Ontogator: Combining View- and Ontology-Based Search with Semantic Browsing

Eero Hyvönen, Samppa Saarela, and Kim Viljanen

Helsinki Institute for Information Technology (HIIT) / University of Helsinki
P.O. Box 26, 00014 UNIV. OF HELSINKI, FINLAND
{Eero.Hyvonen, Samppa.Saarela, Kim.Viljanen}@cs.Helsinki.FI
http://www.cs.helsinki.fi/group/seco/

In: Proceedings of XML Finland 2003, Open Standards, XML, and the Public Sector, Kuopio, October 30-31, 2003.

Abstract. We show how the benefits of the view-based search method, developed within the information retrieval community, can be combined and extended with the benefits of ontology-based annotations and search, developed within the Semantic Web community. As a proof of the concept, we have implemented an ontology- and view-based image retrieval and recommendation browser Ontogator. Ontogator is innovative in two ways. Firstly, the RDFS-based ontologies used for annotating image metadata are used in the end user interface to facilitate view-based image retrieval. The views provide the user with a search function and means for getting useful overviews of the contents in the repository. Secondly, a semantic browsing function is provided by a recommender system. This system enriches instance level image metadata by the ontology and provides the user with links to semantically related relevant images. The notion of a "semantically relevant link" is specified in terms of logical rules. To illustrate and discuss the ideas, a practical application of Ontogator to a photo repository of the Helsinki University Museum is presented.

1 Introduction

There are two major approaches to image information retrieval. In content-based image retrieval (CBIR) [3] the images are retrieved based on their characteristics, such as color, texture, shape, etc. In the metadata-based approach to be discussed in this paper, image retrieval is based on descriptions of the images.

To retrieve images from a database, keyword-based query systems [1, 2] are typically used. Here the user may select filtering values or apply keywords to the different database fields, such as the "creator", "time", or to the content descriptions including classifications and free text documentation. More complex queries can be formulated, e.g., by using Boolean logic.

Keyword annotations and keyword-based information retrieval systems are widely used but have several problems related 1) to the quality of search results and 2) to the usage of systems.

1.1 Answer Quality Problems

The precision and recall of keyword-based search methods is lowered due to many reasons [4]. For example, a keyword in a document does not necessarily mean that the document is relevant, and relevant documents may not contain the explicit word. Synonyms lower recall rate, homonyms lower precision rate, and semantic relations such as hyponymy, meronymy, and antonymy [6] are not taken into account of.

A prominent solution approach to problem is to use ontology-based annotations and information retrieval [16, 17].

1.2 Usability Problems

The standard keyword search methods are not always easy to use [9]:

Formulating the information need Keyword search implicitly assumes that the user has a goal in mind, i.e., to find a set of images with desired characteristics. However, in many applications this is not the case. One may, for example, want to learn about some topic and only have a general interest of finding images related to a vague topic.

Formulating the query The user is often faced with a repository of images whose content is semantically complicated and the domain more or less unknown. In such situations it is difficult to to determine what keywords to use in formulating the query.

Formulating the result set The answer format used in a keyword-based search system is typically a list of hits, the *result set*, ordered according to their expected relevance to the user. Explanations on why different hits are included the result, and/or their semantical grouping would be helpful in analyzing the results, too.

A solution approach to address the usability issues above is the *multi-faceted* or *view-based* search method[1] [14, 8]. Here the idea is to organize the terminological keywords of the underlying database into orthogonal hierarchies and use them extensively in the user interface in helping the user to formulate the queries, in navigating the database, and in grouping the results semantically.

The traditional notion of hit lists and view-based search misses an important aspect of the repository content: the *relations* by which the hit items (images) are related with each other and other images in the database. This relational information should in many cases be a part of the answer as, for example, recommendations. For example, if the query contains the keyword “Sibelius”, and the result set contains an image depicting Jean Sibelius, the Finnish composer of symphonies inspired by the Carelian scenery (a part of Finland), then a relation to images of Carelia (not in the actual result set) could be of interest to the user.

This paper describes a system called Ontogator. Its main novelty lays in the idea of enhancing keyword search accuracy and usability by combining ontology-based knowledge representation with the view-based search method. Furthermore, the notion

[1] See http://www.view-based-search.com for a historical review of the idea.

of view-based searching is complemented with the idea of semantic browsing used in Topic Maps [12] and recommender systems [15].

We describe Ontogator by using a concrete application case in which the system has been developed. However, the idea of Ontogator is not bound to any particular application domain. A goal of the work is to develop a generic semantic image browser for image repositories, whose contents are annotated by associating each images with a set of semantic RDF(S) resources describing its content.

