Department of
Information Engineering
and Computer Science
DISI

UNIVERSITY
OF TRENTO - Italy

DISI - Via Sommarive 14 - 38123 Povo - Trento (Italy)
http://www.disi.unitn.it

LIFELOG EVENT MANAGEMENT: CROWD RESEARCH CASE STUDY

Pil Ho Kim and Fausto Giunchiglia

August 2011

Technical Report # DISI-11-473

Accepted for publication to Joint Workshop on Modeling and Representing Events J-MRE 2011 (EiMM'11 / SRED'11) at ACM MM 2011.

Lifelog Event Management: Crowd Research Case Study

Pil Ho Kim
Department of Information Science and Computer Engineering
University of Trento
TN, 38123, Italy
pilho.kim@disi.unitn.it

Fausto Giunchiglia
Department of Information Science and Computer Engineering
University of Trento
TN, 38123, Italy
fausto@dit.unitn.it

ABSTRACT

Lifelog research involves complex event analysis that requires the efforts of many advanced experts in different disciplines. It also handles privacy and/or copyrighted materials where mutual consent on the use of data is very important. Thus we propose a secure Web-based lifelog database service to help researchers concentrate efforts to solve difficult research issues. This paper details many aspects of lifelog research topics and technical challenges that the author has experienced over one year case study. It also describes economical and efficient methods to implement the back-end system and services for real lifelog event management.

Categories and Subject Descriptors

D.2.11 [**Software Engineering**]: Software Architectures—*Domain-specific architectures*; E.2 [**Data**]: Data Storage Representations—*Composite structures, Linked representations*; H.4 [**Information Systems Applications**]: Miscellaneous

General Terms

Design, Experimentation

Keywords

Event, Information Management, Lifelog, Real-life logging

1. INTRODUCTION

Our life is full of events. However objective data collected using a variety of sensors gives much different subjective feelings to a user by her experiences with that event. This gap between objective data and subjective feelings is filled up with a myriad of relations that have been built upon by the time when a user is exposed to the data, which can be thus evolving with the time.

A data logging activity recording and tracing one's personal real-life events is called a personal lifelogging. Its application domains include but not limited to personal/enterprise information management, health-care system and surveillance for the public and for the military. Recent digitized and developed technologies have dramatically changed the way we store information about ourselves. Now lifelogging activities are broadly linked with our daily lives seamlessly and transparently processing our emails, photographs, geospaital location or health records archiving them back to the cloud storage. Lifelog research is an effort to analyze such real lifelogs to create informative and digitized stories of person's life, while necessary technologies and handling media types significantly vary by the application domain.

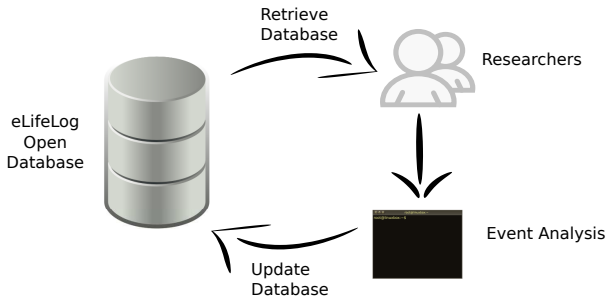
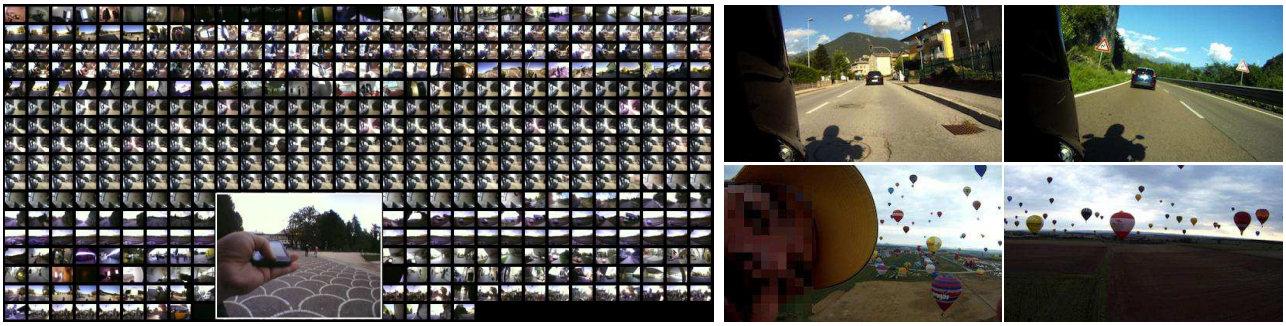


Figure 1: Lifelog research collaboration work flow.

