Supporting the Retrieval Process in Multimedia Information Systems

Rick Kazman, John Kominek
Department of Computer Science, University of Waterloo
{rnkazman, jmkomine}@cgl.uwaterloo.ca

Abstract: This paper discusses problems inherent in retrieval from multimedia information systems. These problems stem from the diverse and inherently unstructured nature of multimedia. We examine some problems in this area, identify some notable successes, and present a set of user interface design principles for creating effective interfaces to multimedia information. We then discuss information retrieval problems associated with Jabber, a system that captures, indexes, and allows users to browse the results of multimedia meetings. Finally, we detail a set of techniques that we are using in Jabber’s user interface that follow the design principles, and attempt to address the problems of retrieving information from a multimedia database.

1. Introduction

This research has been exploring the indexing of video-conferences through two techniques: by what people say and when, and by participant’s temporal interaction patterns. Video-conferences, like most human-to-human discussions, pose a challenge for indexing due to their inherently unstructured nature.

We are attempting to automatically extract meaningful information and present it in a visually accessible form. Our working hypotheses have been [6]:

- That analyzing speech provides the most useful information for indexing. Esoteric technologies (e.g. facial and hand gesture recognition) can at most provide supplemental information.
- That combining multiple indexing techniques provided more powerful indexing and retrieval than any single technique.
- That indexing in real-time can provide feedback while a meeting is in progress.
- That searching and browsing should be presented to the user as a unified task.

To test these hypotheses, we have built several working prototypes of the indexing and retrieval system, the most recent of which is called Jabber-2. Through this prototype, we have been able to show the utility of the indexing techniques. We can search through meetings based upon: the words spoken, the topics discussed (as defined by lexical chaining), or the temporal idioms. We can also find decision points in the meeting by examining changes in temporal idioms.

In addition, we have concentrated on techniques for integrating browsing and searching, so that the transition between the two is simple and natural. Finally, we have used these indexing techniques to create real-time feedback: monitors of a meeting’s activity in terms of temporal interactions and spoken topics.

2. Previous Work

New ways of indexing, organizing and querying multimedia databases are being developed. For example, Smoliar and Zhang [11] report that they have indexed their database of video images according to: free text retrieval of the descriptions of the images, category-based retrieval based upon frame-like categorizations of the images, and content-based retrieval based upon color, texture and shape, following the lead of the IBM’s QBIC project [8]. Lum and Meyer-Wegener reported a similar system [1] wherein images were accompanied by free-text descriptions which were parsed into Prolog predicate calculus assertions. These approaches are promising because they allow a user to pose a query in a form that is relatively natural, and which accords with the way we think about multimedia.

2.1. Accessing Multimedia Databases

In computer-based information systems, as in real life, people access information in one of three ways: selection-based, navigation-based, or retrieval-based. This tripartite distinction is sometimes categorized as: grazing, browsing, or searching. Roughly put, in retrieval-based programs the user “stays where they are” and information is brought to them on command. In navigation-based programs, the user “enters and travels” through the information repository. To state it metaphorically, the first camp treats information as hidden inside a black cave, dutifully tended by servants. The second illuminates the cave and
invites you to enter. When one uses SQL, one is searching; with the World-Wide Web one is browsing. The most widespread example of information grazing is television viewing. We do not discuss grazing in this paper.

Traditional databases support searching via querying their contents. SQL is the most common example of a database query language. However, as databases are extended to handle media other than text, the query languages are correspondingly extended. We describe three extended query methods, presented in chronological order of development: Query by Associated Attributes, Query by Description, and Query by Image Content.

2.2. Query by Associated Attributes

Query by Associated Attributes is the least radical approach, being based on a simple extension to relational schemes. The record-and-tables organization is still used, with images appended as extra fields. For instance in the schema “City(name, pop, lat, long, map),” the first entry is text, the next three are numbers, and the last is a graphic. Often the image or graphic fields just “tag along” as large bitmaps. Information retrieval is made solely through the other, associated fields. A query by name could retrieve “Manhattan”, for example. By specifying a latitude and longitude range about Manhattan, points on Long Island could also be retrieved.

