TELEPHONE, TELEVISION, AND RADIO IN THE HOME OF THE FUTURE

Jim Gemmell, Roger Lueder, Joshua Blumenstock, Evan Solomon, and Gordon Bell
Microsoft Research
455 Market Street, #1690
San Francisco, CA, USA

Abstract

The home of the future will have an all-digital network for all media, backed by multi-terabyte storage. Users will be able keep an entire lifetime of personal media, and vast collections of media that may be of interest for future viewing, reading, or listening. MyLifeBits is a personal store for a digital life, designed to support efficient organization, search, browsing, annotation, and viewing. This paper describes the telephone, television, and radio components of the MyLifeBits system.

Keywords

Audio, video, telephone, television, radio

Introduction

Two clear trends are in effect to shape media in the home of the future. The first is the switch from analog to digital for both storage and transmission of all media, including television, video, music, telephone, and radio. Thus, the current mess of analog wiring, remote controls, and isolated components is poised to be replaced by a fully integrated, all-digital, home media network (Figure 1). The power of computers can be brought to bear in this environment to drastically improve and enhance the user experience [3]. The second trend is the rapid growth of magnetic disk storage capacity. Disk capacity has been doubling approximately every 18 months, and should continue to do so for at least the next five years, implying the standard PC hard drive will grow from the current 100+ gigabytes to a terabyte within five years. The massive storage space available to users will allow them to keep a lifetime's worth of virtually all media.

The combination of these trends points to a home media network backed by massive storage. Ideally, there would be relatively few servers to maintain, but many networked IO devices to provide convenient access around the home. E.g., dynamic picture-frames, monitors, speakers, microphones etc. (Figure 1)

This paper describes prototypical work to support telephone, television and radio in the home of the future. These components are part of the MyLifeBits project [7], an endeavor to realize the Memex vision posited by Vannevar Bush is 1945 [4]. Bush proposed that Memex should have virtually unlimited storage, and should support such features as full-text search, text and audio annotations, and hyperlinks.

MyLifeBits is intended to store a lifetime's worth of *everything* – at least, everything that can be digitized. In addition to TV, radio, and telephone, which are the focus of this paper, it also supports the usual media types found on a PC, including email, photos, documents, contacts, web pages, and music. We attempt to log as much usage information as possible. MyLifeBits supports full-text search, annotations, typed linking, and search result clustering/filtering.

In the remainder of the paper, we describe the television, radio, and telephone components of MyLifeBits. We discuss our experience to date and related work. Finally, we present our conclusions and plans for future work.

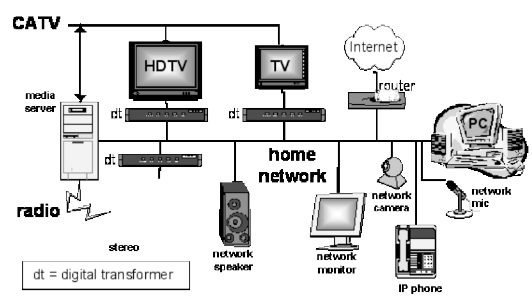

Figure 1: The Home Media Network. Note that legacy analog devices require a "digital transformer" to connect to the digital network.

Television

Television storage consists of a database of metadata, and videos stored in MPEG-2 or WMV format in the filesystem. Both the metadata and the videos are accessible anywhere in the home via a web interface, or custom programs. In this paper, we focus on the web interface. The home page for BarcTV is shown in Figure 3. On the left is a menu, with choices to search/browse the EPG, watch live TV, and to search/browse recorded shows. There are also some statistics – how many shows/hours/gigabytes have been recorded, and the latest date for which we have EPG information.

