# MEDIALIFE: From Images to a Life Chronicle

Amarnath Gupta
Univ. of California Irvine
9500 Gilman Drive
La Jolla, CA 92093, USA
gupta@sdsc.edu

Setareh Rafatirad
Dept. of Computer Science
Univ. of California Irvine
Irvine, USA, CA 92697
srafatirad@gmail.com

Mingyan Gao
Dept. of Computer Science
Univ. of California Irvine
Irvine, USA, CA 92697
gaomingyan@gmail.com

Ramesh Jain
Dept. of Computer Science
Univ. of California Irvine
Irvine, USA, CA 92697
jain@ics.uci.edu

## Categories and Subject Descriptors

H.2.8 [**Database Applications**]: Image Databases; I.2.4 [**Knowledge Representation Formalisms and Methods**]: Semantic Networks

## General Terms

Algorithms

## Keywords

multimedia, annotation, ontology, graph queries

## 1. INTRODUCTION

This demonstration addresses the question "How should personal media data be organized and inter-related to make a personal media management system more useful to casual users?" Today, thanks to affordable digital cameras, mobile phones, webcams and so forth, the need to creatw, store, organize and share people's personal media has turned into a significant market. Naturally, this need is reflected in the software word. It is estimated that as of September 2008, 47 photo-services API, 40 video-services API and over a hundred media composition APIs have been published for writing applications that enable ordinary users[1] maintain and share digital memoirs of their personal lives through media objects.

On their online manifesto, Flickr, Yahoo's photo sharing site (`http://www.flickr.com`) states its two main goals: 1) "to help people make their content available to the people who matter to them", and 2) "to enable new ways of organizing photos and video". Naturally, there is a significant effort to ensure that stored media is searchable. Many media management applications enable automated and user-assisted means to annotate the media data so that the annotations can be used to organize and search media. All these applications allow the user to create her own directory structure to organize media objects. Most systems extract the EXIF metadata that automatically capture the date, time and imaging parameters like the focal length, which can then be used for various purposes like ordering pictures by time. With Flickr and Picasa (`http://picasa.google.com`), a user can tag (i.e., associate a set of keywords with) a picture or a part of a picture. In the same vein, in August 2008, Picasa has enabled face recognition on stored images; as the system automatically determines the number of persons in the picture, the user creates an association between a names from an address book to faces recognized by the system. Similarly, a user may utilize a map interface to store the location where the photo was taken, and add textual annotations like a caption or simply comments. All of these annotations together help a user to search for pictures by using an API, or formulating a conjunctive query over the structured and unstructured annotation.

There is no doubt these photo management systems have succeeded in implementing the infrastructure for storing and sharing media data, their metadata and annotations. But, have they really succeeded in becoming the record keepers of life's moments, and becoming a chronicle of life? Let us consider a user with the following photo-finding requests.
*Create an album of 20 photos of my son's graduation party, ordered by time.*
*Find photos of my family members who visited us last year.*

Today's photo management systems lack the support such *life chronicle queries* due to a number of challenges.

1. Keyword search is often not be very effective when photos are not rigorously tagged. Even if a photo belongs to the graduation party event, it may not be labeled as such.

2. The system may not associate the concept "family member" with individuals in the pictures even though the pictures may be tagged with the names/faces of people in them.

3. The criteria for selection and ranking of pictures is not obvious. If the graduation party has over a hundred pictures, choosing twenty from them is an underspecified problem.

4. The requests need a minimal degree of event interpretation. For example, pictures taken when family

[1]We emphasize ordinary users as distinct from photographers and other professionals who do not capture media as witnesses of their life events.

*SIGMOD'09,* June 29–July 2, 2009, Providence, Rhode Island, USA.
ACM 978-1-60558-551-2/09/06.

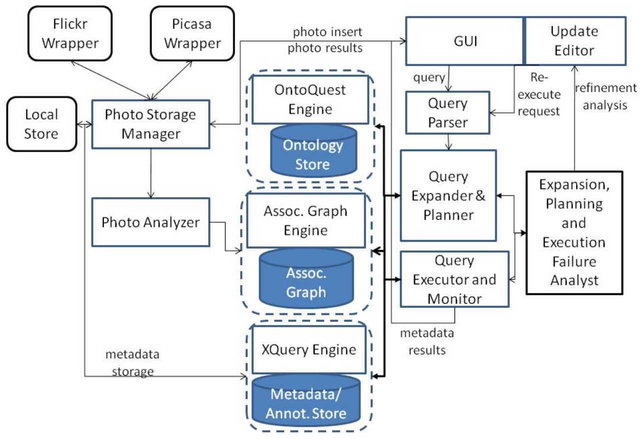

**Figure 1: The medialife system architecture.**

members are visiting us should be distinguished from pictures when we are visiting family members.

