}$. In this respect, a concept definition is similar to a class definition in the context of object-oriented modeling. In the following, we denote the set of all concepts by $\mathcal{C}$, where $\mathcal{C} \subseteq \mathit{Term} \times \mathit{Def} \times \mathbb{P}\,\mathit{Term} \times \mathbb{P}\,\mathit{PDef}$.

The second type of node in our graph model represents Web accessible documents. A document is identified by its URI (Uniform Resource Identifier) of type *String* and optionally has a set of document properties such as author, title, etc. We denote the set of all Web documents by $\mathcal{D}$.

Finally, annotation nodes provide the basis for specifying links between concepts and documents. The set of annotations $\mathcal{A}$ is defined as $\mathcal{A} \subseteq \mathit{String} \times \mathit{Date} \times \mathbb{P}\,\mathit{PVal}$. For a tuple $(\mathit{creator}, \mathit{created}, \mathit{pvals}) \in \mathcal{A}$, *creator* is the author of the annotation, *created* is the creation date/time, *pvals* is a set property instantiations. The set of all nodes $\mathcal{V}$ in an annotation graph is now defined as $\mathcal{V} = \mathcal{A} \cup \mathcal{C} \cup \mathcal{D}$. Links between nodes are represented as directed, typed edges, to which property instantiations are optionally assigned. The types of edges are drawn from the specified concepts (see below) and the property instantiations are determined by the associated concept. Finally, the set $\mathcal{E}$ of all edges is define as

$$\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V} \times \mathcal{C} \times \mathbb{P}\,\mathit{PVal}$$

The meaning of the components of an edge $(\mathit{from}, \mathit{to}, \mathit{type}, \mathit{pvals}) \in \mathcal{E}$ is as follows: the edge connects

the node *from* with the node *to* (in that direction). With the edge the concept *type* is associated, and *pvals* is a set property instantiations.

For example, to represent an "is-a" relationship between two concepts c_{super} and c_{sub} we define an edge $(c_{sub}, c_{super}, isA, \emptyset)$ where *isA* is the identifier of a *relationship type concept*. In an implementation of this model, the kind of nodes connected by an edge should be further restricted, but this is not addressed in this basic model.

Using these definitions, our annotation graph model comprises both the metadata components (concepts) and data components (annotations and documents). An instance of the model then is represented by a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. Naturally, nodes can be connected via edges of arbitrary types. However, in most cases only edges of certain types make sense. In order to deal with the special meaning of the different kinds of nodes and how they can be connected, we introduce the following *default relationship type concepts*. They are contained in any specification of a collection of concepts and can be extended by more relationship type concepts to relate concepts to each other in a well-defined way.

- *annotates* is used to represent edges from annotations to documents. If an annotation is specified for a document, a link referring to this relationship type concept is defined between both of them. Since we are interested in fine-grained annotations, e.g., regions in an image or fragments of a text document, we assume a set of *document context descriptions*. A description comprises information about the document context in which an annotation is specified and is modeled as a property of the relationship. In our current system, we employ two types of descriptions. For text-based documents, we employ a tree-structured document model. In particular, we use a transformation mechanism to handle HTML and plain text documents as XML documents. XPath expressions [4] are then used as document context descriptors. For fine-grained annotations in image data, we use a spatial region object and scale information to encode respective context descriptions.

- *annotatedBy* is a relationship type concept representing the inverse of *annotates*.

- the concept *ofConcept* specifies that an annotation is based on a certain concept, i.e., it assigns the concept to the annotated document and instantiates the properties defined by this concept.

- *hasAnnotation* describes the inverse of the relationship *ofConcept*. In addition, each annotation $a \in \mathcal{A}$ must be related to a concept.

- *isA* denotes an "is-a" relationship type concept between concepts, thus implementing inheritance of property definitions, i.e., $\forall e \in \mathcal{E} : e.rel = isA \rightarrow e.from, e.to \in \mathcal{C} \wedge e.from.pdefs \supseteq e.to.pdefs$. As mentioned above, other relationship type concepts can be introduced to model how (base) concepts are related. This can include spatial, temporal, and general semantic types of relationships.

2.2 Query Operations

In order to effectively utilize a collection of annotated documents, certain query operations need to be supported. For a graph-like structure as adopted for our annotation graph model, such operations basically consist of node selection and path traversal. The input for a selection is always a homogeneous set: either one of the basic sets $\mathcal{A}$, $\mathcal{C}$ or $\mathcal{D}$ (but not a union of them) or a derived set resulting from a prior operation. Let S be one of the sets $\mathcal{A}$, $\mathcal{C}$ or $\mathcal{D}$ and $P(s)$ a predicate on $s \in S$. Then the selection operation σ_P is defined as

$$\sigma_P(S) = \{s \mid s \in S \wedge P(s)\}$$

A predicate P is a boolean expression made up of clauses of the form *prop <op> value* and which can be combined through logical connectives. In addition, path expressions of the form $rel_1.rel_2 \ldots rel_n.prop$ specifying the traversal of edges (relationship type concepts) rel_1, rel_2 etc. from the current node to the property *prop* of the target node are allowed as long the result is a single-valued expression. Path traversal enables following links between nodes of the graph. Given a start

node v_s and a relationship type concept rel, the operation ϕ_{rel} returns the set of target nodes based on existing edges:

$$\phi_{\mathit{rel}}(v_s) = \{v_t \mid (v_s, v_t, \mathit{rel}) \in \mathcal{E}\}$$

