MyLifeBits: Fulfilling the Memex Vision

Jim Gemmell, Gordon Bell, Roger Lueder, Steven Drucker and Curtis Wong
Microsoft Research
455 Market St., #1690
San Francisco, CA, 94105
{jgemmell,gbell,rlueder,sdrucker,wong}@microsoft.com

ABSTRACT

MyLifeBits is a project to fulfill the Memex vision first posited by Vannevar Bush in 1945. It is a system for storing all of one's digital media, including documents, images, sounds, and videos. It is built on four principles: (1) collections and search must replace hierarchy for organization (2) many visualizations should be supported (3) annotations are critical to non-text media and must be made easy, and (4) authoring should be via transclusion.

Categories and Subject Descriptors

H.3.0 [**Information Storage and Retrieval**]: General
H.5.4 [**Information Interfaces and Presentation**]: Hypertext/ Hypermedia – *Architectures, Navigation, User issues*

General Terms

Management, Design, Human Factors.

Keywords

Memex, hypermedia, annotation, multimedia, database.

1. INTRODUCTION

In 1945, Vannevar Bush posited Memex: "a device in which an individual stores all his books, records, and communications, and which is mechanized so that it may be consulted with exceeding speed and flexibility" [2]. Bush did not foresee the exact technology to accomplish this, but he correctly foresaw two of the fundamental features: annotation and links. The MyLifeBits project is an effort to implement a personal digital store. It is Memex, extended beyond Bush's vision to handle audio and video, to perform database style queries, and to allow multiple visualizations in the user interface.

Bush posited an era of virtually unlimited storage: "yet if the user inserted 5000 pages of material a day it would take him hundreds of years to fill the repository, so that he can be profligate and enter material freely." In 2002, such abundant storage is finally on the horizon. Within five years, terabyte hard drives will be common and inexpensive (<$300). Thus, purchasing an additional terabyte of personal storage every year will be feasible for the average computer user. It turns out that filling a terabyte is not easy. Table 1 shows that it is hard to take/view enough pictures, read enough documents, or listen to enough audio in a year to fill a terabyte. Only video is up to the task of readily filling a terabyte in a year. Therefore, we must prepare for the era of profligate users that Bush predicted.[1] Users will eventually be able to keep every document they read, every picture they view, all the audio they hear, and a good portion of what they see.

Table 1: Trying to fill a terabyte in a year: for each item, the number of items it takes to fill a terabyte, and the number of items per day to fill a terabyte in a year is given.

Item	*Items/TB*	*Items/day*
300 KB JPEG picture	3.6M	9800
1 MB Document	1.0M	2900
1 hour 256 Kb/s audio	9.3K	26
1 hour 1.5 Mb/s video	290	4

Supposing one did keep virtually everything – would there be any value to it? Well, there is an existence proof of value. The following exist in abundance: shoeboxes full of photos, photo albums & framed photos, home movies/videos, old bundles of letters, bookshelves and filing cabinets. There are also profitable services in this niche, including professional video/photography at junior sporting events and companies selling materials for deluxe photo-album preparation. While many items may be accessed only infrequently (perhaps just a handful of times in a lifetime) they are treasured; given only one thing that could be saved as their house burns down, many people would grab their photo albums or such memorabilia.

Treasured as they may be, the sheer quantity of media poses a problem. In a giant shoe-box of photos, it is hard to find what you are looking for. Most items will be forgotten so that you would not look for them in the first place. It is daunting to attempt any organization. If you pick up a given item, you might have difficulty recalling what it is and what its significance was.

We are guided by a number of principles in designing MyLifeBits. First, that the user must not be constrained by a strict hierarchy as they organize their media. Second, that many

[1] Note that for all but video, the delete operation may well become obsolete: the user's time for the delete operation will be more costly than the storage to keep the item. Keeping everything does not imply that the user will be overwhelmed by the size of their collection: it is easy to filter out objects by lack of use, lack of links, and/or a low rating annotation.

ACM Multimedia '02, December 1-6, 2002, Juan Les Pins, France.

visualizations of their life bits are desirable to help understand what they are looking at. Third, that the value of non-text media is dependent on annotations. Fourth, and finally, that authoring tools create two-way links to media that they include in new media - so-called "transclusion" [7].