With the fast progress of mobile computing, personal lifelogging gets easier taking a little effort for automation, data recalling services to help a user locate or remind the event in her lifelogs still lacks supports in major aspects due to the difficulty in creating a useful environment for users to take advantage of. Though its necessity is very high, technical challenges in building up a useful lifelogging services lay on the fact that lifelogging involves sophisticated technologies from the step gathering and archiving data, analyzing its features, extracting patterns and to providing back a service for users that we can get meaningful experiences without much burdens in using the system.

Prior lifelogging activities have been more focused on total capture [6, 7, 5] where the back-end system and its architecture have been proprietorially implemented. High-level lifelog event analysis using such commonly shared knowledge on human activities however needs mutually agreed knowledge database and the developer system to build up and merge analysis results is very necessary. In the other aspect, Sellen et. al [12] argued that rather than trying to capture everything, lifelog system design should focus on the psychological basis of human memory. The connection in between the psychological memory and the objective digi-

J-MRE'11, November 30, 2011, Scottsdale, Arizona, USA.



(a) (b)

Figure 2: Personal lifelogging cases: (a) Daily activities captured using ViconRevue and (b) Outdoor events using GoPro.

tized data needs intensive event analysis to link micro-events from low-level data events to high-level physiological events. Thus this work is focused on finding a systematic method to support the research progress in finding a way to manage diverse events to match up with human knowledge. Additional features, or challenging research topics, of lifelog events are : (1) They involve a personal history that highly depends on one's prior experiences; (2) source data objects are disparate both in terms of properties and structures; and (3) modeling of personal activity events needs to merge significant amounts of common knowledge on human behaviors which also needs intensive reasoning process to link events analyzed from her lifelogs.

The problem is that these issues can not be addressed by a single research team. Thus essential prerequisites we recognized from a year of case study is the urgent need to build a collaborative research environment to promote lifelog research activities. We first launched a web site[1] for a case study to identify real challenges through the process in utilizing, circulating and enhancing event analysis results from the collaboration (See Figure 1).

Following sections address and introduce our solutions during the step to achieve above proposed goals. Section 2 introduces the characteristic of lifelog data and our approaches to implement lifelog back-end system. Section 4 describes our experiences in building the services for the lifelog research expert group. It includes our year-long lifelogging practices, UI development and ground truth data generation. Finally, Section 5 concludes this paper with the objectives of our activities.

2. MODELING LIFELOG DATA AND ITS ARCHIVE DESIGN

Lifelog inherently involves many heterogeneous data objects where many data object types and structures are not pre-determined when setting up the initial data back-end system. Its collection mostly gets complex as more archives and their various analysis results are added on. This significantly raises up the complexity, for instance, in the query composition for information retrieval. However blind training methods to automatically structure relations [11] between such complex data objects often lacks the context of events without which recognized events do not make a sense for a real-life. Rather we think a comprehensive back-end system that facilitates a user or a system to store complex data and to navigate through complex network is more necessary to find clues for information retrieval based on the generalized understanding on events of interests. To make problems and research goals concrete, let us review a number of representative scenarios that frequently happen in lifelog research works.

2.1 Complex Data Analysis Case

Let us start with a relatively simple personal lifelogging case in which a person captures her daily activities using her smart phone and hand-held camera by which she tracks her locations, takes photos and movies, and exchanges emails and SNS messages with others. We assume that all such activities are fairly well time-stamped and uploaded at the cloud server (See lifelogs in Figure 2.) While a given lifelogging environment looks simple, actual lifelog data often needs very powerful data analysis techniques to retrieve events of user's interests.

For instance in image processing, many photos are taken under various circumstances where lighting conditions and/or motion stability varies significantly by cases. Considering the fact that most image processing algorithms get affected by such environmental conditions, no one algorithm can act on all cases. Even an object detecting algorithm well-trained to detect one or a number of event types often needs a different run-time configuration per environmental changes. Or it needs prior works to balance and control the quality of images before processing to keep the algorithm performance reliable. Therefore we often apply multiple algorithms in practice to detect same event types from data. Though their outputs might be disparate in terms of data structures or in values, they semantically indicate very correlated events. What we need is then a mechanism to blend and assimilate outputs to draw semantically matching tags to merge event analysis results. In fact, this image processing case is not limited to that media but happens on most other lifelog media types and their data analysis technologies.