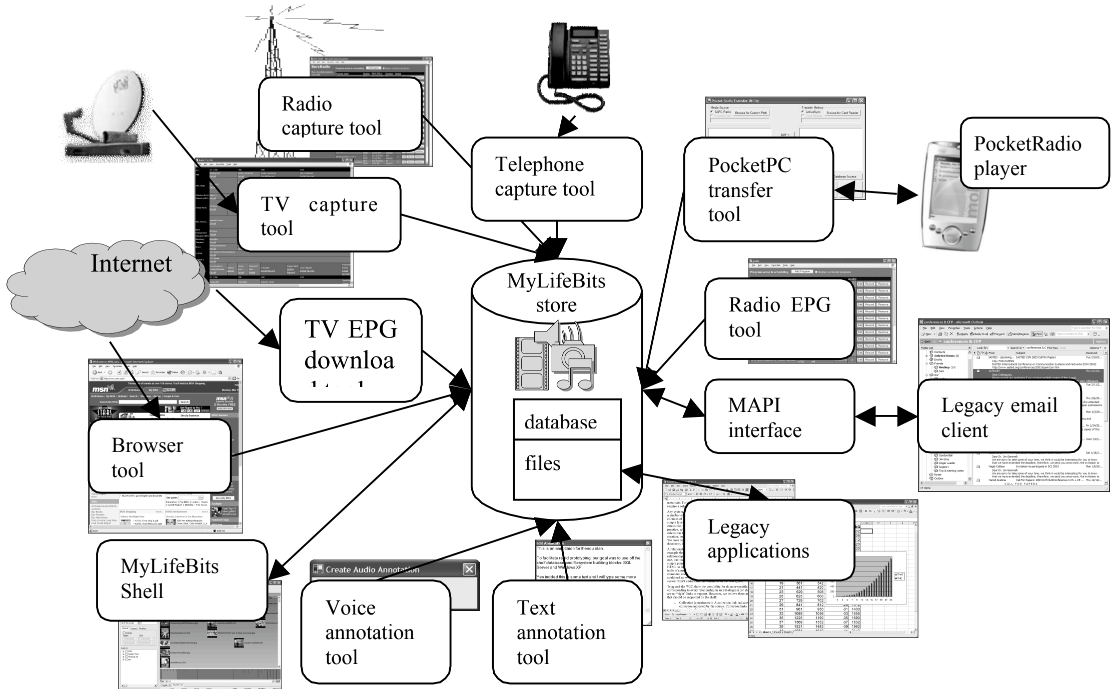

Figure 2 – The MyLifeBits system

The first step in the television system is downloading the Electronic Program Guide (EPG) information. The EPG has a schedule of what shows are being aired on each station. It includes information for each show, such as the title, a descripton, MPAA rating, cast members, etc. This information is downloaded nightly, and converted from its raw format into tables in our database.

Figure 4 shows the EPG schedule. For each show in the schedule, the user can click for more details, or click to record the show. Shows that are scheduled for recording are colored dark blue. Shows that are available for recording are colored light blue. Shows that cannot be recorded because of a tuner conflict are colored gray. The system will support multiple tuners, and will reassign tuners to shows as necessary to prevent conflicts.

Figure 3 – The BarcTV homepage.

For each tuner, there is a recording application that polls the database every few seconds to see if there is a show scheduled to record. If there is, it launches a recording. We have setup one recording application to use the Windows Media Encoder, and another to use an MPEG-2 card to record shows. Storage of the video is distributed. The database keeps a list of all network drives available to store video on. The recording application selects the drive with the most free space to record its next show on. Presently, we have two PCs, each with about 7 drives on them, for a total of over 2 terabytes of storage space for video. We have recorded over one terabyte of video (500+ shows, 700+ hours). In the future of commodity terabyte hard drives, it will be affordable to buy one every year, allowing users to build vast video libraries.

The details display is shown in Figure 5. It includes meta-data about the show, links to look up the show on the internet (at Yahoo! Movies or IMDB) and text input area for creating annotations about the show.

The interface for simple search is shown in Figure 6. The string entered is searched through all meta data for a show, including title, description and cast. Figure 7 shows the results of the search. Figure 8 shows how shows can be browsed by genre.