In the following, keyword, view-based and ontology-based approaches to image retrieval are first discussed. After this the Ontogator system is presented and its underlying ideas are discussed. In conclusion, the lessons learned and contributions of this work are summarized.

2 Semantic Image Annotation and Retrieval

2.1 Keyword-Based Annotation and Retrieval Schemes

The content of images in a database are typically described by associating each image with a set of keywords that describe its content. The keywords can be either explicitly user-given or be determined automatically from free text and other descriptions of the images.

Keywords are often selected from controlled vocabularies or thesauri [7] in order to maintain annotations mutually coherent and to ease image retrieval. Since controlled thesauri are not complete and new keyword terms emerge, also to use of uncontrolled vocabularies is often necessary.

Additional expressive power to keyword annotations can be obtained by hierarchical thesauri and classification systems, such as the Art and Architecture Thesaurus [13] or ICONCLASS[2] [18]. They classify different aspects of life into hierarchical categories. A category is described by its name and a set of additional keywords. When an image is annotated by a category *C*, the image automatically inherits the keywords of *C* and its super-categories. For example, since "castle" is a subcategory of "building", an image annotated with the keyword "castle" is found using the keyword "building". The same idea of enlarging a keyword with related terms in order enhance recall is used in thesaurus- and concept-based query expansion techniques [10].

2.2 View-Based Search Method

The thesaurus can be constructed in a systematic fashion by a set of semantically orthogonal hierarchies such as "Time", "Place", etc. that are often called *facets* or *views*. The facets provide complementary views of the contents along different dimensions.

The facets can be used for indexing the content and to help the user during information retrieval [14]. Firstly, the hierarchies give the user an overview of what kind of information there is in the repository. Secondly, the hierarchies can guide the user in creating the query and in selecting appropriate keywords that are likely to lead to non-empty answer sets — a recurring problem in IR systems. Thirdly, the hierarchies can

[2] http://www.iconclass.nl

be used to disambiguate query terms. Fourthly, the facets can be used as a navigational aid when browsing the database content [8].

The idea in multi-facet search is that the user makes a selection of categories of interest from different facets, and the system then constructs the corresponding query. If the categories selected are $C_1, ..., C_n$ and the subcategories of $C_i, i = 1...n$, including itself are $S_{i,1}, S_{i,2}..., S_{i,k}$, respectively, then this corresponds to the following boolean AND-OR-query:

$$(S_{1,1} \vee ... \vee S_{1,k}) \wedge (S_{2,1} \vee ... \vee S_{2,l}) \wedge ... \wedge (S_{n,1} \vee ... \vee S_{n,m}) \quad (1)$$

An additional idea of the multi-facet search is to compute proactively the number of hits in every direct subcategory of $C_i, i = 1...n$ and show it to the user. In this way, the user can be hindered from making a selection leading to empty result set and be guided toward such selections that are likely to constrain or relax the search appropriately.

This idea was used, e.g., in the HiBrowse system [14] in the 90's. A later application of the multi-facet approach is the Flamenco system [8], the first web-based proto of the Ontogator system [9] and its current version described in this paper. In Flamenco, the next level of subcategories for the selected categories are exposed to user. View-based search is adapted into a navigational browsing-searching scheme for the Web. The query is constrained further during the navigation flow by clicking subcategory links from the next direct subcategory level below, or by relaxing formerly selected constraints. After each step, next selection possibilities and the sizes of the corresponding result sets are computed. Extensive user studies [11, 5] have recently been carried out to show that a direct Google-like keyword search interface preferred if the users know precisely what they want. However, if this is not the case, then the multi-faceted search method with its "browsing the shelves" sensation is clearly preferred over keyword search or using only a single facet.

In Ontogator, the search interface is based on the HiBrowse model. However, the whole hierarchy, not only the next level of subcategories, can be opened for selections. Moving between hierarchy levels is more flexible because at any point any new selection in the hierarchy opened is possible. The "browsing the shelves" sensation is provided by a separate recommendation system based in the underlying ontological domain knowledge. This provides a semantically richer basis for browsing than the keyword hierarchies used in Flamenco.

2.3 Ontology-Based Annotation and Search

If view-based search is based on keywords then the search method does not solve the answer quality problem of keyword-based search, which is due to the fact that a set of keyword cannot describe accurately the contents of images. For more accurate descriptions, semantically richer ontology-based annotations can be employed [16, 17]. Such annotations are not atomic keywords but can be more detailed structured descriptions that are linked with other resources in a semantic graph. With the help of ontologies, the user can also express the queries more precisely and unambiguously, which leads to better precision and recall rates. Furthermore, through ontological class definitions and inference mechanisms, such as property inheritance, instance-level metadata can be

automatically enriched and used for determining implicit semantic relationships in the database. The price to be paid is that much more work is needed when constructing the ontologies and during the content annotation phase. Using more complex and detailed ontological structures in the user interface may also make the system less understandable to the end-user.