Since we mainly have to deal with sets of nodes in query expressions, this operation is also defined on a set of nodes $V \in \mathcal{V}$:

$$\Phi_{\mathit{rel}}(V) = \{v_t \mid \exists v_s \in V : (v_s, v_t, \mathit{rel}) \in \mathcal{E}\}$$

A special kind of the path traversal operation is the operation for computing the transitive closure. This operator extends ϕ_{rel} by traversing the path indicated by the relationship rel as long as edges can be found that have not already been visited. The set of all nodes traversed thus is defined as

$$\phi^+_{\mathit{rel}}(v_s) \quad = \quad \{v_t \mid (v_s, v_t, \mathit{rel}) \in \mathcal{E} \vee \exists v_i \in \phi^+_{\mathit{rel}}(v_s) : (v_i, v_t, \mathit{rel}) \in \mathcal{E}\}$$

As for ϕ_{rel}, this operation is defined on a set of nodes, too:

$$\Phi^+_{\mathit{rel}}(V) \quad = \quad \{v_t \mid \exists v_s \in V : (v_s, v_t, \mathit{rel}) \in \mathcal{E} \vee \exists v_i \in \Phi^+_{\mathit{rel}}(v_s) : (v_i, v_t, \mathit{rel}) \in \mathcal{E}\}$$

Using these basic operations, query expressions for selecting nodes and traversing edges can be formulated. Note that the initial set of nodes for a traversal always has to be obtained by applying a selection on one of the basic sets $\mathcal{A}$, $\mathcal{C}$ or $\mathcal{D}$. From this, following the relationships type concepts *ofConcept*, *annotates* etc. allows to go to another type of nodes.

For the purpose of more developer-friendly query formulation, we have designed a simple language in the spirit of XPath. Here, the sets $\mathcal{A}$ (*annotations*), $\mathcal{C}$ (*concepts*), and $\mathcal{D}$ (*documents*) are valid root elements. If views on these sets are defined, they can be used as root elements as well. Selections are formulated by appending [*condition*] to a term. In *condition*, the properties of nodes can be specified and —in combination with the usual operators and logical connectors— used for formulating complex predicates. The Φ-operator is expressed by appending */relship* to the term. *relship* denotes the relationship type concept that has to be used for following the links. The optional + indicates that the transitive closure has to be computed.

The following is a simple example of a query that specifies all documents annotated using a given concept named *C*. First, the desired concept is selected. Then, by traversing the relationships to the annotations (*hasAnnotation*) and then to documents (*annotates*), the final result, here a set of documents, is obtained:

$$\phi_{\mathit{annotates}}(\phi_{\mathit{hasAnnotation}}(\sigma_{\text{name='C'}}(\mathcal{C})))$$

An equivalent query, written in our query language, looks as follows:

concept[name='C']/hasAnnotation/annotates

Additional selections can be applied to each of the intermediate results of the query. As an example, it is possible to restrict the annotations to be considered to a certain author or date. If all those documents are to be retrieved that are either annotated using a concept C or a concept that is a (in)direct subconcept of C (assuming the relationship type concept *subtype* is the inverse of *isA*), the above query is extended to

concept[name='C']/subtype+/hasAnnotation/annotates

After selecting the concept named *C*, all concepts in its transitive closure with respect to the *subtype* relationship are obtained. From this set, the annotated documents are determined as described above. While these examples illustrate the querying of distributed documents in a uniform fashion using concepts, the next example shows how "related" documents can be determined:

document[uri='www...']/annotatedBy/ofConcept/hasAnnotation/annotates

Starting from a document with a given URI, first all annotations for that document are determined. Following the *ofConcept* relationship, their underlying concepts are obtained, and by going "downwards" from there to the annotations and documents, we can find "related" documents (or regions), i.e., documents that have been annotated using the same concepts as the specified document.

Since annotations are first-class objects, they can be linked to more than one document. That is, they can be used for annotating several documents at the same time. In the following, this aspect is utilized by retrieving all documents that are linked to a given document by an annotation based on concept *C*:

document[uri=...]/annotatedBy[ofConcept.cname='C']/annotates

In summary, the proposed annotation graph model and associated query language supports a wide variety of query operations against distributed collections of annotated documents, the most important classes of queries being

- *concept* → *document*: Starting from concept (at the schema level), documents annotated using that concept are retrieved. This operation is equivalent to retrieving the extension of a given class.
- *document* → *concept* → *document*: Documents annotated by the same or a similar concepts are retrieved. This corresponds to a concept-based similarity operation.
- *document* → *document*: In this class of queries, links represented by annotations are utilized for finding related documents. This operation is similar to a join between data objects.