We believe that the fundamental shortcoming of current systems comes from their inability to capture the semantics of the media. *Our demonstration shows the implementation of a system where the philosophy is to marry asset management with semantic information, so that together they can create a digital chronicle of life.*

To achieve this goal, media storage and sharing system needs to be significantly augmented with a personal information system [3] that uses a basic model of people's life events. MEDIALIFE, the system we propose to demonstrate addresses these concerns by building on top of one or more existing photo management systems like Flickr or Picasa, (also over a local store if desired), but uses a *model-based approach*. MEDIALIFE uses a **personalized world model** based upon a **personal ontology** to create and incrementally update an **information-association-graph** over the collection of pictures and annotations. Users' photo requests are handled by performing a **query analysis** step, followed by a **query reformulation** step, both of which are based on the personalized world model of the individual user. An ambiguous query is resolved by asking the user to define the unknown part of the query in terms of the ontology and the association graph. The disambiguating response triggers a **model update** (also called *knowledge refinement*) process that refines the ontology, and updates the association graph, and then the query is re-evaluated.

## 2. SYSTEM COMPONENTS

The system architecture is shown in Figure 1. **Storage.** The MEDIALIFE system is created as a web application and uses existing (Flickr, Picasa) systems and their API for the storage and retrieval. The system accepts an upload request and internally uses the appropriate API to create a virtual album (with sub-albums if chosen by the user) over these systems.

**Metadata and Media Annotations.** The MEDIALIFE system uses a suite of image feature extraction plug-ins to compute different characteristics of the images uploaded. Some plug-in modules, such as face recognition are computed by external systems like Picasa. Other modules, like the "outdoor images with people" finder have been developed by us. These modules use a combination of automatically extracted metadata (e.g., EXIF parameters) and user-provided metadata (e.g., location). The metadata and media annotations are represented as XML data served by an XQuery engine. The semistructured representation is essential to ensure flexibility of the information a user may wish to capture. The XML schema used admits temporal and spatial data types in addition to the data types permitted by the XML Schema standard. For example, the location where a picture was taken may be represented as:

```
<location>
   <loc:relative:room>41
     <loc:relative:floor>3
       <loc:relative:wing>west
         <loc:absolute:building>Hoover
          <gml:Point>
       <gml:pos>37.38911780598221
                    -122.08638668060303</gml:pos>
      </gml:Point>
          ...
</>
```

where the element names representing relative location (like `loc:relative:room`) come from a place ontology defined in the system.

**Ontology Management.** We have previously demonstrated OntoQuest, our ontology management system [2] and its applications [4]. Specifically for MEDIALIFE, we have developed an upper-level ontology for events based on BFO (Basic Formal Ontology) and DOLCE, an extension of the PIM ontology (`http://dev.nepomuk.semanticdesktop.org/wiki/PimoOntology`) for personal information and relationships, and a media ontology. To handle these OWL-based ontologies, the original OntoQuest system has now been extended perform graph-like operations on *relative relationships*, different forms of locational and temporal information (e.g., named time intervals, relative time, relative values), event-subevent structures, natural and human-participated events, discrete, transient and smoothly transitioning events, events that occur due to changes in properties of objects and spatial entities and so forth.

**Association Graph Management.** The association graph captures the concrete relationships between instances of ontological classes as well as standard data types. Association graphs are node and edge-label directed graphs, created by several semi-automated processes we will demonstrate. A simple graph query engine is used to provide subgraph retrieval and graph navigation operations. This engine is similar to a SPARQL engine, but is simpler in the sense that it does not permit all constructs of the SPARQL language, yet at the same time it permits reachability operations over edge-labeled acyclic graphs that SPARQL does not permit. The engine internally uses acyclic graph index structures [1] for all transitive, irreflexive relationships, and interoperates with OntoQuest to retrieve class-level properties specified in the ontology (see an example in the next section).

**Mapping Relations.** The map relations are like join indices and represent the relationships between nodes of the association graph and the media objects and their annotations. Let's a photo has the annotation that "David Wilcox" is present in the photograph, and the association graph has the information that "David Wilcox" is the primary user's sister's husband, the mapping relation materializes the join between the two references to David Wilcox.

**Query Engine.** The MEDIALIFE system uses a simple query language where the query is conjunctive with only atomic negations, and the result of a query is always a set of photos. The system supports incremental, interactive queries, where the user may use the result of the last query to formulate the next query. A typical use of this is to find "related pictures" where the axes of relationships relevant for a specific picture is given by the association graph that connects the metadata of the current picture to that of other potentially relevant pictures. A salient feature of the query planner is its ability to determine when the query cannot be answered directly from the metadata and should be interpreted by consulting the association graph and the ontology. The query executor maintains an account of the successful and failed predicates so that the failing predicate(s) can be sent to a failure analysis module to check if the failure is due to lack of query interpretation or for lack of data.