Recorded shows can be searched, browsed, and annotated in a similar fashion to scheduled shows. They can also be listed in order of most recent recording (Figure 3). Their video files can be played in the Windows Media Player, or played embedded in the web page with HTML + time (including skipping ahead or back by 15 seconds - Figure 9)

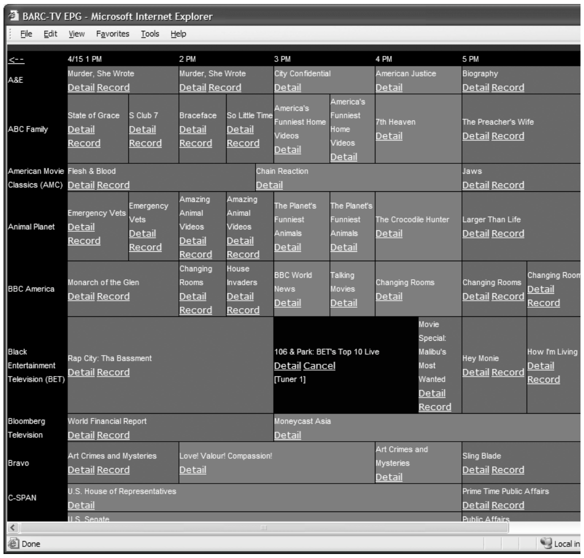

Figure 4 – Television EPG

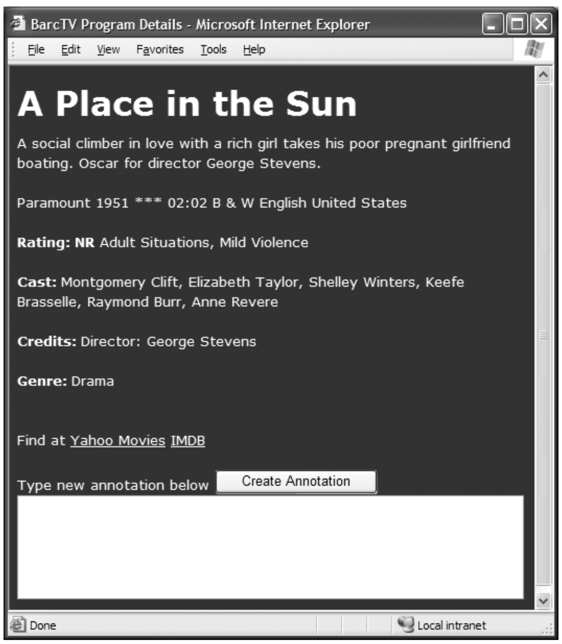

Figure 5 – Details of a show

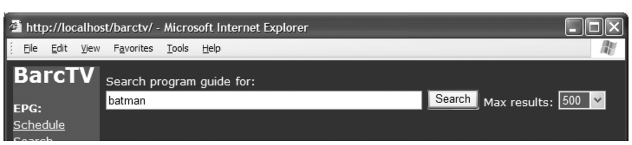

Figure 6 – Simple EPG search interface.