2.3 Views

In collaborative research environments such as those described in the introduction, data in form of documents are continuously generated by researchers at different sites. In order to allow for different foci of interest in an integrated collection of documents, views are an essential mechanism for structuring or grouping concepts (perhaps originating from different existing vocabularies), annotations, and documents according to such foci and research interests. In particular, views allow to support different vocabularies or conceptual structures, as they frequently occur even among research groups that have the same research interest.

In our annotation graph model, a view restricts the set of nodes to be considered in queries. More precisely, a view specifies a (virtual) sub-graph $\mathcal{G}' = (\mathcal{V}', \mathcal{E}')$ of an annotation graph $\mathcal{G}$. Due to the distinction of several node types, $\mathcal{V}'$ is specified through separate queries on the base sets $\mathcal{A}$, $\mathcal{C}$, and $\mathcal{D}$ using query operations presented in Section 2.2. For example, if we want to provide a view containing only (1) concepts that originate from the Neuronames collection, and (2) annotations made this current year, we could define this as follows:

`define view` *my_view* `as`
annotation := *annotation[created >= '01/01/2002']*
concept := *concept[origin = 'Neuronames']*

If a base set is not restricted in the view definition, as in the above case the document set, the complete set is used when querying the view. For individual restricting queries, any valid query expression is allowed as long as it returns a proper subset, e.g., a set $C' \subset C$ of the concept set. Because the set of concepts contains essential concepts like *isA*, *annotates* etc., the set of default relationship type concepts is treated in a special way, guaranteeing the inclusion of the built-in concepts in each view.

It should be noted that we currently do not provide a separate view definition language. Instead a view is defined using a dedicated service. A view is used in a query by simply giving its name as an additional parameter to the invocation of the query processor. The view resolution itself is performed as part of the query translation, which is detailed in Section 4 .

3 Integration Process

The primary task in integrating scientific data (or documents) using the annotation graph model is to create concepts and relationships among them, and to establish mappings between regions of interest in documents and concepts using annotations. In this context, two major issues arise:

- The mapping cannot be done automatically because it requires an interpretation of the document content. For example, a certain region in an image has to be identified by a researcher as an image of an instance of the concept "cell". While it is conceivable that for text documents such a conceptualization can be done in a semiautomatic fashion, using, e.g., NLP techniques, thesauri etc., in the general case and for images in particular one has to rely on the manual approach.

- How does one define concepts representing a form of global metadata schema? In principle, for a given application domain, a schema consisting of concepts and relationships could be modeled in advance and used for annotating distributed documents. However, often not all domain specific concepts are known in advance but are introduced while researchers investigate the information content of document. This is particular true if new information is discovered that cannot be associated with any existing concept. In this respect, the concept part of an annotation graph can be a dynamic and growing structure, and the integration process has to take this aspect of an "evolving schema" into account.

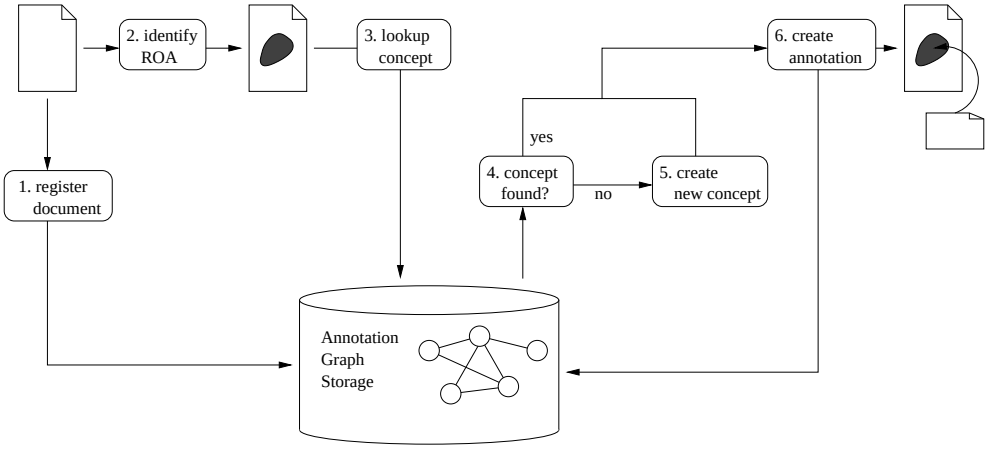

Figure 2: Process of integrating schema-less data using concepts

Due to these characteristics, we propose the following process for integrating data through concept-based annotations (Figure 2):

1. An initial set of agreed-upon concepts is specified, including the most important and obvious relationships types concepts such as sub/supertype or, as it is the case in for neuroanatomical data about the human brain, spatial relationships, and stored in a centralized annotation graph storage.

2. For annotating a document, that is, integrating the document and its information content into the global collection, an appropriate concept has to be chosen. If no such concept exists, a new concept is specified (perhaps by deriving it from an existing concept) and added to the concept collection. In addition to this, relationships to other concepts can be specified.