2.2 Inherited Data Analysis Case

In addition to the above one image data case, research works on diverse types of lifelog data result in many heterogeneous data where the way to merge and process could be very complex. For an example in geo-spatial data, original satellite GPS data can be further analyzed to compute person's moving direction, speed and paths. Moreover moving speed can be semantically categorized to indicate user's ac-

[1]https://www.elifelog.org

tivity status as standing, walking, running or whether she was riding a vehicle. Reversed geo-coded street addresses can also be tagged by a user as, for instance, her home address, work address or the restaurant where she had a dinner with a number of friends.

All above data are stemmed from, or whether in partially related with, the original GPS data though data structures (or called schemas) of descendant data objects are all different in the view of database object management. Its back-end should take this into the account or else data silos would be soon filled up with too diverse data objects in different structures raising the complexity in managing the data objects. Also this issue actually affects almost all aspects of lifelogs from data logging, analyzing, and proving a service back to a user. Though this problem has been a vivid challenging issue in lifelog research fields, it has been mostly handled using proprietary storage solutions without addressing a generalized event model for lifelogs.

2.3 Federated Data Grouping Case

The above problem gets more complex when merging multiple federated personal lifelogs to create the group of lifelogs. Each one may have used different types of sensors and analyzing their interactions would need a significant work even to bring them onto the table for comparison. In addition, for instance, ownerships of events, complex structural analysis on social relations and tracking physical events shared by members are all such challenging topics to the back-end database system.

3. PROPOSED HYBRID EVENT MODEL: E-MODEL

From the above observation, we can deduce a number of database design criteria to model real life events[2]:

M-1 A structured data model works well for sensor stream data, which are mostly structured and well-sampled whether in tabular format or in tree format, due to its efficiency in predetermined structured data parsing and the benefits from the established legacy database system.

M-2 However data assimilation to expand search over others lifelogs or other types of similar results of one's own logs needs a graph-type database back-end that supports a variety of value examination functions on complex relations for navigation. This is where the connection between data objects are more important in many cases.

M-3 In terms of user experiences, services for user interaction at the step of data query is essential not even to compensate the lack of details of collected events but also to expedite user's reminiscence process with better understanding on the trace of events. During this user interaction, system's capability to prepare the cache to hit for fast response or for selection suggestion is a great measurer in enhancing user experiences.

For the application like lifelog, which needs a very abstract and generalized data structure to accommodate and manage streams of disparate data sources, a graph data model fits by theory with its rich supports for object inheritance, abstraction (or abstract nodes) and multiple relations (or labeled edges) good for representing complex information networks. However putting all data into such a network-type graph is not practical and it causes the difficulty even from the first step for data logging, which violates our data model design criteria M-1. Thus to cross over the big gap between M-1 and M-2, we designed a new hybrid data model (named E-model), in which various data models can co-exist in rich relations.

[2]This section is the extension of [9] with more details on the E-model database system for interoperation with existing data models and their instances.

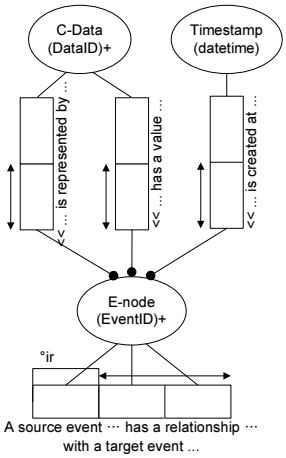


Figure 3: The E-model triplet structure.

The idea of E-model is to allow the insertion of new data instances in their original structured form and also the access limited to those instances to go through its original query mechanism. However at the moment of insertion, new data instances are also translated into the E-model graph so that it can co-exist with other instances in different structure to be used for graph navigation.

The conceptual E-model structure is depicted in Figure 3. Unlike legacy triple stores or recent NoSQL back-ends [13], a number of distinctions we made are: (1) The use of c-data (composite data type) objects for both the name (or called the key) and the value of data, which is devised based on the observation that when writing a query, data names are often objects to query in handling complex data structures; (2) It is also useful in supporting an imprecise human query not only on data values but also on data names (or keys); (3) By the system design, it allows a significantly fast blind search on both keys and values using only a part of data. This is often a critical system performance measurer in the search engine to boost up the query speed of user's initial blind (i.e. unstructured) query; (4) The concept of the super e-node from which all group e-nodes inherit represents the ownership of events keeping its origin even when sharing c-data objects with other users e-node graph.