3 Ontogator Approach

The key idea of the Ontogator system is to combine the usage benefits of multi-facet search with the answer quality benefits of ontology-based search, together with semantic recommendations. To describe the system, we use the promotion ceremony image database of the Helsinki University Museum[3] as a case study. Promotion ceremonies consist of several academic occasions and parties and last for several days. The database contains 629 photographs about the ceremony events and documents, ranging from the 17th to 21th century, and more images are acquired after every promotion. The problem is to provide the museum guest with a sensational information retrieval system for investigating the contents of this semantically complicated image database, i.e., to illustrate the inner life and traditions of the university.

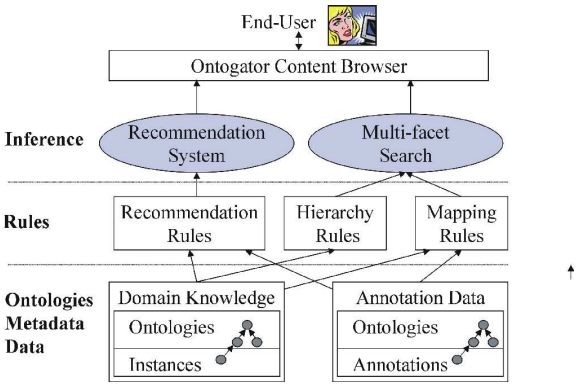

Fig. 1. Architecture of Ontogator.

Figure 1 depicts the overall architecture of Ontogator. The system is used by the Content Browser and is based on two information sources: Domain Knowledge and Annotation Data.

Domain Knowledge consists of an ontology that defines the domain concepts and the individuals. In our case, the domain ontology consists of some 329 promotion-related concepts, such as “Person” and “Building”, 125 properties, and 2890 instances, such as “Linus Torvalds” and the “Entrance of Cathedral of Helsinki”.

[3] http://www.helsinki.fi/museo/

Annotation Data describes the metadata of the images represented in terms of the annotation and domain ontologies. Annotation ontology describes the metadata structure used in describing the images. It is assumed, that the subject of an image is described by associating the image with a set of RDF(S) resources of the domain knowledge, classes or instances. They occur in the image and hence characterize its content. The difference with simple keyword annotations is that the associated resources are part of the domain ontology knowledge base, which disambiguates the meaning of annotations (synonym/homonym problem) and provides additional implicit semantic metadata. The annotations also include other metadata, such as the photographer, free text descriptions and some technical information of the images, but in the promotion case application only the subject descriptions are used for multi-facet search and recommendations.

Based on the domain knowledge and the annotation data, Ontogator provides the user with two services:

Multi-facet search The underlying domain ontologies are mapped into facets and facilitate multi-facet search. In our example case, there are six facets "Happenings", "Promotions", "Performances", "Persons and roles", "Physical objects", and "Places". The facets provide different views into the promotion concepts and data and are used by the user to focus the information need and to formulate the queries, as described earlier.

Recommendation system After finding an image of interest by multi-facet search, the domain ontology model together with image annotation data can used to recommend the user to view other related images. The recommendations are based on the semantic relations between the selected image and other images in the repository. Such images are not necessarily included in the answer set of the multi-facet search query. For example, images of the next and previous events in the promotion ceremonies can be recommended or images of the relatives of a person in the figure. The recommendation system makes it possible to browse the contents using semantic navigation.

The two services are connected with the information sources by tree sets of configurations or rules.

Hierarchy rules The hearth of the multi-facet search engine is a set of category hierarchies by which the user expresses the queries. The hierarchy rules are a set of configurational rules that tell how to construct the facet hierarchies from the domain ontologies.

Mapping rules Annotations associate each image with a set of resources of the domain ontology. Mapping rules extend these direct annotations by describing the indirect relations between the images and the domain knowledge. For example, a direct image annotation may tell that a particular person, say Linus Torvalds, is in an image. However, the person may be in different images in different roles, e.g., as a "Promovend" or as a "Doctor Honoris Causa". Roles are a useful facet to the user of the system. An image in which Linus Torvals is present in the role of Doctor Honoris Causa should therefore be annotated with this role in order to distinguish

the image from other images of him. However, then the image is not annotated directly as an image of Linus Torvals and the view-based search would not find the image as an image of the person. Mapping rules solve the problem by specifying the relations by which images are related with domain resources.

Recommendation rules The domain ontology defines not only the concepts and their hierarchical structure, but also the relations by which the actual domain classes and individuals are related with each other. Based on these relations, recommendation rules are used to find associations between an image and other images of potential interest to the user. The recommendations are defined in terms of logical Horn clauses. For example, "Related Person" -rule may link a person with another persons through family relations. If the user selects an image exposing a person p, then images exposing persons in different family relations with p can be recommended to the user.