Of particular interest in data integration scenarios is also the handling of structural and data conflicts. In [14], we have detailed an approach to deal with such scenarios in the context of annotating image data. Here, we will give only a brief overview of the basic ideas.

An annotation graph typically is constructed in a collaborative fashion. That is, several users introduce and modify concepts and annotations over time. In order for the annotations to be useful in

data retrieval, services need to be provided that check for a certain degree of consistency in terms of how concepts and annotations have been used. There has been quite a lot of work in the context of the creating and sharing ontology like structures, e.g., [13, 29, 10], and it is well known that consistency issues in such structures pose a hard and in general not completely solvable problem.

The basic idea we employ to deal with such scenarios is that if concepts, as core components of such ontological structures, are used to annotate data, such annotations can provide important input to some consistency aspects. Consider the scenario where some agreed upon base concepts have been introduced and these concepts are used by different users in annotating documents. Different users might have a different focus and thus it is totally reasonable that, e.g., different concepts are used to annotate the same documents and regions of interest. This is a natural aspect in collaborative research environments, in particular if features are detected in data for which no concepts have been introduced yet.

In our current system prototype, we use information about how annotations are used to provide users with some feedback regarding the creation of annotations and concept. This feedback then can be used to investigate possible inconsistencies by, e.g., a group of domain experts who review submitted concepts or annotations. Among others, the following important scenarios are considered:

Annotation granularity and redundant concepts. Assume a user specifies an annotation A_1 for a document based on a concept C_1. With each annotation, a context is associated that details the granularity of the annotation. This can be either a spatial object (e.g., polygon) in case of an image, or an XPath expression in case of a text document (see also Section 2.1). During annotation creation time, the system checks whether there already exist annotations A' for the same document such that there is a containment relationship between annotation contexts (which are well-defined for spatial objects and subtree structures). Assume there is another annotation A_2, based on concept C_2 such that the context for A_2 is contained in the context for A_1. If both annotations refer to the same concept, then for either A_1 or A_2 there should be a modification. If A_1 and A_2 use the same context but different concepts, then this either indicates an inconsistency or just a different viewpoint among the users. In particular the first aspect turns out to be very valuable since users get an insight into how other users interpret the data in a perhaps different research context. Other scenarios based on containment of document contexts and relationships (in particular "is-a" relationships) among used concepts are checked by the system and are discussed in more detail in [14].

Propagating modifications. The deletion of a concept naturally implies the deletion of all annotations that are based on this concept. However, in case an "is-a" relationship exists between the deleted concept and some other concept, another option would be to "relink" the annotations either to the more specific or more abstract concept, depending on the direction of the "is-a" relationship. Such a scenario can take place in a user-guided fashion and preserves annotations and concept instance properties to a certain degree. Finally, the deletion of a document results in the deletion of all annotations associated with that document.

In summary, document and data integration as proposed in this paper is an iterative, evolving process with continuous modifications at the concept and annotation level. We support this by providing tools for querying and browsing concepts and by implementing mechanisms for compatibility and integrity checks.

4 Implementation of the Integration Platform

In this section, we describe the implementation of the presented framework as well as its application in our project conducted in the context of the Human Brain Project. In this project, brain slices are digitally photographed under a microscope and utilized by researchers who identify and mark specific regions (e.g., individual cells or cell structures) and assign concepts (e.g., a cell type) to these regions. The annotation graph model is used to represent content descriptive metadata templates in form of concepts as well as relationships among such concepts. In this way, it provides an integrated view on the annotated features of documents distributed among Web sources. Furthermore, the query model allows to formulate declarative queries for traversing the graph and retrieving data.

The integration platform is realized by an annotation server, which provides basic services for

defining concepts, associating annotations with images and text documents, and evaluating queries. The architecture of this server is shown in Figure 3.

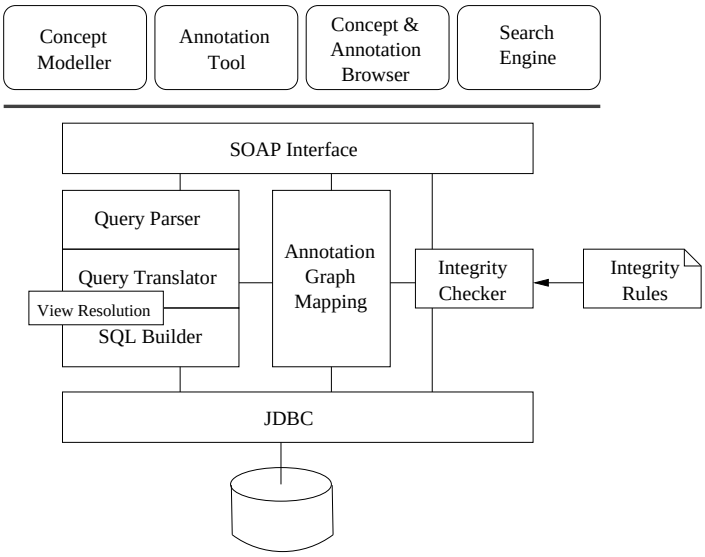

Figure 3: Architecture of the annotation server

The main component of the system is the graph mapping module. It realizes the mapping of the conceptual model of an annotation graph consisting of nodes and edges to relations stored in a relational database. The database has the following relation schemas:

CONCEPT*(id, def)*
CONCEPTTERM*(id* → CONCEPT*, term)*
CONCEPTPDEF*(id* → CONCEPT*, kind, name)*
CONCEPTRELSHIP*(id*→ CONCEPT*, fromTable, toTable)*
DOCUMENT *(id, uri, title)*
ANNOTATION*(id, creator, created)*
ANNOTATIONPVAL *(id*→ ANNOTATION*, name, val)*
RELSHIP *(id, from, to, rel*→ CONCEPT*)*
RELSHIPPVAL *(id*→ RELSHIP*, name, val)*

Concepts are stored in the normalized relations CONCEPT, CONCEPTTERM (for terms), and CONCEPTPDEF (for property definitions). If a concept is a relationship type concept, an additional tuple is created in the relation CONCEPTRELSHIP specifying which kind of nodes (identified by their names) are linked. Information about documents is managed in the DOCUMENT relation, and annotations are stored in the relation ANNOTATION together with the property values in ANNOTATIONPVAL. Finally, the relations RELSHIP and RELSHIPPVAL are used to manage relationships type concepts and properties of respective instantiations.

The query model is tightly coupled with the mapping module. It consists of the parser for the query language, a translator that transforms a given query into a relational algebra expressions, and an SQL builder for deriving and executing the corresponding SQL query. The transformation from a query expression into relation algebra is specified by a set of rules. We assume the well-known relational algebra operators: σ_{cond}, $\bowtie$, $\bowtie_\theta$, π, ρ (for renaming) as well as the recursive operator α [1]. Utilizing the α operator is a feasible approach since most of today's database systems support some kind of recursive queries, e.g., Oracle with the `CONNECT BY` clause or IBM DB2 with `RECURSIVE UNION`, or allow at least to implement user-defined table functions that can perform transitive closure computation. We use the notation

$$\omega(Exp) \mapsto \overline{\omega}(Exp)$$

to express the transformation of an expression $\omega(Exp)$ into a relational expression $\overline{\omega}(Exp)$ by substituting ω with $\overline{\omega}$. Furthermore, *eval*(Exp) denotes the evaluation of the expression Exp, and *relation*(T) denotes the relation with name T.

The transformation of a query expression into relational algebra can now be specified by the following set of rules:

(1) $\mathcal{A} \mapsto$ ANNOTATION

(2) $\mathcal{C} \mapsto$ CONCEPT

(3) $\mathcal{D} \mapsto$ DOCUMENT

(4) if *eval*$(Exp) \subseteq \mathcal{A}$ then
$\sigma_P(Exp) \mapsto \sigma_{\overline{P}}(Exp) \bowtie \rho_{A_1}(\text{ANNOTATIONPVAL}) \bowtie \cdots \bowtie \rho_{A_n}(\text{ANNOTATIONPVAL})$
Here, $\overline{P}$ is constructed as follows. For each clause c_i of the form "*prop op val*", there is a corresponding clause "A_i.name = '*prop*' $\wedge$ A_i.val *op val*".

(5) if *eval*$(Exp) \subseteq \mathcal{D}$ then $\sigma_P(Exp) \mapsto \sigma_P(Exp)$

(6) if *eval*$(Exp) \subseteq \mathcal{C}$ then $\sigma_P(Exp) \mapsto \sigma_P(Exp \bowtie \text{CONCEPTTERM})$

(7) $\Phi_r(Exp) \mapsto T_{to} \leftarrow \pi_{to}(\sigma_{id=r}(\text{CONCEPTRELSHIP})$;
$\pi_{T_{to}.*}(Exp \bowtie_{id=from} (\sigma_{rel=r}(\text{RELSHIP})) \bowtie_{to=id} \textit{relation}(T_{to}))$

(8) $\Phi_r^+(Exp) \mapsto T_{to} \leftarrow \pi_{to}(\sigma_{id=r}(\text{CONCEPTRELSHIP})$;
$\pi_{T_{to}.*}(Exp \bowtie_{id=from} \alpha(\pi_{from,to}(\sigma_{rel=r}(\text{RELSHIP})) \bowtie_{to=id} \textit{relation}(T_{to}))$

Rules (1)–(3) specify simple substitutions of sets by the corresponding relations. In rules (4)–(6), the selection operator is translated utilizing the normalization of the relations. Rule (7) translates the path traversal operator into a 3-way join between the relations *from*, *to*, and RELSHIP. The transformation of path expressions as part of a selection condition is handled in a very similar way and therefore omitted here. Finally, using rule (8), the path traversal operation based on the transitive closure is translated into an algebra expression that computes all edges of the given relationship type concept and joins them with the *from* and *to* attributes using the α operator.