To support various data structures, E-model also has the concept of the schema as shown in Figure 4. Schema objects for the E-model are composed of the fundamental elements required for application domain data models. In E-model, an e-node is an ordered 3-tuple object composed of key, value, and timestamp. Because the timestamp of an e-node is the transaction time that will be automatically

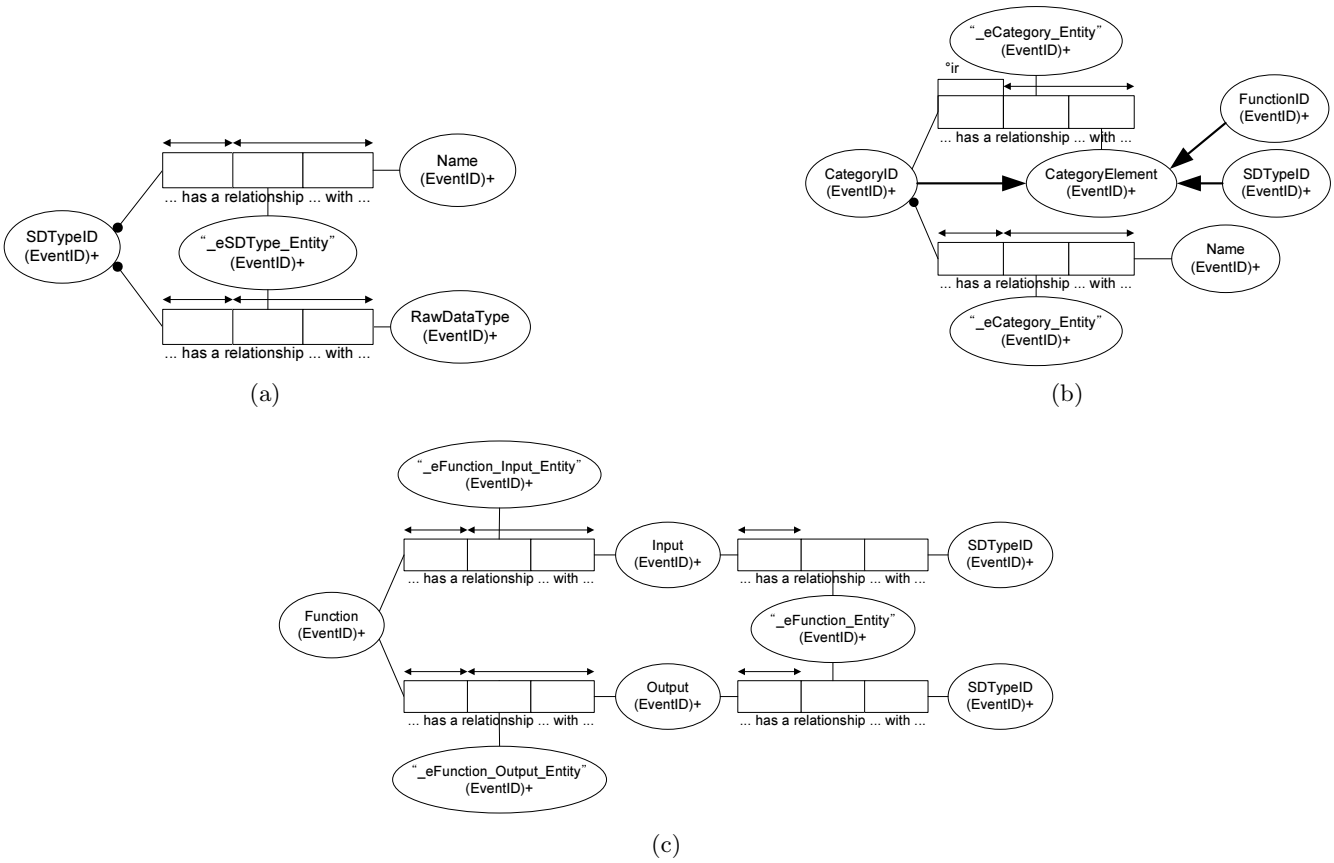


Figure 4: E-model schema: (a) E-model SD type schema; (b) E-model function schema; and (c) E-model category schema.

logged when an e-node is registered, the key and the value of an e-node constitute the specification for an e-node. Therefore in defining the structure of an e-node, its key and raw value data type are elements of specification. In this view, an E-model symbol and data type pair or e-SD type, e_{sd}, is a type definition of an e-node. An e-SD type is a set of two predicates: one e-node $\Phi_\delta(e_{sd})$stores its key and the other e-node $\Phi_\tau(e_{sd})$ that stores its raw data type.