4 View-Based Search Method

The view-based search makes faceted hierarchical categories used to describe the image content visible to the user. Hierarchies may represent, for example, places, happenings, time, roles and persons. Figure 2 shows the search interface of the Ontogator prototype. The query is formulated and executed by selecting resources of interest from the facet hierarchies. When the user makes a selection, the system retrieves the images that are related to the selected resource. When several resources are selected from different views, the result is the intersection of the images related to these resources (cf. section 2.2). After each selection the result set is recomputed. For each resource in the opened hierarchies, a number ratio n/k is counted and shown. It tells that if the resource is selected, then there will be n images in the result set out of the k images related to that resource totally. A selection leading to empty result set ($n = 0$) is disabled and shown in gray color.

The inner nodes of the hierarchies are associated with their sub-nodes by the search system. If the facet hierarchy is projected using `rdfs:subClassOf` and `rdfs:type` relations, and C is the selected class, then the system retrieves images that are either related to the class C directly or to any of its subclasses or instances.

The underlying hierarchies can also be used to formulating the results of a query. For example, Ontogator constructs descriptions of images by listing the resources of the selected categories that actually appear in a picture and shows them beside the thumbnail pictures. Another possibility would be to group the results according to the subcategories of a user-selectable facet [8].

4.1 Hierarchy Rules

Hierarchy rules define the root categories of the facets and how the facet hierarchies are projected starting from these. Hierarchy rules are needed in order to make the classifications shown to the end user independent from the design choices of the underlying Domain Ontologies. The view-based search system itself does not differentiate between differently projected hierarchies.

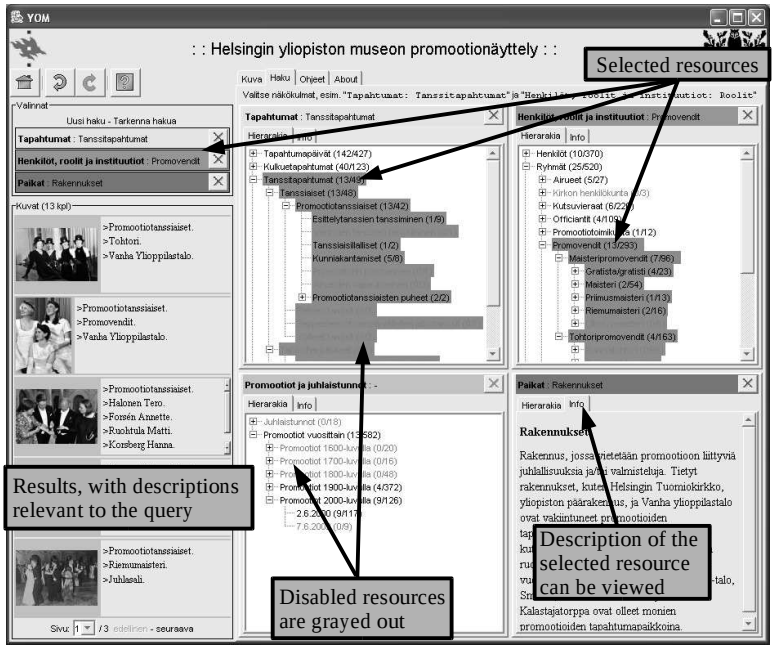

Fig. 2. Ontogator user interface for view-based search.

An obvious way to extract a facet hierarchy from the RDF(S)-based domain knowledge is to use the subclass-of hyponymy relation. Then the inner nodes of the hierarchy consist of the classes of the domain ontology, and the leaves are the direct instances of these classes. Using only hyponymy for facet projections would, however, be a limitation in the general case. For example, places may constitute a meronymical part-of hierarchy, and this would a natural choice for a facet in the user interface.

If the hierarchies intersect each other, then a resource selection should be considered in the context of the hierarchy. For example, Helsinki may be viewed as an instance of the class City, a legal body, or as a part of Finland. Choosing Helsinki from the part-of hierarchy should match pictures of squares, beaches and other places that are situated in Helsinki, but it would be confusing, if pictures of beaches were returned by a query where Helsinki is selected in the sense of a legal body.