These transformation rules are always applied in an inside-out manner. For example, for the query expression $\sigma_{\textit{author='Ted'}}(\Phi_{hasAnnotation}(\sigma_{\textit{name='C'}}(\mathcal{C})))$, the resulting algebra expression is

$$\sigma_{\textit{author='Ted'}}(\pi_{\text{ANNOTATION}.*}(\sigma_{\textit{name='C'}}(\text{CONCEPT} \bowtie \text{CONCEPTTERM}) \bowtie_{id=from} \sigma_{\textit{rel=hasAnnotation}}(\text{RELSHIP}) \bowtie_{\textit{to=id}} \text{ANNOTATION} \bowtie \text{ANNOTATIONPVAL}))$$

From this algebra expression, an equivalent SQL query can easily be derived. Only the α operator requires a special treatment. Due to the limitations in the version of the Oracle DBMS used in our implementation, we have implemented this operator by a table function *tclosure (src_id, rel)*, which returns a table of node identifiers of the transitive closure of node *src_id* with regard to the relationship *rel*. In this way, a query like

concept[name='C']/subtype+

is translated into the following SQL query

```
select   ct3.*, t4.*, p5.*
from     CONCEPT c1, CONCEPTTERM t2, table(tclosure(c1.id, "subtype")) as ct3,
         CONCEPTTERM t4, CONCEPTPDEF p5
where    c1.name='C' and c1.id=t2.id and ct3.id=t4.id and ct3.id=p5.id
```

All components of the system are implemented in Java using JDBC for accessing the DBMS. The interface to the services of the system is realized using SOAP. In this way, the annotation server can be used as a Web Service by different (Web-based) applications, e.g., a tool for annotating images or a concept editor/browser.

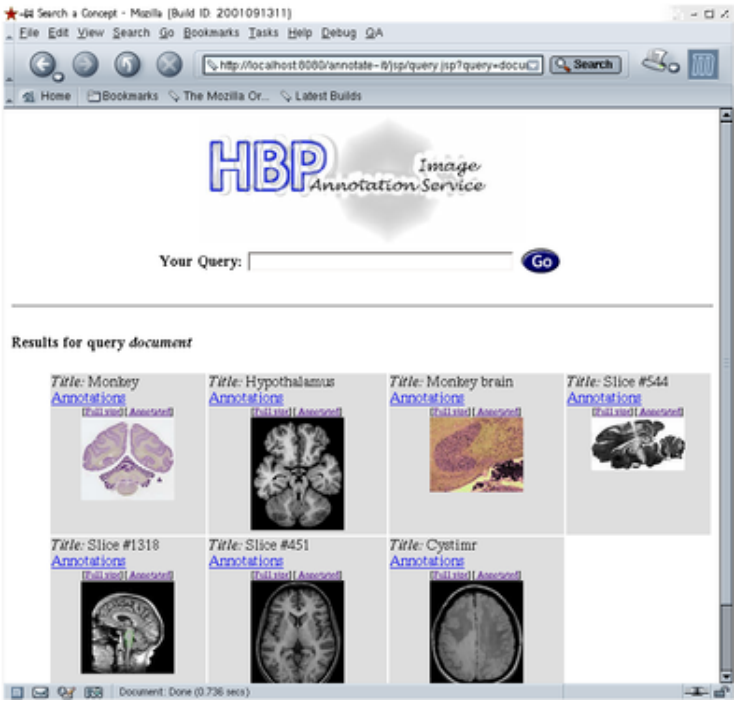

Figure 4: The search engine

The primary application for utilizing the integrated documents and data is a search engine (Figure 4). It evaluates queries formulated in our query language using the query engine of the annotation service and displays the results. Depending on the type of results (annotations, concepts, or documents (regions)) it additionally displays links to the associated objects, i.e,

- for documents (in our application domain these are images), links to the annotations,
- for an annotations, the annotated documents as well as the concepts and instance properties underlying the annotations, and
- for a concept, all annotations based on this concept.

All these links are represented as queries. In this way, both querying and browsing are supported as paradigms for interacting with the integrated data. Furthermore, since a query result always represents a homogeneous set of nodes, a query can be refined. That is, a new query can be applied to the currently shown result set.

The integration of document data is done by researchers using specific annotation tools. In our application scenario, this is a tool for annotating images. This tool allows users to load and display high-resolution images where regions of interest are marked and annotated using the template induced by a previously selected concept. This concept is selected using a browser for querying and traversing the concept part of an annotation graph. If no concept satisfying the selection criteria can be found, e.g., in case of newly discovered structures, a new base concept can be defined. existing one. Creating an annotation requires not only to specify a region of interest and select appropriate concept, but also to instantiate properties defined by the concept. These property values describe the concrete instance represented by the region and can be utilized later in queries.

5 Related Work

Researchers in computational sciences are now facing the problem of sharing, exchanging, and querying highly heterogeneous forms of data generated by experiments and observations in an uniform and transparent fashion. However, because of the heterogeneity of the data and the lack of

common schemas, standard approaches to data integration, e.g., based on multidatabases or federated databases, are often inapplicable or not efficient. Thus, several new approaches focusing on semantic interoperability have been proposed in the past few years.