One example of such an approach is PicQuery, developed Joseph and Cardenas [4]. PicQuery emulates the style of QBE (Query By Example), another traditional query language. The user specifies a search by filling out a table of specifiers. To illustrate, consider Figure 1.

```
<table>
<thead>
<tr>
<th>PIC</th>
<th>Variable</th>
<th>Relation</th>
<th>Value</th>
<th>Logical</th>
<th>Group</th>
</tr>
</thead>
<tbody>
<tr>
<td>PIC</td>
<td>elevation</td>
<td>.gt.</td>
<td>1000</td>
<td>.or.</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>elevation</td>
<td>.lt.</td>
<td>1000</td>
<td>.and.</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>rainfall</td>
<td>.gt.</td>
<td>3</td>
<td></td>
<td>2</td>
</tr>
</tbody>
</table>
```

**Figure 1: An Example Form Filling Query**

This translates into “Identify all the places in Utah with elevation greater than 1000 feet. Also include those places less than 1000 feet that have annual rainfall greater than 3 inches. Display the results and call that picture PIC.”

In a similar vein is PSQL (Picture Structured Query Language) [10]. Here is an example query.

```
select city, state, population, location
from cities
on us-map
where location WITHIN windows
  (4 +/- 3, 11 +/- 9) AND
  population > 450,000
```

This query is similar to regular SQL, except that it has been extended to handle spatial information. Instead of typing in latitude and longitude as numbers, one simply manipulates a resizable rectangle in the display window. The returned information is displayed partly as a table of values and partly as a graphic as shown in Figure 2.

```
<table>
<thead>
<tr>
<th>City</th>
<th>State</th>
<th>Population</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jacksonville</td>
<td>Florida</td>
<td>672,971</td>
</tr>
<tr>
<td>New Orleans</td>
<td>Louisiana</td>
<td>496,938</td>
</tr>
<tr>
<td>St. Louis</td>
<td>Missouri</td>
<td>396,685</td>
</tr>
<tr>
<td>Memphis</td>
<td>Tennessee</td>
<td>610,337</td>
</tr>
<tr>
<td>Nashville</td>
<td>Tennessee</td>
<td>545,651</td>
</tr>
</tbody>
</table>
```

**Figure 2: Results of a PSQL query**

It is unlikely that a small set of new data types will suffice for general multimedia databases. Nonetheless, by restricting themselves to geographical information, Roussopoulos et al needed only three: points for cities, regions for lakes and counties, and segments for rivers and roads, all of which are objects embedded within a data record. The database manager is unconcerned with the details of how an object represents an item, only that specific information can be calculated (e.g. area).

A PSQL expression could return, for example, all the rivers that cross a particular highway (though phrasing such a query is perhaps a little too complicated for non-programmers). So, in addition to retrieving an image via associated attributes (e.g. river names), the associated attributes can be retrieved via queries made on the spatial properties of that image (e.g. those rivers that cross a given highway).
2.3. Query by Description

The hallmark of a good system for data management is direct data access. However, many existing database packages force the user to spend more time interacting with the interface than with the information. Query by Description attempts to do for multimedia databases what query by example did for traditional ones: lessen the cognitive distance between the user and the data. The idea is that by storing descriptions along with each image the user can ask things like “find me modern bungalows with two-car garages,” in intent if not in exactly those words. An experimental system that has achieved moderate success is reported by Lee and Macgregor [7]. Their image database consisted of a thousand paintings from various cultures and periods. To retrieve an image based on the database, the user plays a game of 20 questions, with the computer asking things like:

1. Can you see people or parts of people in the picture (including angels, cupids, ghosts)?
2. Can you see bare feet?
3. Can you see a boat?
4. Can you see any fruit?
5. Can you see any sky?
6. Can you see any fruit?
7. Can you see any sky?

The computer compares the user’s list of yes/no/maybe answers to those compiled previously for each image in the database, retrieving the closest matches. Experiments have established that high retrieval accuracy is possible, but not until a good set of questions are determined. Terms in the questions must be carefully chosen so as to discriminate the images well—preferably, each question cuts the candidate field in half. Had all the items been landscape paintings then asking if there is visible sky is not helpful.