The idea of viewing an RDF(S) knowledge base along different hierarchical projections has been applied, e.g., in the ontology editor Protégé-2000[4] that allows to choose the property by which the hierarchy of classes shown to the user is projected. When using the default hierarchy constructed by the `subClassOf`-relation, the root of the hierarchy is always the top class (”Thing” in Protégé-2000, ”Resource” in RDF Schema). When using some other relation, a problem is how to determine the root since, for example, there may be cycles in the RDF graph. In Protégé-2000 the selected class becomes

[4] http://protege.stanford.edu

the root of the hierarchy. Protégé-2000 also has a general option ”References” that uses any instance-valued property to construct the hierarchy. This idea could be applied in Ontogator as well. However, in many cases a more precise specification of the projection than a single property is needed. For example, the hyponymy projection already employs two properties (`rdfs:subClassOf` and `rdf:type`). Furthermore, the ordering of the sub-resources may be relevant. In our case, for example, the sub-happenings of an event should be presented in the order in which they take place. In the Ontogator prototype, the projections are created by special purpose Java methods implementing the hierarchy rules, and only hyponymy projections were used. Ordering of the sub-nodes can be specified by a configurable property.

Hierarchy rules tell how the hierarchies used in the view-based search are projected. A closely related but still separate question is how these hierarchies should be shown to the user. For example, in our case study, the ontology was created partly before the actual annotation work and had more classes and details than were actually needed. The projected Objects facet hierarchy, for instance, had many classes of decorations not related to any picture. A hierarchy may also have intermediate classes that may be useful for knowledge representation purposes but are not very natural categories to the end user. One needs to be able to filter unnecessary resources away from the user interface, yet they should be present internally in the search hierarchies. In our work, configurational filtering rules for showing the facets have been investigated but not yet implemented in the system.

4.2 Mapping Rules

An image is annotated by associating it with a set of domain knowledge resources. This set is, however, not enough because there may be complex indirect relations between images and resources describing its content. Mapping rules are used to specify what indirect resources describe the images in addition to the direct ones. Through such rules it is possible to achieve a search system that is independent of both the annotation scheme and the domain ontology design. The search system itself does not make any distinction between the ways in which resources and images may be related.

For example, there are the classes Role and Person in the domain ontology of promotions. The subclasses of Role, such as Master and Doctor Honoris Causa, are used to indicate the role in which some person may appear in a picture. If the role r of a person p appearing in a picture is known, then the picture is annotated by an instance of the Role. As a result, the picture is found using r in the multi-facet search, but not with p, which is unsatisfactory. The system should be able to infer that the images, that are about an instance of Role are also images about the person in that role.

A similar kind of situation occurs with the concept of promotion. All instances of the class Promotion are related to a particular university, faculty, and the conferrer of degrees of the promotion. The system should be able to infer that pictures related to some promotion are also related to the university and faculty of that particular promotion happening. However, in contrast, the conferrer of degrees is related to the image only if actually appearing in it. This kind of distinctions are highly domain dependent and difficult to find out automatically without explicit additional information, such as

mapping rules. The mappings between annotations and resources of domain knowledge can be quite complex.

In Ontogator, mapping rules are given as RDQL[5] query templates. The templates are applied for hierarchy resources *r*, classes and instances, by first instantiating them with the URI of *r*. The resulting RDQL query is applied to the RDF(S) knowledge base consisting of the Domain Ontology and the Annotation Data. The result is a set of image resources related to *r*. All mappings between facet resources and the images are determined when constructing the system's inner representation of the facet hierarchies. This strategy of computing mappings during the startup makes the search system faster but at the price of the memory needed for the search data structures. In the future versions of Ontogator, we intent to develop a dynamic version that uses the underlying RDF representation directly through a general inference engine.

The applicability of a mapping rule is usually limited only to a certain subtree of the facet hierarchy. Furthermore, if facet hierarchies intersect, then the mapping rules may be facet-dependent.

5 Rule-Based Recommendations

In addition to the multi-facet search mechanism, Ontogator also has a rule-based recommendation utility. It allows the user to browse semantically related images in the spirit of Topic Maps [12]. However, while the links in a Topic Map are given by the map, the links in Ontogator are inferred based on rules and the underlying knowledge base.

Figure 3 illustrates the recommender system. The Query overview lists the selected facet categories and Query results the result set. The Recommendations for the Selected picture are grouped on the right based on three rules: one for determining pictures depicting related persons (family relations), one for determining pictures of the preceding event, and one for determining pictures of the following event.

An RDF(S) knowledge base contains relations between the resources described in the ontologies and the metadata. Not all of the relations and resources may be of interest of the user, and it may also be undesirable to show all information. (For example, the limited monitor size of a mobile device constrains the amount of information to be shown.) Furthermore, a related resource may not be interesting in itself, but only as a mediating resource for another resource. In such cases, the user would be interested in knowing directly the indirect resource behind the mediating resource.

Recommendation rules are needed for selecting the most interesting relations from the wealth of relations between resources in the knowledge base. The relations may be either direct property links or indirect ones. Through recommendation relations the end user gets a more fluent, semantically guided browsing experience between resources interest.