First, there is an increasing amount of work on models and methodologies to semantically enrich the Web (see `www.semanticweb.org` for an extensive overview). The major focus in these works is on building semantic rich and expressive ontology models that allow users to specify domain knowledge. The most prominent approaches in this area are the Ontobroker project [8, 9], SHOE [15], the Topic Maps standard [30] as general ontology frameworks, and TAMBIS [26] and OIL [27] as specific ontology frameworks tailored to the biological domain. We consider these ontology-centric works as orthogonal to our annotation-centric approach. Furthermore, whereas the above approaches concentrate on querying ontologies using, e.g., RDF-based languages, our focus is to have a simple, expressive, and easy to implement language that allows to query all three components, concepts, annotations, and Web accessible documents in a uniform fashion.

A second class of approaches in this context focuses on information integration by combining query or mediator systems and domain models/ontologies. For example, in the OBSERVER system [20], metadata and ontologies represent knowledge about the vocabularies used in the sources and are utilized to handle heterogeneity for query processing. In the SCOPE system [23], semantic relationships between schema elements of different sources are identified using ontologies and exploited for evaluating queries. The MOMIS approach [3] is based on a shared ontology for defining an integrated view on heterogeneous sources and semantic optimization of queries. The mediator described in [19] uses domain maps representing semantic nets of concepts and relationships and semantic indexes of source data into a domain map. Recent surveys on current works in this area are given, e.g., in [28] and [24]. However, all these approaches require database-like structures and schema information. Thus, they are not applicable to schema-less data such as images or textual documents.

At the other end of the spectrum, several systems have been proposed that provide users with means to annotate data. This includes the multivalent document approach [25], the SLIMPad approach [6], the Annotea project [18] as well as some commercial systems (see, e.g., [11, 16] for an overview). While none of these approaches supports a query framework for annotations, only [6] support the notion of concept like structures underlying annotations.

6 Conclusions and Future Work

In many computational sciences, the association of different types of metadata with heterogeneous and distributed collections of scientific data play a crucial role in order to facilitate data retrieval tasks in an integrated and uniform way. In this paper, we have presented a framework that allows researchers to associate well-defined metadata in form of concept instances with images and text data. This framework comprises a graph-based data model, operations for traversing graphs of concepts and annotations as well as a view mechanism. The model and its realization provide all features researchers in collaborative environments deem necessary to enrich data and thus to "semantically" index and integrate data that is not easy to classify and analyze otherwise.

Furthermore, we have described the mapping of our model to a relational database schema and an appropriate query translation scheme. In this way, we can rely on efficient data management facilities provided by full-featured DBMS without exposing the user to the complexity of pure SQL queries for data retrieval in the graph structure.

While the usage of the first prototype of our system has shown that the generic mapping approach achieves a reasonable performance for moderate-sized graphs, we are currently investigating more sophisticated mappings, e.g., involving materialization of paths for frequently used relationships such as the relationships of the "built-in" types.

Further challenges are the scalability of the system as well as effectiveness of user interfaces. The scalability issue addresses the support of larger communities. A distributed approach using multiple instances of the graph storage and query services could provide this but requires replicas of essential metadata like concepts and partitioning/clustering of annotations and relationships. With respect to the user interface we plan to investigate visual query interfaces exploring the graph structure of the

data.

References

[1] R. Agrawal: Alpha: An Extension of Relational Algebra to Express a Class of Recursive Queries. In *Proc. 1987 ACM SIGMOD International Conference on Management of Data*, 580–590, ACM, 1987.

[2] C. Baru, A. Gupta, B. Ludäscher, R. Marciano, Y. Papakonstantinou, and P. Velikhov. XML-based information mediation with MIX. In *Proc 1999 ACM SIGMOD International Conference on Management of Data*, 597–599, ACM, 1999.

[3] S. Bergamaschi, S. Castano, and M. Vincini. Semantic Integration of Semistructured and Structured Data Sources. *SIGMOD Record*, 28(1):54–59, 1999.

[4] J. Clark, S. DeRose. XML Path Language (XPath) Version 1.0, W3C Recommendation, Nov 1999.

[5] K.E. Campbell, D.E. Oliver, E.H. Shortliffe: The unified medical language system: towards a collaborative approach for solving terminology problems. *JAMIA*, Volume 8, 12–16, 1998.

[6] L.M. Delcambre, D. Maier, S. Bowers, M. Weaver, L. Deng, P. Gorman, J. Ash, M. Lavelle, J. Lyman: Bundles in Captivity: An Application of Superimposed Information. In Proc. of the 17th International Conference on Data Engineering (ICDE 2001), IEEE Computer Society, 111-120, 2001.

[7] Dublin Core Metadata Initiative, dublincore.org/.