The terms must be unambiguous—qualities such as dynamic, vibrant and summery are subjective, and thus open to widely divergent interpretations by untrained users. Concrete terms are preferable to abstract (fruit is better than food), but cannot be too concrete or else they would appear in only a few images (sailing ship, barge, punt, and steamer are best merged into boat). In sum, an ad hoc approach to selecting and naming image features does not work. Only through empirical analysis can it be discovered that bare feet, for instance, forms the nucleus of a good question.

2.4. Query Reformulation

A number of Multimedia Information Systems (MMISs) distinguish themselves in an important way: queries are performed iteratively, thus combining some of the features of searching and browsing ([8], [9]). Because human memory is associative, being able to refine a search based on the results of successive stages is desirable. It matches the computer to the way humans operate, not the reverse.

The value of iterative searching or “retrieval by reformulation” has long been recognized by user interface researchers, though less so by the database community. Williams et al pioneered this strategy in RABBIT, a prototype developed in 1981 at Xerox Parc [13]. Fischer extended the idea with HELGON, a software environment designed to ease the problem of using tools and finding information from complex resources (in their case the interLisp class library). It is not the size of an information body, primarily, that makes retrieval difficult, but its heterogeneity. A phone book is big, but not daunting. In contrast:

Users of high functionality systems, which contain tens of thousands of object and tools, suffer from a lack of knowledge about the interdependencies between problem articulation and specification, and of knowledge about the tools that exist for solving these problems [2].

To put it simply, users cannot ask questions about things they do not know exist. Taking the cue from scientific visualization, HELGON graphically displays the structure of the underlying information store (a hierarchy of categories). This allows users to refine their description based on feedback until a suitable item is found or until the absence of such an item is established. As an adjunct to retrieval by reformulation, users can browse the entire structure in search of a desired object. If the categorization is tractable, users will gain an overall sense of the system.

3. Present Work

In the Jabber-2 system, we are also looking for methods to support the retrieval of multimedia information, and we are looking for retrieval techniques that mirror the ways in which humans think and work. Thus, we must support smart natural language-based retrieval, query reformulation, relevance feedback, and query by associated attributes. Our particular domain of interest has been multimedia information created as a result of video-conferences (potentially with shared applications). Unlike the applications described above, this domain does not leverage pure video information well. For the most part the video consists of talking heads.

This forces us to rely on what participants say (and how they say it). Therefore, speech recognition is necessitated. But as we will explain, 100% recognition accu-
racy is not required by our system. Even if we have 100% recognition, however, the spoken words do not in themselves create a meaningful index. This is because of the “vocabulary problem”.

3.1. The Vocabulary Problem

The vocabulary problem arises because there are many ways of expressing similar ideas using natural language. For example, some people might use the term bungalow to describe a house, some might use detached home, and some use single story dwelling. Because of this ambiguity, a Query by Description scheme will only be able to find a fraction of the desired images, thereby hindering a successful complete search. The vocabulary problem is a significant one for multimedia databases, because so many of them rely on associated descriptions to index and retrieve stored resources. Approaches to dealing with the vocabulary problem in multimedia databases draw from the field of information retrieval, the second major influence on the advancement of multimedia information systems.

Furnas et al found that even within a constrained area of discourse people rarely agree on what to name things [3]. Expert cooks, in one experiment, were asked to extract keywords from recipes. The probability that two people applied the same term to an object was 18%. In another test, a group of 48 typists were shown typed pages with editorial markups and asked to devise names for each markup operation. Here the terminology overlap was only 7%. The operation of crossing out a word was variously called delete, remove, change, spell, make into, and so on. The authors found that heavy aliasing (using about twenty synonyms to describe the same concept) was required for moderate retrieval rates.

The vocabulary problem also runs in the opposite direction—the same word often has multiple meanings. The word gravity, for example, might refer to a physical force or to a state of dignity and sobriety.

In sum, indexing by keywords is prone to false hits and missed hits. What is really needed for successful information retrieval is an index of topics, not of spoken words.

3.2. Topic Identification

To identify topics in the Jabber-2 system, we first use speech recognition to recognize the audio portion of a meeting, and then use a technique known as “lexical chaining” to recover semantic clusters of words from a discourse [6]. These semantic clusters serve to identify common topics. Lexical chaining attempts to group words into coherent sets, using lexical relations between the words as a guide. A large number of relationships are contained in WordNet, a 90,000 word on-line thesaurus that links words through semantic relationships. WordNet links synonyms, and antonyms, as any thesaurus would, but also links words according to a richer and more sophisticated set of relations such as: part-whole, subtype, super-type, entailment, causality.