[5] http://www.hpl.hp.com/semweb/rdql.htm

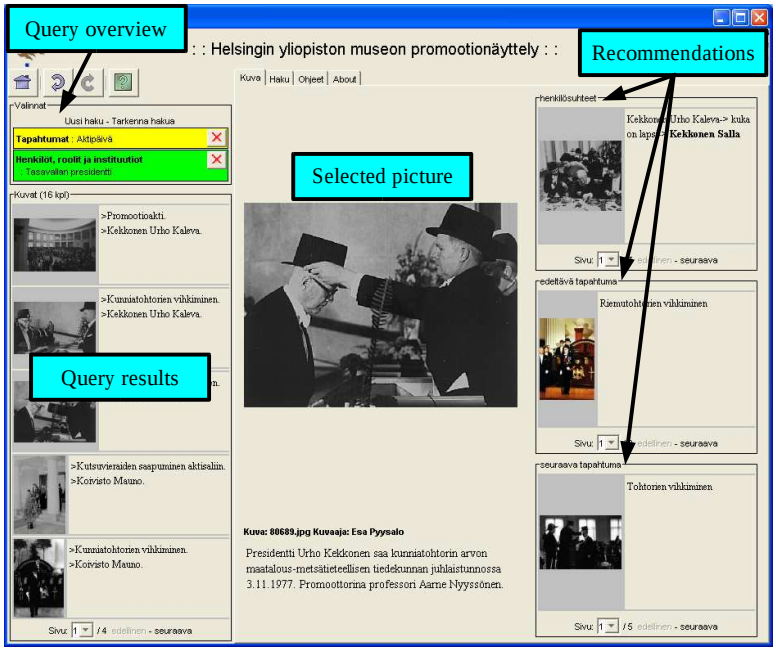

Fig. 3. Screenshot of the recommendation system.

5.1 Alternatives for Recommendations

Recommendations can be created in various ways [15]. In our case we have been considering following three alternatives: user profile-based, similarity-based, and rule-based recommendations.

User profile-based recommendations are based on information collected by observing the user, or in some cases by asking the user to explicitly define the interest profile. Based on the user's profile, recommendations are then made to the user either by comparing the user's profile to other users' profiles (collaborative filtering/recommending) or by comparing the user's profile to the underlying document collection (content-based recommending).

The strength of user profile-based recommendations is that they are personalized and hence serving better the user's individual goals. In our case application, personalization is however difficult, because the users cannot be identified. It is not even known when the user's session begins and when it ends because the users are using the same physical kiosk interface located in the museum. The profiling must be easy for the user because most of the users use the system perhaps only once in their lifetime. Finally, it is difficult to identify whether the user liked or disliked the current image without asking the user to rate every image explicitly. A weakness could be also that explaining the recommendations to the user can be difficult, because they are based on heuristic

measures of the similarity between user profiles and database contents, and of the user's actions.

With similarity-based recommendations we refer to the possibility to compare the semantical distance between the metadata of a selected image and the other images. The nearest images are likely to be of more interest and could be recommended to the user. A difficulty of this recommendation method is how to measure the semantical distance between metadata. In many cases the most similar image is not the most interesting one but rather just another picture of the same event. A simple method is to use the count of common or intersecting annotation resources as a distance measure, but there are lots of other choices based on the ontology available.

The idea of rule-based recommendations used in Ontogator is that the domain specialist explicitly describes the notion of "interesting related image" with generic rules. The system then applies the rules to the underlying knowledge base in order to find interesting images related to the selected one. This method has several strengths. Firstly, the rule can be associated with a label, such as "Images of the previous event", that can be used as the explanation for the recommendations found. It is also possible to deduce the explanation label as a side effect of applying the rule. Recommendation rules are described by the domain specialist. The rules and explanations are explicitly defined, not based on heuristic measures, which could be difficult to understand and motivate. Secondly, the specialist knows the domain and may promote the most important relations between the images. However, this could also be a weakness if the user's goals and the specialists thoughts about what is important do not match, and the user is not interested in the recommendations. Thirdly, the rule-based recommendations do not exclude the possibility of using other recommendation methods. For example, the recommendation rules could perhaps be learned by observing the users actions and then used in recommending images for the current or future users.

In the first web-based version of Ontogator [9], we implemented profile-based and similarity-based recommendation system that recommended more semantically similar images. The recommendation were not static but were modified dynamically by maintaining a user profile and a history log of image selections. Then a rule-based recommendation system was implemented due to the benefits discussed above. The idea was tested in the promotion application that currently contains 1) rules for recommending images of related persons, such as children, parents, wife, husband, etc., and 2) rules for determining images of the next and previous events based on the general promotion programme used in (almost) all promotions.