[8] S. Decker, M. Erdmann, D. Fensel, R. Studer: Ontobroker: Ontology based Access to Distributed and Semi-Structured Information. In *Database Semantics - Semantic Issues in Multimedia Systems, IFIP TC2/WG2.6 Eighth Working Conference on Database Semantics (DS-8)*, 351–369. Kluwer, 1999.

[9] D. Fensel, J. Angele, S. Decker, M. Erdmann, H.-P. Schnurr, S. Staab, R. Studer, A. Witt. On2broker: Semantic-based access to information sources at the WWW, 1999. In: Proceedings of the World Conference on the WWW and Internet (WebNet 99), 1999.

[10] A. Farquhar, R. Fikes, J. Rice: The Ontolingua Server: A Tool for Collaborative Ontology Construction. Technical Report KSL-96-26, Knowledge Systems Laboratory, Stanford, CA, 1996.

[11] J. Garfunkel: Web Annotation Technologies. look.boston.ma.us/garf/webdev/annote/software.html

[12] H. Garcia-Molina, Y. Papakonstantinou, D. Quass, A. Rajaraman, Y. Sagiv, J. Ullman, V. Vassalos, and J. Widom. The tsimmis approach to mediation: Data models and languages. *Journal of Intelligent Information Systems*, 8(2):117–132, March 1997.

[13] T. R. Gruber. Toward Principles for the Design of Ontologies user for Knowledge Sharing. *International Journal on Human-Computer Studies (1993)*.

[14] M. Gertz, K. Sattler, F. Gorin, M. Hogarth, J. Stone: Annotating Scientific Images: A Concept-based Approach. To appear in 14th Int. Conference on Scientific and Statistical Databases.

[15] J. Heflin, J. Hendler: Dynamic Ontologies on the Web. In *Proc. of the 17th National Conference on Artificial Intelligence (AAAI 2000)*, 443–449, AAAI/MIT Press, 2000.

[16] R.M. Heck, S.M. Luebke, C.H. Obermark: A Survey of Web Annotation Systems, www.math.grin.edu/∼luebke/Research/Summer1999/survey_paper.html

[17] S. Koslow, M. Huerta (eds.): *Neuroinformatics: An Overview of the Human Brain Project*. Lawrence Erlbaum Associates, NJ, 1997.

[18] J. Kahan, M.-R. Koivunen, E. P. Hommeaux, R. R. Swick: Annotea: An Open RDF Infrastructure for Shared Web nnotations. In *Proc. 10th International World Wide Web Conference (WWW10)*, 623–632, ACM, 2001.

[19] B. Ludäscher, A. Gupta, and M. Martone. Model-Based Mediation with Domain Maps. In *Proc. of the 17th Int. Conf. on Data Engineering, April 2-6, 2001, Heidelberg, Germany*, pages 81–90, 2001.

[20] E. Mena, V. Kashyap, A. Illarramendi, and A. Sheth. Domain Specific Ontologies for Semantic Information Brokering on the Global Information Infrastructure. In *International Conference on Formal Ontologies in Information Systems (FOIS'98), Trento (Italy)*, pages 269–283, 1998.

[21] braininfo.rprc.washington.edu/mainmenu.html, Neuroscience Division, Regional Primate Research Center, University of Washington.

[22] Neuroinformatics – The Human Brain Project. www.nimh.nih.gov/neuroinformatics/index.cfm.

[23] A. Ouksel and C. Naiman. Coordinating Context Building in Heterogeneous Information Systems. *Journal of Intelligent Information Systems*, 3(2):151–183, 1994.

[24] A. Ouksel and A. Sheth. Semantic Interoperability in Global Information Systems: A Brief Introduction to the Research Area and the Special Section. *SIGMOD Record*, 28(1):5–12, 1999.

[25] T. A. Phelps, R. Wilensky: Multivalent Annotations. In *Research and Advanced Technology for Digital Libraries – First European Conference*, 287–303, LNCS 1324, Springer, 1997.

[26] R. Stevens, P. Baker, S. Bechhofer, G. Ng, A. Jacoby, N.W. Paton, C.A. Goble, A. Brass: TAMBIS: Transparent Access to Multiple Bioinformatics Information Sources. *Bioinformatics* 16(2):184–186, 2000.

[27] R. Stevens, C. Goble, I. Harrocks, S. Bechhofer: Building a Bioinformatics Ontology using OIL. To appear in a special issue of IEEE Information Technology in Biomedicine on Bioinformatics, 2001.

[28] A. P. Sheth: *Changing Focus on Interoperability in Information Systems: From System, Syntax, Structure to Semantics*. Kluwer Academic Press, 1999.

[29] B. Swartout, R. Patil, K. Knight, T. Russ. Toward Distributed Use of Large-Scale Ontologies, 1996. In *Proc. 10th Knowledge Acquisition for Knowledge-Based Systems Workshop*, Alberta, Canada, 1996.

[30] Topic Maps. www.topicmaps.org

[31] UC Davis/UC San Diego Human Brain Project Informatics of the Human and Monkey Brain, neuroscience.ucdavis.edu/HBP.