## 3. DEMONSTRATION

We will demonstrate the system on a collection of over 10000 personal photographs. For the demonstration, the viewers will see the interfaces for three different tasks.

**The Edit Interface.** The edit interface is used to upload a photo, together with its metadata and annotations, or edit information of an existing photo. This interface will showcase the utilization of the ontology in the annotation process, and the interaction between the two. When a photo is uploaded, a number of feature extraction and classification processes are executed and all automatically extracted information is brought to the interface. If a user does not agree with a classification result, the system uses the ontology to suggest alternate classes. The addition of user-provided metadata, tagging and annotation are deeply guided by the ontology. Thus the entry of any metadata can be direct (just filling in a value in a form), or it can be assisted, where the system initiates a dialog that fills in the metadata with increasing specificity. For example, if the user trying to fill in the location of a picture enters "hotel room", the system will use its ontological knowledge-base to recognize it as a relative location, and will attempt to "ground" it by asking more details about the hotel. If however, a previously uploaded photo from the same event(heuristically determined from the time-stamp) already has the hotel information, this enquiry will be skipped.

**The Personal Knowledge Management Interface.** This interface is used for defining, searching and editing the ontology and the association graphs in the system. This interface has two components – one for the application developer and a second for the end user.

*App. Developer's Knowledge Interface:* The system comes preloaded with the upper ontology and some common application ontologies. However, for any specific application, it is necessary to add either a complete new ontology that references the upper ontology or an extension that "hangs from" the current elements of an existing application ontology. For example, one may extend an existing "travel ontology" with a subgraph detailing water travel. The demonstration will show a wiki-based ontology interface that the user can simply update to add domain knowledge to the ontology. Two kinds of updates are allowed – most updates result in updating the ontology, and the association graph. Some updates are expressed as non-recursive conjunctive rules; these are transformed into views on top of the stored relations managed by OntoQuest.

*User's Knowledge Interface:* The user's update of the knowledge base mostly involves the association graphs. These graphs are initially formed by 1) extracting information from relationships expressed in Facebook accounts and Outlook / GMail contacts of a user, 2) analyzing tags of people (e.g., using face recognition where possible) and their co-occurrence in pictures. The initial relationship assignment is performed through a set of rules (i.e., views), that we will show as part of the presentation. The user then refines these relationships – thus two people with an initial annotation of "acquaintance of" may get refined to a more specific relationship like "student of".

**The Query and Refinement Interface.** The system is queried through a simple form-based interface. The demonstration will first show the query at a user-level. We will also present a "console view" to describe the structure of the query object, and how it gets progressively expanded by consulting with the ontology. The primary feature to showcase will be the analysis of a failed query. We will walk a viewer through cases where 1) a query completely fails, and the query analysis suggests a "suspect clause" that is presented back to the user for further definition, 2) a query succeeds and yet the query analysis determines that the results are incomplete due to an "under-defined clause" that needs further knowledge refinement. In both cases, the refinement process will performed by showing the appropriate part of the ontology or application graph on the Personal Knowledge Management Interface for update. After the knowledge refinement, the query will be automatically re-executed to produce improved query results. We will also showcase some variants of result ranking based on event density.

## 4. CONCLUSION

In this demonstration paper, our goal is to show that a photo management system when supplemented by (1) semantic information such as an individual's life-events and social relationships, and (2) the machinery to process this semantic information significantly improves "life chronicle queries" of casual users.

## 5. REFERENCES

[1] Chen, L., Gupta, A., and Kurul, M. E. Stack-based algorithms for pattern matching on dags. In *Proc. 31st Int. Conf. on Very Large Databases (VLDB), Stockholm* (2005), pp. 493–504.

[2] Chen, L., Martone, M. E., Gupta, A., Fong, L., and Wong-Barnum, M. Ontoquest: Exploring ontological data made easy. In *Proc. 31st Int. Conf. on Very Large Databases (VLDB)* (2006), pp. 1183–1186.

[3] Czerwinski, M., Gage, D. W., Gemmell, J., Marshall, C. C., Manuel A. Pérez-Qui n., Skeels, M. M., and Catarci, T. Digital memories in an era of ubiquitous computing and abundant storage. *Commun. ACM 49*, 1 (2006), 44–50. Spl. Issue, *Personal Info. Mgmnt.*

[4] Gupta, A., Gupta, S., and Condit, C. Graphitti: An annotation management system for heterogeneous objects (demo track). In *Proc. IEEE Conf. Data Engineering (ICDE), Cancun* (2008), pp. 1568–1571.