5.2 Implementation

The current version of the rule-based recommender is implemented in SWI-Prolog[6]. For reasons of efficiency, the recommendations are determined in a batch process before using the application. The recommender reads the metadata and ontologies in RDF(S) format into the Prolog interpreter. The semantic relations between the images are then determined by the rules. Finally, the program produces a XML-file containing the recommendations which is loaded into the Ontogator browser at the next startup of the

[6] SWI-Prolog version 5.1.5, http://www.swi-prolog.org/

browser. A limitation of this static approach is that it is not possible to create on-line dynamic recommendations based on the user's profile and usage of the system.

The Prolog rules are divided in three groups: domain specific recommendation rules, system specific rules (the "main" program creating the recommendations) and RDF Schema specific rules (such as rules implementing the transitive subclass-of closure). The domain specific rules are created by the domain specialist. System specific rules and RDF Schema specific rules are domain independent and need to modifies only if the system or the RDF Schema specification changes.

When processing the data, the program iterates through all images and their metadata. The recommendation rules are applied to every different resource *r* in the images' metadata (a URI reference). If recommendations are found, they are stored as recommendations for *r*. The recommendations are created for each metadata resource *r* and not for each image in order to minimize the size of the XML-file. The Ontogator browser then shows as the recommendations of an image the recommendations related to each resource *r* used in the image's metadata.

The recommendations contain the recommended resource (URI), a natural language explanation for the relation between the resources, and a category for the recommendation (e.g., "related persons"). In the current implementation, the natural language descriptions are relatively simple such as "Person A -> a child of -> Person B". The texts are based on the labels of the resources defined in the RDF descriptions. Also the reverse relation is described, which would be in the previous example "Person B -> a parent of -> Person A". This facilitates symmetric usage of the recommendation associations.

In the promotion application, the recommendation system is working as planned and creates recommendations shown to the user. The recommendations extend Ontogator usage from just searching images to browsing between related images. However, evaluating the quality and relevance of recommendations can only be based on the user's opinions. In our case, only a small informal a user study of has been conducted using the personnel of the museum. The general conclusion was that the idea seems useful in practice.

Logic programming seems to be a very flexible and effective way to handle RDF(S) data by querying and inferring when compared with RDF query languages, such as RDQL and RQL. The definition of the recommendation rules requires programming skills and may be difficult to a domain specialists who is not familiar with logic languages. Computational efficiency and central memory requirements can be a problem if the RDF knowledge base is large and if the rules are complex.

In the domain model of the promotion application, the properties did not constitute hierarchies. However, property hierarchies would make it easier to create recommendation rules in situations where a recommendation rule could be described using the top level property and be "inherited" to its sub-properties. For example, if there is a property "human-relation", the sub-properties could include "spouseOf", "parentOf", and "childOf". The definition of a "human-relation" recommendation rule could then be done using the "human-relation" property and be (automatically) available also, e.g., for determining the persons with the spouseOf relation.

Observing the end users using the system could give valuable additional information about what recommendations are mostly used and the flow of viewing the images. The value of such studies is, however, limited by fact that that the ontologies and current recommendations limit the possibilities of the user to select any image from the collection as the next image.

6 Conclusions

This paper developed a method for combining the benefits of RDF(S)-based knowledge representation, the view-based search method, and knowledge-based recommendations. An implementation prototype, Ontogator, and an application of it to a practical image retrieval problem was discussed. Ontologies and facet hierarchies can be used to help the user in formulating the information need and the corresponding query. Furthermore, the ontology-enriched knowledge base of image metadata can be applied to constructing a more meaningful answer to a query than a simple hit-list. The recommender system can provide the user with a semantic browsing facility between semantically related images.

The integration of the view-based and ontology-based search paradigms turned out to be more complicated than expected. The main difficulty is how to model and deal with the indirect relations between the images and domain ontology resources, and how to project the facet hierarchies from the RDF knowledge base. If not properly modeled the precision and recall rates of the system are lowered.

A reason for choosing RDF(S) is its domain independent nature an opennes. This makes it possible to apply Ontogator more easily to other image repositories by reconfigurations. During our work, we actually reused the promotion ontology and instance data easily in another application.

Defining new logical recommendation rules on top of the RDF-triple format is flexible, but required a fair amount of expertise concerning the underlying ontological structures and Prolog programming. A problem encountered there is how to test and verify that the recommendations for all images are feasible without having to browse through the whole database.

During our work, Protégé-2000 was used as the ontology editor. Jena's[7] basic main memory -based model (ModelMem) was employed to load the RDFS-models into Ontogator's internal representation form. Protégé turned out to be a versatile tool with an intuitive user interface that even for a non-programmer could use for constructing ontologies (from the technical viewpoint). A good thing about Protégé is that it is not limited to RDFS semantics only, but enables and enforces the use of additional features.