In the remainder of this paper, we elaborate on these guiding principles, discuss our implementation, and briefly survey related work.

2. Guiding Principles

Beginning with Bush, many have recognized that having to file objects into a single hierarchy is too constraining [4, 5, 7, 8]. After all, items may belong in more than one category. For example, my favorite pictures of my sister's 14th birthday could be filed under birthdays, favorite pictures, or my sister's name. Also, in many cases the user does not want the hassle of categorizing an item at all. Unfortunately, filing cabinets and most computer file systems force the user to place all files in a hierarchy. Computer file names mangle together the concept of ID, name, physical location and hierarchical organization. The obvious solution is to allow an object to be assigned to zero or more sets or "collections". This allows hierarchy but does not enforce it. Furthermore, excellent querying capabilities must exist to find objects that have not been put in collections, and also to define collections as dynamic entities via saved queries. Dourish et al [4] suggest the concept of "fluid collections" which consist of a query, a list of inclusions, and a list of exclusions, any of which may be null. This supports everything from conventional collections (a list of inclusions) to a saved query, and everything in between.

Having been freed form the straightjacket of strict hierarchy, it seems absurd to restrict the way we view our media. However, it has been suggested that other single metaphors replace hierarchy. For example, viewing media by time only [5, 8]. Van Dam and Foley coined the phrase "don't metaphor me in," pleading that there be more than one way of looking at things [9]. We agree. Computer visualization for gleaning insight into large datasets is a burgeoning field. Some interesting visualization possibilities have been demonstrated for video query results [3] and for transcluded material [7]. Even in the Windows Explorer we see that a folder's contents can be viewed as a list, as icons, as thumbnails, as a "filmstrip", or as a slideshow. Multiple visualizations increase our understanding and insight of our media.

Non-text media (images, video and audio) may have little value if it is not annotated by any text. To begin with, text-based searches are not possible. Furthermore, even if one is presented with the item, it may be difficult to remember what it is. Even when you remember what it is, annotation may remind you of nuances of your past thoughts on the object (or someone else's thoughts, if someone else authored the annotation). For instance, you see a photo. It is some people you remember working with in a previous job, but that's all you can recall. Now, if the system has kept the date the photo was taken, it may help a little. "8/20/1993" – now you know it was the end of summer. If the system has tracked the usage of the photo, that may also help: it was opened 18 times and emailed to 11 people. Well, it must have been considered one of the "good" photos. If the user has annotated the photo, even a little, its value jumps immensely. It might be labeled "ACME dim sum intern farewell lunch". Now you recall the occasion of the lunch (saying farewell to an intern at the end the summer) and the cuisine. If you had remembered any of this already, you could have searched for "ACME", "dim sum", "intern", "farewell" or "lunch" and found the photo.

The pinnacle of value is achieved when the user constructs a "story" out of media. By story, we mean a layout in time and space. Stories are an extension of Bush's trails – they take his simple sequence of objects and allow customized presentation. Examples of stories are slide shows, photo albums, video highlight reels, and PowerPoint presentations. Stories create the highest value for two reasons: first, because the user will select the best media to include in the story; second, because the user will attempt to present the media in the most compelling manner.

While stories are the most valuable form of annotation, it is important that they are implemented as annotations, and not just as new media, with no reference to other media that has been included. Nelson has argued for "transclusion", where two-way links between the included and including media are maintained. While Nelson's broad vision covers transclusion via edit lists, servers, micro-payments, and even copyright, the important principle for our purposes is maintaining the two-way links. Whether the implementation actually copies the source material into the new object, or just holds a reference to it can be considered a detail of caching strategy.

The links are critical because they let the user find context and commentary. When a quote is given, or a clip out of a longer video used, the pointer back to the original material answers the question: what is the context this was taken from? When viewing the original material the pointer to the new media answers the question: what commentary exists on this media? When a photo is viewed, it is extremely valuable to be able to follow a link to a story that includes the photo. When viewing the story, it is useful to go back to the original photo; it may have been cropped, revealing additional details, and one can find out what other annotations link to it – perhaps it is even used in another story. Thus, the user will build up an annotion/story web which can be browsed like the world-wide-web.