Employing WordNet allows us to mitigate the vocabulary problem. For instance, in the sentence “I’ve decided to sell nickel and buy iron and copper” the close relation between iron, nickel, and copper allows us to safely reject the sense of nickel as “a coin worth 5 cents”.

Having identified likely topics, we can then use them to build indexes back into the various information streams. To create these indexes, we record the time at which each word was uttered, and store this time along with the word. Given that such indexes have been created, we can now support theme- or topic-based retrieval of video contents.

Also, with this approach we can accommodate the low recognition rates typical of even the most advanced commercial speech recognition systems. Given that themes (or topics of conversation) involved many related, and hence redundant words, the implications of missing 1/4 or 1/3 of these words—as is anticipated when using speech recognition—are far less damaging when we are indexing by theme. This is because 2/3 or 3/4 of the theme words will be indexed, and so the user will still be able to locate the relevant portion of the conversation through the theme.

Although retrieval by theme is much more powerful than retrieval by keyword, we recognize that no single indexing technique is as powerful as the synergistic combination of multiple techniques.

3.3. Temporal Idioms

Meetings are more than discussion topics and agenda items. They contain salient patterns of interpersonal discourse—we call common patterns “temporal idioms.” These idioms can be identified by examining the interaction among the meeting participants over time. The identification of these idioms presents an intriguing view on a meeting, allowing indexing in terms of how people interact, as well as what they say.

We determine temporal idioms by detecting periods of speech and pauses in an audio sample, completely ignoring the spoken content. Once we have this information we feed it through an recognizer, implemented as an augmented finite state machine, that can identify patterns of
individual speech and of combinations of speakers. For example, we have identified the following conversation types, purely by their patterns of temporal interaction:

- Presentation
- Question and Answer
- Discussion
- Argument
- Agreement

Not only are temporal idioms useful as an indexing technique for later meeting retrieval, but we have noted that changes in idioms occur at decision points in a meeting. For example, if a long presentation by one speaker is followed by a discussion, which is followed by a short question and answer period (e.g. “Do you agree with the proposal?”), then this indicates that a consensus is being reached. The temporal idiom indexing technique alone can be quite useful. Used in combination with theme-based retrieval, however, a user can more easily find “interesting” portions of a meeting.

4. User Interface Design Principles

To counter the contemporary notion that anything with a GUI is necessarily great software, Tognazzini defines the term “visible interface” [12]. A visible interface is a complete environment in which users can work comfortably, always aware of where they are, where they are going, and what objects are available to them along the way. A simple counter-example: consider removing the cursor from a data entry screen; everything else is still there to be seen, but what remains is not a visible interface.

Here, then, are some principles of design for multimedia information organization; a mixture of Tognazzini's views and our own:

- **Center the application about the user.** Consider a painter at his easel. The easel provides a central work-space for activities, with the necessary tools (paint, brushes, thinner, etc.) close at hand. Good software is like that. Internally the user may be traversing a labyrinth of code; externally, functionality and information comes to them. Tognazzini provides the example of a library simulation built with Macintosh HyperCard. One HyperCard might show a card catalog sitting in a library. Click on the label marked M-P and that drawer opens up. To the program the user has moved from one node in the hypertext network to another. To the user the drawer simply opens. There is no sense of navigation to contend with, and no way of getting lost.

- **Make the interface visible.** In particular, incorporate familiar landmarks into an application, especially in an information retrieval application. People need visual landmarks to serve as anchor points lest they become disoriented—the problem with driving a car in an unfamiliar city is a lack of landmarks.

- **Provide a unifying theme.** Consider a multimedia manual for a jet aircraft. The information could have been decomposed into an electronic parts manual, a maintenance manual, an engineering design manual, and so on. That would have been more true to familiar paper manuals, but an inferior program. The three dimensional representation of the aircraft—being able to manipulate and query upon it—provides a central unifying theme and a powerful anchor. The context of what's available is always visible.