Ontology evolution poses a problem with Protege even in the simple case that a name (label) of some class changes. Protégé derives URI's of the classes from their names, and if a name changes then the classes URI (ID) changes also. This leads to configurational problems. Rules and mappings for one version of the ontology do not apply to the new version, even though the actual classes have not changed, only their labels. Multi-instantiations would have been desirable in some situations but this is not possible with Protégé.

[7] http://www.hpl.hp.com/semweb/jena.htm

The major difficulty in the ontology-based approach is the extra work needed in creating the ontology and the detailed annotations. We believe, however, that in many applications — such as in our case problem — this price is justified due to the better accuracy obtained in information retrieval and to the new semantic browsing facilities offered to the end-user. The trade-off between annotation work and quality of information retrieval can be balanced by using less detailed ontologies and annotations, if needed.

7 Acknowledgements

Kati Heinämies and Jaana Tegelberg of the Helsinki University Museum and Avril Styrman provided us with the actual case database, the ontology, and annotated the images. Our work was partly funded by the National Technology Agency Tekes, Nokia, TietoEnator, the Espoo City Museum, the Foundation of the Helsinki University Museum, the National Board of Antiquities, and the Antikvaria-group.

References

1. M. Agosti and A. Smeaton, editors. *Information retrieval and hypertext*. Kluwer, New York, 1996.
2. R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval*. Addison-Wesley, New York, 1999.
3. J. P. Eakins. Automatic image content retrieval —are we getting anywhere? pages 123–135. De Montfort University, May 1996.
4. D. Fensel (ed.). The semantic web and its languages. *IEEE Intelligence Systems*, Nov/Dec 2000.
5. J. English, M. Hearst, R. Sinha, K. Swearingen, and K.-P. Lee. Flexible search and navigation using faceted metadata. Technical report, University of Berkeley, School of Information Management and Systems, 2003. Submitted for publication.
6. C. Fellbaum, editor. *WordNet. An electronic lexical database.* The MIT Press, Cambridge, Massachusetts, 2001.
7. D. J. Foskett. Thesaurus. In *Encyclopaedia of Library and Information Science, Volume 30*, pages 416–462. Marcel Dekker, New York, 1980.
8. M. Hearst, A. Elliott, J. English, R. Sinha, K. Swearingen, , and K.-P. Lee. Finding the flow in weg site search. *CACM*, 45(9):42–49, 2002.
9. E. Hyv¨onen, S. Kettula, V. Raatikka, S. Saarela, and Kim Viljanen. Semantic interoperability on the web. Case Finnish Museums Online. Number 2002-03 in HIIT Publications, pages 41–53. Helsinki Institute for Information Technology (HIIT), Helsinki, Finland, 2002. http://www.hiit.fi.
10. K. J¨arvelin, J. Kek¨al¨ainen, and T. Niemi. ExpansionTool: Concept-based query expansion and construction. *Information Retrieval*, 4(3/4):231–255, 2001.
11. K.-P. Lee, K. Swearingen, K. Li, and M. Hearst. Faceted metadata for image search and browsing. Proceedings of CHI 2003, April 5-10, Fort Lauderdale, USA. Association for Computing Machinery (ACM), USA, 2003.
12. Steve Pepper. The TAO of Topic Maps. In *Proceedings of XML Europe 2000, Paris, France*, 2000. http://www.ontopia.net/topicmaps/materials/rdf.html.
13. T. Peterson. Introduction to the Art and Architecture Thesaurus, 1994. http://shiva.pub.getty.edu.

14. A. S. Pollitt. The key role of classification and indexing in view-based searching. Technical report, University of Huddersfield, UK, 1998. http://www.ifla.org/IV/ifla63/63polst.pdf.
15. J. Ben Schafer, Joseph A. Konstan, and John Riedl. E-commerce recommendation applications. *Data Mining and Knowledge Discovery*, 5(1/2):115–153, 2001.
16. A. T. Schreiber, B. Dubbeldam, J. Wielemaker, and B. J. Wielinga. Ontology-based photo annotation. *IEEE Intelligent Systems*, 16:66–74, May/June 2001.
17. G. Schreiber, I. Blok, D. Carlier, W. van Gent, J. Hokstam, and U. Roos. A mini-experiment in semantic annotation. In I. Horrocks and J. Hendler, editors, *The Semantic Web – ISWC 2002. First international semantic web conference*, number LNCS 2342, pages 404–408. Springer–Verlag, Berlin, 2002.
18. J. van den Berg. Subject retrieval in pictorial information systems. In *Proceedings of the 18th international congress of historical sciences, Montreal, Canada*, pages 21–29, 1995. http://www.iconclass.nl/texts/history05.html.