Finally, we observe that your media may be most highly valued by your descendants. For your great-grandchildren to have any appreciation of your media at all, it is clear that annotations and stories are essential. We also observe that as media is shared between family and friends that multiple annotations will be made (and annotations of annotations) with snowballing value. A single "keener" creating annotation of media from a group event can enhance the value of the media for everyone involved.

3. Implementation

MyLifeBits is a database of resources (media) and links. A link indicates that one resource annotates another. A resource can annotate any number of other resources and may be annotated by any number of others. Collections are annotations, with a recursive semantic: a collection is understood to contain all its descendants. In contrast an annotation may not apply recursively to items annotated by the resource(s) it is linked to.

While any store could be used, database features are clearly required, e.g., consistency, indexing, pivoting, queries, speed, and scalability. MyLifeBits uses SQL Server with Index Server supplying full-text search. The database schema is very simple: there is one table for resources, one table for annotation links and one table for collection links. Resources have a number of standard properties (database columns) including type, size, creation date, last modified date, and a short description.

The time interval property is used to store the time range which the content of the object refers to. For example, a scanned photo may have a creation time indicated when the scan was performed, while the time interval would be set to the date it was taken. If the date the photo was taken is uncertain, the range may be set to indicate approximate knowledge, e.g., 1975-1980. The range may also indicate what the content is about, so a document about World War II could have a range of 1939-1945.

Text searches can be performed over resource descriptions and blobs. A search can also be made for annotation text, in which case resources annotated by the specified text are returned rather than the resource of the annotation itself. For example, a search for "birthday" annotations would return some photos which had been annotated, rather than the annotation "Johnny's birthday" itself.

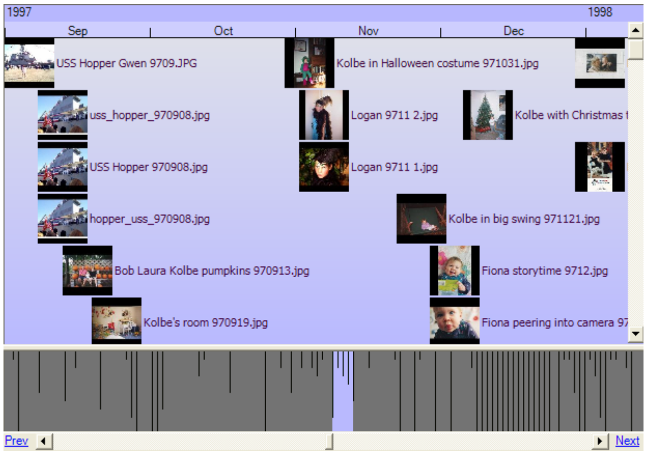

Figure 1: Timeline view of query results.

Query results can be switched between multiple views. So far, we have implemented four views: detail, thumbnail, timeline, and clustered-time. Detail view displays a list of the resources including each property. Thumbnail view shows miniature images of the resources in a grid formation. Timeline view (Figure 1) displays thumbnails on a linear time scale. The scale of the timeline view can be set to hours, days, week, or months. A histogram-like display indicates the position of the view within the entire timeline. Often, this histogram itself is just as interesting as the view, as it illustrates how the resources in the query result are distributed across time. Clustered-time view (Figure 2) clusters thumbnails by similar time, and arranges them in time order. The threshold for clustering can be adjusted (e.g., same year, month, day). All views with thumbnails allow the size of the thumbnails to be adjusted between 32 and 120 pixels wide.

A goal for all our views is information density. The UI should avoid making the user perform extra clicks (or, worse, open a new window). In particular, they should not have to click to find a dead end. So, for example, each resource in detail view shows the number of times it is annotated, number of collections it belongs to, and the number of its children (things it annotates/collects). In thumbnail view this is represented graphically. This saves the user from having to click to find out a collection is empty, or that a resource has no annotations. We have found it is useful to sort on these counts, since they often act as rough measures of interestingness. We also want to minimize the action needed to have a sense of what something is, so we display thumbnails of increasing size on mouse hover, and have an optional preview window for the selected item. Additionally, there are optional windows to show annotations, collections and children. With all these windows open, you can quickly see what a resource is, and what it is linked to.