E-model uses the concept of the function in place of the relation to distinguish directed relations from source nodes to target nodes. An e-node relation introduced here is a labeled directed edge between two e-nodes. Thus, the schema of the e-node relation has source and target e_{sd} objects. Its name represents the semantic relation between two e-nodes. An E-model function or e-function, e_f, consists of a domain (source) e-sdtype X and a codomain (target) e-SD type Y with constraints on $x \xrightarrow{e_f} f(x)$, where $x \in X$ and $f(x) \in Y$ and both are e-sdtypes.

A category of the data model is (1) a collection of objects, (2) a collection of functions between objects, and (3) child category objects to support inheritance. E-model supports a similar concept with which aforementioned E-model objects are associated. An e-category, e_c, is a grouping schema object that constitutes a set of structured data objects. An E-model category consists of three elements: (1) e_{sd} as its basic data objects, (2) e_f to specify relations between data objects, and (3) child e_c sets for inheritance. An instance of the e_c schema object is an e-group node. Its child e-nodes have structured relations with an e-group node. The network composed of e-group nodes is a directed acyclic graph (DAG) strictly following the temporal order of event occurrence.

Our system is further extended to best utilize the power of human reminiscence with the supports for semantic tagging on c-data objects (i.e. semantic tags for both data names and values.)[3] For instance, when retrieving the job title of a user, the column name of a job title node could be various like position, title, job or role. In E-model, such query on data can be simply formulated just like data value constraints using the same or extended APIs without proprietary tweaks for add-on semantic search supports. In addition, all E-model schema objects are also the group of e-nodes which means that external redundant data object to manage data schema is not necessary in the E-model system and the schema evolution is the addition of new schema-type e-nodes to existing graph.

From above conceptual designs and rules on the schema, an E-model database prototype is built atop existing database systems. By theory, any database system supporting first-order labeled relations can work as the back-end for E-model. However, we chose a relational database in our early implementation as our back-end to get benefits from optimized indexing and processing methods and for the maximal

[3]Currently WordNet 3.0 [4] is adopted for semantic tagging.

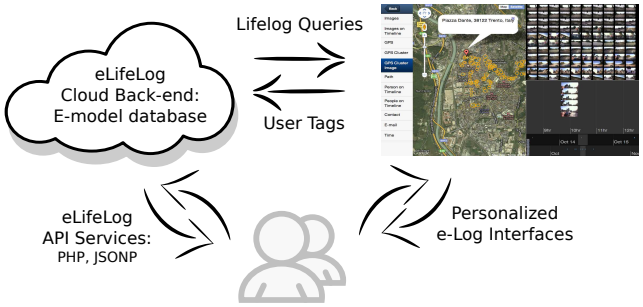


Figure 5: eLifeLog research collaboration service flow.

interoperation with existing RDBMS data instances. The first E-model codes are written in pure SQL with hundreds of stored procedures and functions to maximize its portability with existing relational databases and most recently we released a PHP version of E-model source codes to extend the support for external programming languages to facilitate complex programming and the use of external libraries. Technical details of our implementation and its promising benchmark results are introduced in detail at [8] and source codes both in SQL and PHP versions are freely opened for the community[4].

4. SERVICES FOR CROWD RESEARCH

Lifelog is a research domain with a wide variety of application domains that needs efforts of researchers from across all computing disciplines to ensure successful outcomes. To facilitate their collaboration, Web services are now the great way for researchers to use and share group research results. The overall service structure of adopted in our case study is depicted in Figure 5.

4.1 Web Service Implementation

The E-model database we developed in Section 3 is configured to run as a cloud database [10, 1] for researchers to have them access the database at anywhere and at anytime when connected to the internet. A cloud database also saves much burdens in data safety, consistency and maintenance, which are left to the cloud service provider. However the cons are also evident that there is a privacy concern and also the database server configurations may not be adjustable per user's intents.