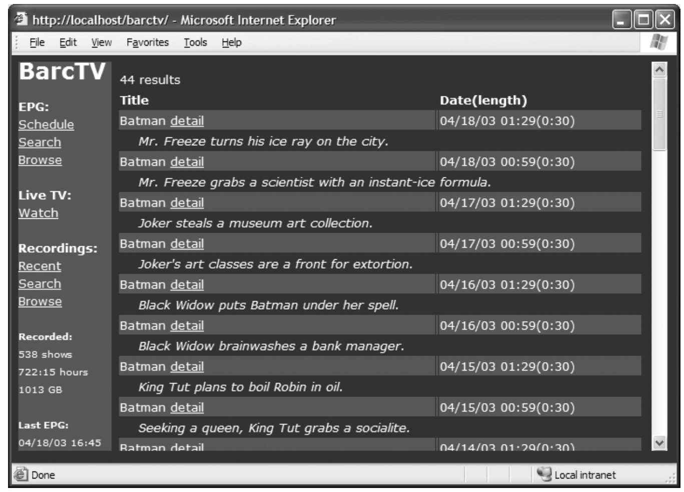

Figure 7 – Search results

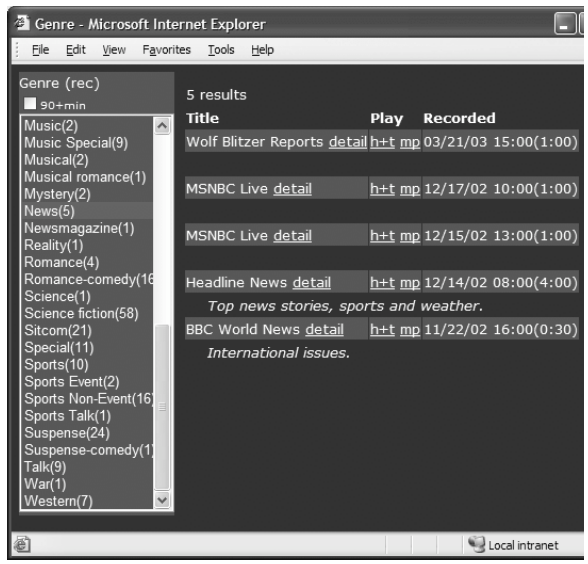

Figure 8 – Browse by genre

Figure 9 – embedded HTML+time player

Radio

Radio is similar to TV. There is a database of EPG information. Recording programs poll the database and store audio recordings on network file shares. We use a USB FM tuner and record using the Windows Media Encoder.

There is no EPG source for radio. It may be possible to parse each radio station's web page, but at this time it is easier and more flexible to allow the user to define regularly scheduled shows of interest to record. Figure 11 shows the interface for defining radio EPG entries and scheduling their recording.

As with TV, radio can be played live, or streamed on the home network. Recorded shows can be searched or browsed. Figure 10 shows the browse interface, with HTML+time to support playback with time skipping.

In addition to playing in the home, we felt that radio is critical for mobile use, especially in a car. Therefore, we made a prototype mobile radio player using a PocketPC. The PocketRadio transfer utility (Figure 12) is used to transfer radio shows and other audio (e.g. MP3 and WMA songs) to the PocketPC. The PocketRadio player application plays the audio. It logs all user activities: every pause, skip, play, etc. The next time the transfer utility is used, this log information is added to the database. This allows us to build a user profile to understand what shows and parts of shows a user enjoys.

The PocketRadio player has a GUI. However, looking at a GUI while driving a car is, of course, dangerous. Therefore, we re-mapped all the keys on the PocketPC to functions for PocketRadio (e.g., next program, skip 15 seconds). This helps, but it is disorienting to the user to guess at what show is being played, or, she may wonder if the right key has been pressed. To provide more "eyes-free" feedback, we implemented a spoken UI, using text-to-speech. Every key press causes a spoken response. For example, pressing the skip button results in the verbal feedback "skip". When the user finishes skipping, the name of the show is spoken, e.g. "BBC news, recorded March 10th at 7 PM". All audio needed for the spoken UI is pre-generated and stored as audio clips so that the player does not need to run any speech-to-text code. Ultimately, the production version of our prototype should be an actual car stereo, with wireless networking that automatically performs the transfer function when the car is in the home garage.