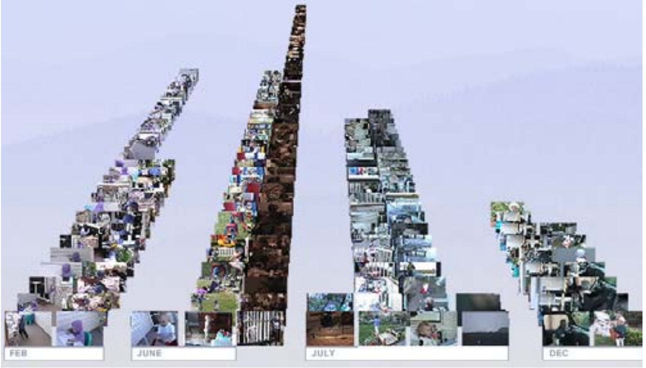

Figure 2: Clustered-time view of query results.

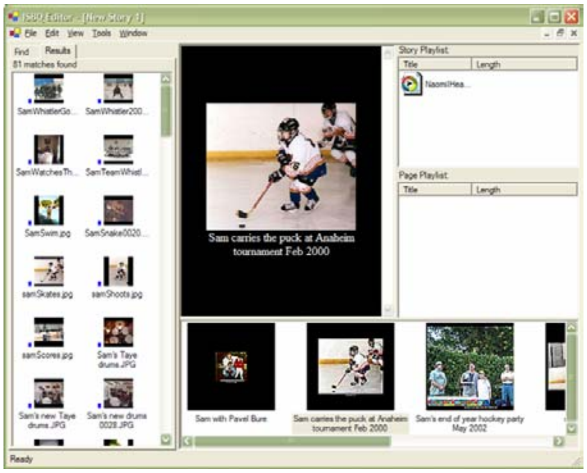

Figure 3: Interactive Story By Query slide show interface.

MyLifeBits includes a number of features to make manual annotation as ***easy*** as possible. Resources in a query result may be gang annotated, i.e., any number of them may be selected, and the right-clicking the mouse reveals an option to create a text annotation that will have links to all the selected resources. Gang collection is also supported, which is identical except that links are created to add the resources to a collection. Typing can sometimes be a barrier, so MyLifeBits also allows audio annotations to be created. Silence from the beginning and end of the recording are automatically trimmed, allowing the user to create a clip with just a start and stop click (and no editing). The Microsoft Speech SDK is used to perform speech-to-text on the audio, yielding a text version that may be searched. Another way to avoid typing is by using pre-defined annotation. For example, "thumbs up" and "thumbs down" are right-click options that create rating annotations.

Ease-of-annotation techniques have also been integrated into web browsing, in an Internet Explorer toolbar which records every page visited into MyLifeBits. As a short cut to repeated annotation, a user may type or speak a "continuous annotation" which is applied to every page subsequently viewed until turned off. Thus, a user may easily annotate an entire session of browsing.

MyLifeBits makes stories easy with Interactive Story By Query (ISBQ). ISBQ lets users make queries, and then drag and drop selections from the query result into a story. So far, we have implemented two story types: (1) A slide show (Figure 3), allowing images to be dragged and dropped into a sequence with captions added, an audio clip to be spoken (or dragged and dropped) for each image, and audio clips may be added for background music. The resulting slide show has music and commentary, and all images are transcluded, so that clicking on them opens a query window showing all resources that they are linked to. (2) A time sheet, which is a composition of multiple timelines. Resources are again dragged and dropped into each timeline. The timelines may have their scrolling locked together to allow comparison between the timelines.

Stories are stored as annotations which link to all the media they include. The content (blob) of a story contains its layout information. Our slide show is constructed using HTML+Time, and a time sheet is stored using XML that our time sheet application renders.