Our intermediate solutions are first not directly putting private data (ex. image) into the database but placing them at where direct web connection is not allowed. We designed the back-end management system that any access to the private data must pass through the validation process that authenticate user's identity and her level of access. Data access APIs also designed not to use server-side database functions that most database service providers prohibit to prevent server-halting infinite loop attacks.

We also have archived initial lifelog data for crowd research experiments. Due to lifelog data's strong nature of privacy concerns, the best for the lifelog research community is having their own real-life research data which are mutually granted for community members. This case study has been lasted over 1 year from July 2010 to present and the author has archived long-term personal lifelogs using the mix of sensors including ViconRevue, GoPro 3D camera, iPhone and Garmin GPS. Many members also kindly contributed their own daily and/or outgoing lifelogs including G. C. de Silva's one year life logs traveling around the world [3, 2] taken in various situations. More details on our web service implementation can be found at [9].

[4]https://www.elifelog.org/book/e-model-database

4.2 Software As A Service

Considering the fact that members for crowd lifelog research have all different technical backgrounds, unified server-side services for data processing is an essential feature to create the centralized event archive and to manage the nexus of their complex relations analyzed in various aspects by many researchers. Here a web service can act as middleware to integrate distributed software systems contributed by members converting their software as the service (SaaS) [15, 14] for the community. Such a framework should allow them easily integrate their computing resources and software services to be registered as the API service for the community. To achieve the above goal, we are working on the back-end database and distributed cloud computing services that features encapsulating both automated and man-powered crowd services as one virtually unified service using the concept of check-in and -out data with versioning and authoring supports.

4.3 User Interaction and Interface Design

Due to the huge amount of lifelogs, we need effective retrieval methods when looking back into the logs. However some distinct aspect in lifelog system implementation compared with a legacy information systems is that it should strategically support the weakness of human memory (i.e. uncertainties in remembering exact times and/or locations for information retrieval.) Thus we aimed to develop lifelog UIs to put a user into the query loop to maximally utilize her reminiscing abilities for achieve the best experience.

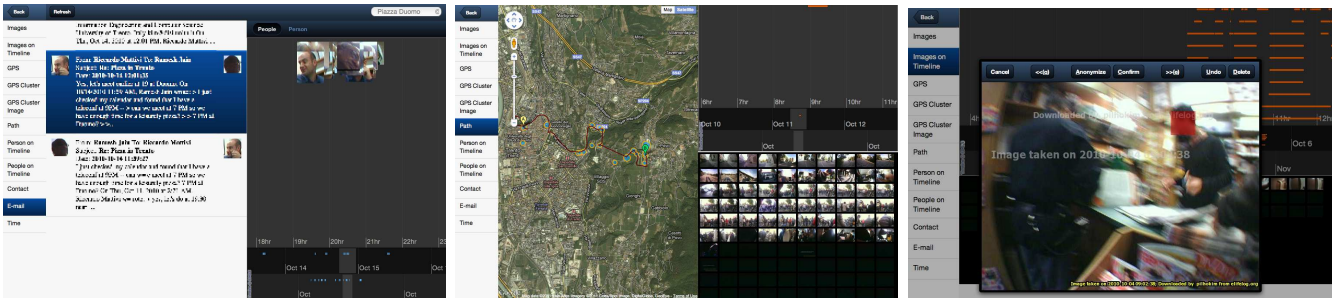
At the first stage, we focused on developing UI APIs for the community member so they can instantly build up UIs to visualize their lifelogs through the Web. For an example, a researcher directly inserts data into her own database table or upload through our Web APIs, they can instantly visualize the data using our distinguished UI services. We found from the case study that this UI support removes a big burden from researchers wasted for GUI implementation.

Followings show a number of selected UIs under services. Figure 6a is a service that a user can search her emails with a keyword. It shows the timeline with a list of people she met from the moment she exchanged the selected email. This is intuitive for users on the case searching over events happened after emailing. Our event GUI APIs are modulated to be integrable with other types of event viewers while reacting to user interactions or data instance changing triggers. Figure 6b shows one such integrated interface example that visualizes events by space and time. More demo videos are available at our site[5].