Figure 10 - Browsing and playing recorded radio

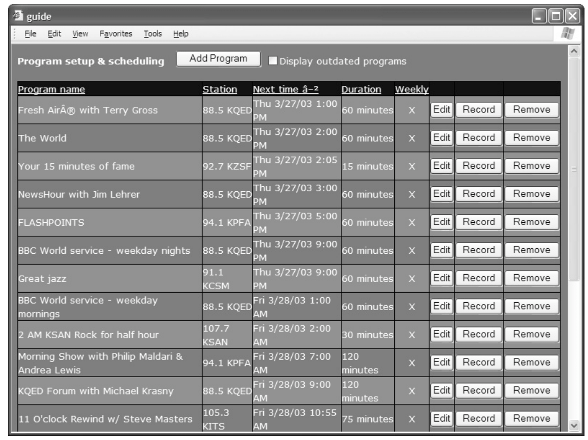

Figure 11 – Radio EPG interface

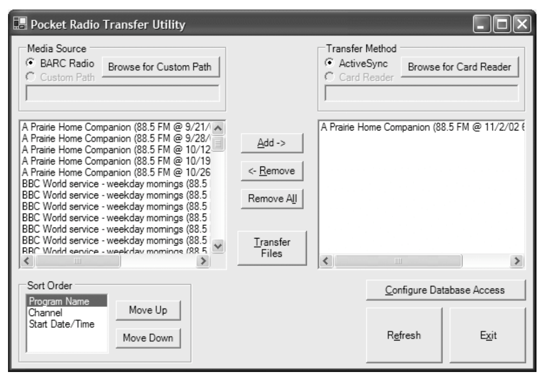

Figure 12 – Pocket Radio transfer utility

Telephone

The telephone recording application is intended to act as an answering machine, and to record two-way calls. All data for the call is stored in the MyLifeBits database, including meta-data, and the audio file itself (stored as Windows Media Audio – WMA). To play the audio, it is copied out of the database to a temporary file and played. The meta-data includes the time of the call, the duration, and the phone number obtained by caller-ID.

Various laws pertain to telephone recording. In California, where our lab is located, parties must be notified that recording is happening. For the answering machine, the usual greeting played suffices. For live calls, we have a warning message (set by the user) that is played letting the other party know that the call may be recorded.

MyLifeBits supports hyperlinks for organization of a user's media. When the telephone app detects that the caller-ID number corresponds to a contact in the user's database, it suggests a link between the contact and the phone call. Usually, this link will be correct, but, on occasion, you may receive a call from someone else using that contact's telephone and the link would not be correct and should be rejected. The MyLifeBits shell shows all linked instances to the currently selected instance, so selecting a contact will show phone calls with that contact (along with photos of that contact, etc.) as in Figure 13

Figure 13 – MyLifeBits shell. The contact Connie McFadden is selected in the top right pane. The lower right pane show instances linked to Connie, including a photo of her (with a link type "capture of"), and a telephone call (with link type "caller").

The telephone recording application supports speech-to-text using the Microsoft Speech API. At this time, the low quality of the telephone audio, combined with little or no training for most speakers, leads to poor results. However, as we plan to keep the recorded telephone calls forever, we look forward to the time the speech-to-text works with phone-quality audio. Also, we expect that as we do more training with our own voices that we can get good recognition for at least the user's part of the call.

Discussion

Our experience to date suggests that having massive storage changes the way users will interact with their media, and makes new demands on user interfaces. For example, with terabytes of shows that are likely to be of interest to the user, the odds of the user preferring a live show are very low. The exception is for live events (sports, news), although we expect that users may come to prefer time-shifted versions of even these events in many cases. One author discovered that he only listened to about 15 minutes of the radio news hour once he had the ability to skip. This time-savings led to a preference for recorded news over live news, even if it was a few hours out of date.

The implication of this is that the standard time-grid of EPG information (Figure 4) is likely to be neglected by users in the future, and, consequently, unimportant. This is true even for scheduling recordings; the user will not generally care when a show is being broadcast, she will just want to create a "wish list" of shows to be recorded, regardless of schedule time. We intend to create a UI for the wish list, and algorithms for selecting one of possibly many air times to record the show. We also plan to add recommendations, based on previous user interaction (addition to wish list, playing the show, annotation with a ranking, etc.).

What will remain important is search and browsing. The UI for these functions must scale to an extreme number of shows. Flashy UIs that only display a few search results at a time on the screen will not be useful when the user needs to browse hundreds of items. This presents a real challenge when the UI is on a PocketPC, or on a TV screen (TVs have very limited resolution, so they cannot display many lines of text). Very involved searching and browsing will best be performed on a PC with a high resolution display. Perhaps after some amount of searching and browsing on the PC, with certain shows flagged, the lower resolution devices will be able to work with the shortened list of flagged shows. We believe that the tablet PC will become the remote control device of desire in the future, with plenty of resolution for large search results right in your lap, as you watch TV (or listen to your stereo, etc.).

It is clear that annotation is an important feature. Text annotation is searchable, so it can be used in future searches. Ratings annotations help in learning user preferences. Audio annotations, besides being useful when translated into text, can be very interesting and valuable in their own right. Consider, for example, and audio annotation of a phone call, created just after the call to capture one's impressions of the call. Even without speech-to-text, these verbal notes can be very valuable in their own right for the content and emotion they contain.