4. Related Work

Nelson followed up on Bush's work with ideas such as transclusion and side-by-size visualization [7]. Christel and Martin studied searching a large video library and developed a number of interesting visualizations [3].

Lifestreams [5] is a personal store that abandons hierarchy in favor of time-based visualization. They support a form of saved query to filter what is viewed. Time-machine Computing [8] extends this concept to allow the user to give 2D spatial arrangement to items in a timeline.

The work most similar to MyLifeBits is Haystack [1]. Haystack is also a personal store that supports annotations and collections. In contrast to our efforts to ease annotation and provide specific visualization tools, they have taken an ontological/agent approach, with UIs and views constructed by agents using user and system defined ontologies [6]. Placeless Documents [4] is another personal store with annotations and collections. Their innovation is "active properties" where objects may have executable properties.

5. Conclusion

MyLifeBits' charter is to fulfill Memex vision and to extend it to include multimedia data types. It supports annotations and collections, and has features to ease their creation. It enables story by query with transclusion to achieve the highest possible value-add to non-text media. It supports multiple query visualizations, including two novel time-based renderings. MyLifeBits introduces the concept of a time interval which refers to the semantic content of the object, rather than the digital representation of it, and covers a span in time.

We have begun using MyLifeBits, and initial experience is a success. Gordon Bell, our apha user, has digitized nearly everything possible from his entire life, and will have eliminated all paper (except those legally required) by the time this paper is published. MyLifeBits is an useful research tool when combined with the web, helping you organize and quickly find papers that you have reviewed. It is also a tremendous memory aid. We are finding the multiple visualizations useful for different tasks, depending on what you are looking for, or what set of objects you want some insight into.

Our future research will focus on scaling and performance issues for MyLifeBits, which, after all, will be a multi-terabyte database with millions of entries. Query performance, dealing with recursion (e.g. for collections), visualization of huge data sets, and efficient handling of video must be addressed. Different and appropriate visualization techniques will be tried and user tested. We are working on an ActiveX control to allow ISBQ/transclusion to be harnessed by any authoring tool (e.g. PowerPoint). We would also like to collaborate with AI researchers to add features that mine MyLifeBits for such things as face recognition, similarity clustering, and video segmentation.

6. ACKNOWLEDGMENTS

We are grateful for comments from Jim Gray and Ted Nelson. Victoria Rozycki helped digitize Gordon Bell's personal media.

7. REFERENCES

[1] Adar, Eytan, David Karger, and Lynn Andrea Stein. Haystack: Per-User Information Environments, In Proceedings of CIKM '99 (Kansas City, MO, November 2-6, 1999), ACM Press, 413-422.

[2] Vannevar Bush. As We May Think, The Atlantic Monthly, 176(1), July 1945, 101-108.

[3] M.G. Christel and D.J. Martin. Information Visualization within a Digital Video Library, J. Intelligent Info. Systems, 11(3), 1998, pp. 235-257.

[4] Paul Dourish, Keith Edwards, Anthony LaMarca, John Lamping, Karin Petersen, Michael Salisbury, Douglas Terry and Jim Thornton. Extending Document Management Systems with User-Specific Active Properties, ACM TOIS, 18(2), 2000, pp. 140-170.

[5] Freeman, Eric, Gelernter., David. LifeStreams: A storage model for personal data, ACM SIGMOD Bulletin 25, 1, March 1996, pp. 80-86.

[6] David Huynh, David Karger, and Dennis Quan. Haystack: A Platform for Creating, Organizing and Visualizing Information Using RDF. Semantic Web Workshop, 2002.

[7] Theodor Holm Nelson. Xanalogical Structure, Needed Now More than Ever: Parallel Documents, Deep Links to Content, Deep Versioning, and Deep Re-Use, ACM Computing Surveys, 31(4es), December 1999.

[8] Rekimoto, J. Time-machine Computing: A Time-centric Approach for the Information Environment, Proceedings of UIST '99 (Asheville, NC, Nov. 1999) in) , ACM Press, 1999, pp. 45-54.

[9] Van Dam, A., Hypertext '87 Keynote Address, CACM 31(7), July, 1988, pp. 887-895.