4.4 Building Ground Truth Data

Pivotal to many tasks in relation to lifelog research and its related scientific algorithm development is the availability of a sufficiently large data set and its corresponding ground

[5]https://www.elifelog.org/article/preliminary-e-log-analysis-result-demo-video



(a) (b) (c)

Figure 6: e-Log user interfaces: (a) Keyword event search interface over the timeline: Find an event with a specific location name, (b) Spatio-temporal event viewer composed of UI service modules: Find GPS tracks (Commuting events in this case) between two places and review images taken then; (c) Web-based crowd sourcing interface to build up the ground truth data: Human classification case is shown in the figure.

truth data for quantized and/or qualified algorithm evaluations. For this, we have started building up the ground truth corpus using the web-based crowd working environment to create the data set with the collaboration from members around the world. Figure 6c shows one of our current prototype UIs developed to work in such a web-based crowd collaboration environment for human classification.

5. CONCLUSION

This paper intents to promote the lifelog research collaboration using most developed web service environments and technologies. It details many aspects of lifelog research topics and technical challenges that the author has experienced over one year to build the system. eLifeLog.org is the result of our efforts that is under operation for many members around the world. As far as we know, it is the first Web-based research supporting service opened for the lifelog research community. Archives and services provided here are for diverse research communities to promote the research progress in modeling and analyzing complex real lifelog events.

6. ACKNOWLEDGMENT

This work is partially supported by the grant from the European Union 7th research framework programme Marie Curie Action - COFUND with the Provincia Autonoma di Trento in Italy and by the European Commission under contract FP7-248984 GLOCAL.

7. REFERENCES

[1] F. Chang, J. Dean, S. Ghemawat, W. Hsieh, D. Wallach, M. Burrows, T. Chandra, A. Fikes, and R. Gruber. Bigtable: A distributed storage system for structured data. *ACM Transactions on Computer Systems (TOCS)*, 26(2):1–26, 2008.

[2] G. de Silva and K. Aizawa. Retrieving multimedia travel stories using location data and spatial queries. In *Proceedings of the seventeen ACM international conference on Multimedia*, pages 785–788. ACM, 2009.

[3] G. De Silva, T. Yamasaki, and K. Aizawa. Sketch-based spatial queries for retrieving human locomotion patterns from continuously archived gps data. *Multimedia, IEEE Transactions on*, 11(7):1240–1253, 2009.

[4] C. D. Fellbaum. *WordNet: An Electronic Lexical Database*. MIT Press, 1998.

[5] A. Fitzgibbon and E. Reiter. Memories for life: Managing information over a human lifetime. *UK Computing Research Committee Grand Challenge proposal*, 22:13–16, 2003.

[6] J. Gemmell and G. Bell. Mylifebits: fulfilling the memex vision. In *Tenth ACM International Conference on Multimedia*, pages 235–238, Juan-les-Pins, France, 2002.

[7] V. Kalnikaite, A. Sellen, S. Whittaker, and D. Kirk. Now let me see where i was: understanding how lifelogs mediate memory. In *Proceedings of the 28th international conference on Human factors in computing systems*, pages 2045–2054. ACM, 2010.

[8] P. H. Kim. *E-model: event-based graph data model theory and implementation*. PhD thesis, Georgia Institute of Technology, Atlanta, Georgia, USA, 2009.

[9] P. H. Kim. Web-based research collaboration service: Crowd lifelog research case study. In *Proceedings of the 7th international conference on Next Generation Web Service Practices*. IEEE, 2011.

[10] A. Lakshman and P. Malik. Cassandra: a decentralized structured storage system. *ACM SIGOPS Operating Systems Review*, 44(2):35–40, 2010.

[11] M. Ono, K. Nishimura, T. Tanikawa, and M. Hirose. Structuring of lifelog captured with multiple sensors by using neural network. In *Proceedings of the 9th ACM SIGGRAPH Conference on Virtual-Reality Continuum and its Applications in Industry*, pages 111–118. ACM, 2010.

[12] A. Sellen and S. Whittaker. Beyond total capture: a constructive critique of lifelogging. *Communications of the ACM*, 53(5):70–77, 2010.

[13] M. Stonebraker. Sql databases v. nosql databases. *Communications of the ACM*, 53(4):10–11, 2010.

[14] W. Sun, X. Zhang, C. Guo, P. Sun, and H. Su. Software as a service: Configuration and customization perspectives. In *Congress on Services Part II, 2008. SERVICES-2. IEEE*, pages 18–25. IEEE, 2008.

[15] M. Turner, D. Budgen, and P. Brereton. Turning software into a service. *Computer*, pages 38–44, 2003.