Usage logging can be mined to find user preferences. It also is useful for searching. E.g., find recently played phone calls, or most replayed videos. Within a single show, sections that have been replayed more than others can be designated as highlights.

Having a database integrated in our storage system yields powerful features. For example, we can quickly generate reports and graphs to help understand what is in the store, and how it has been used. The database also helps by allowing fast pivoting operations when the user is not able to directly search for a desired item. For example, suppose Bob wants to replay a certain phone call with his lawyer. However, he has had very many, and he cannot remember the date of the call. So, there is nothing he can think to search on, and browsing would take too long. Suppose, though, that Bob recalls looking at a web page about patent laws during the phone call. Since Bob has not viewed many web pages like that, a search for "patent laws" quickly finds the page. Bob the performs a "time pivot", i.e., the entire database is sorted by time, and the cursor in the UI is moved to the point in time when the web page was viewed. Only a little scrolling from this point is needed to find the phone call, since it happened on the same day. In fact, with MyLifeBits, the result of the pivot can be "filtered" to only include phone calls, making it even easier to find.

Finally, we should mention that Digital Rights Management (DRM) in no way negates the vision we postulate here. DRM has already been designed that will permit users to keep copies, and to only license (pay for) them upon actual usage. Also, schemes are possible where advertising (loaded separately) can viewed in lieu of payment.[1]

Related Work

A number of research projects have explored different ways of organizing a personal store. Some have advocated a purely time-based organization/UI [6] or time, combined with a location on the user's desktop [9]. Others have relaxed hierarchy, allowing files to have more than one parent [1][5]. The ShoeBox project [8] emphasized the importance of annotation (especially audio annotation) for retrieval of images.

Commercial Digital Video Recorders (DVRs) already perform many of the functions of our television component. Examples are Tivo, ReplayTV, and Windows XP Media Center Edition. However, they tend to be oriented towards the traditional EPG grid, and have UIs that do not scale well to extremely large video collections. The audible.com audio player has some spoken UI, to our PocketRadio, but does not have spoken show names, or recording of usage information.

Bates [2] has described some of the difficulties is scaling search interfaces, including Bradford's law, which implies exponential browsing to get all desired results, and the fact that the vocabulary remains somewhat fixed as the store scales, resulting in more hits for any query over time. She concludes that UIs that work for small numbers may not work for large numbers.

Conclusion

We have presented the telephone, radio, and television components of the MyLifeBits system. They show what media management will be like in the home of the future, with an all digital network backed by terabytes of storage. Having such massive storage capacity will make search and browsing of large numbers of media items a top priority. Key features in MyLifeBits are logging, annotation, and organization via collections and hyperlinks. Our PocketRadio audio player supports a fully "eyes-free" spoken UI for use in an automobile.

We plan to add a number of features, including a wish list with automatic recommendations, recording and search of TV close-captioning, and a tablet PC browser that acts as a remote control to drive a large screen and speakers.

References

[1] Eytan Adar, 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] Marcia J. Bates, Indexing and Access for Digital Libraries and the Internet: Human, Database, and Domain Factors, *Journal of the American Society for Information Science*, Vol. 49, No. 13 (November 1998), pp. 1185-1205

[3] Gordon Bell and Jim Gemmell, A Call For The Home Media Network, *Communications of the ACM*, 45(7), July 2002, pp. 71-75.

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

[5] 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.

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

[7] Jim Gemmell, Gordon Bell, Roger Lueder, Steven Drucker, and Curtis Wong, MyLifeBits: Fulfilling the Memex Vision, *ACM Multimedia '02*, December 1-6, 2002, Juan-les-Pins, France, pp. 235-238.

[8] Timothy J. Mills, David Pye, David Sinclair, Kenneth R. Wood , Shoebox: A Digital Photo Management System *AT&T Technical Report* 2000.10

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

[1] Groups like the TV-Anytime forum are working on such DRM issues http://www.tv-anytime.org/