- **Make object interaction direct.** This is the advantage of geographic databases over some extended version of SQL or QBE. Imagine a command line driven paint program where the coordinates and characteristics of every graphic element must be typed in. The user should interact directly with information, not struggle with interface artifacts. Software of a more serious nature must also make this transition from indirect to direct interaction.

- **Provide many open doors.** Of the many approaches to information organization, don't feel compelled to pick the one best approach. A complex information repository will require many open doors and the best system is a hybrid—perhaps a combination of Spatial Organization with Query by Image Content and other query abilities. This is not an argument against interface consistency. Rather, match the needs of your application to the tools available. A well designed system pays heed to all design principles, including this one.

5. Jabber's Approach to Information Retrieval

The Jabber-2 system provides a unified approach to retrieving the stored contents of multimedia meetings. It is “unified” in two senses: it attempts to unify searching, browsing, and grazing, and it unifies several different indexing techniques. Users can access these indices individually or in combination. Figure 3 shows a screen of Jabber-2 being used as a search engine. The Jabber-2 user interface is comprised of several areas. The largest section is devoted to a time-line view of the meeting. An iconographic view of each meeting participant is shown
at the left of a time-line. As the meeting progresses the time-lines grow dynamically, moving from left to right. Text is recognized by the speech recognition engine, and connected together by the lexical chaining algorithm. Recognized words are shown as buttons along the time line. A user can click on any time-line button and the stored meeting records will replay from that time.

Below the time-line view in Figure 3 is a theme view. Themes are shown on the left, and the words that belong to the theme are shown to the right.

In this figure, a user is specifying different kinds of search criteria. When this search is executed, the system responds (returns a result) by coloring all buttons in the time-line that correspond to the query. The user is then able to browse the results of the query within their original time-line context, to reformulate a query based upon the returned results, or to begin a new query.

5.1. Meeting Monitoring

As described in section 3.3, we have the ability in Jabber-2 to automatically recognize temporal idioms such as “argument”, “discussion”, “presentation”, etc. in a meeting, by feeding a filtered audio signal for each participant through an augmented finite state machine recognizer. Currently the recognizer’s accuracy at determining meeting themes is about 90% for an individual user (that is, it can be trained to mimic a human’s subjective judgements with 90% accuracy), and 70% for a group of users. The accuracy is lower for a group of users because humans differ in how they subjectively classify temporal idioms; one user’s argument is another’s discussion. Even at 70% accuracy, however, the indexing technique is useful.

For example, we have mentioned that changes between idioms can signal a decision point in a meeting. This is a powerful way of indexing a video-conference that goes beyond word-based retrieval.

This same technique can also be used to monitor a meeting, with some important social implications. We can use temporal idioms to discern (and present, in real time) the idioms used in a meeting. In this way, we can make meeting monitors, which show who is involved in what kind of idiom, and for how much time. This kind of
information can help identify “bottlenecks” or “rat-holes” in a meeting—portions of a meeting that are dominated by a small clique, frequently discussing a sub-topic of little general interest.

Meeting monitoring can also identify people who are not involved. However, there exists a problem in this approach. People typically do not want to be monitored. Such monitoring will only be accepted if the meeting participants consider it a form of assistance.

One of Jabber-2’s temporal idiom views of a meeting is shown in Figure 4. Each idiom is represented as a vertical bar. The vertical bar is divided (and colored) by meeting participant. Each participant has an individual color, as indicated by the legend at the bottom of Figure 4. Thus, as the meeting progresses, the relative amount of time spent in each idiom, and the relative amount that each participant contributes, is graphically displayed and dynamically updated.

Our approach to temporal idioms is typical of our approach to all of Jabber-2’s user interface: every type of information should be available for browsing or querying, in real time (while the meeting is taking place) and for post-facto retrieval. Thus, temporal idiom information is also available in the time-line view, as shown in Figure 5.

In this view, the temporal idioms are shown in a single line running across the bottom of the time-line pane. Each idiom is assigned a different color, and the span of the color segment indicates the portion of the meeting covered by that idiom: question-and-answer is green, presentation is purple, discussion is blue, and so on.
5.2. Blurring Querying and Browsing

Not only do we support both querying and browsing in Jabber-2, but in a heuristic evaluation study of our user interface, we have determined that users have lower retrieval accuracy, higher retrieval times, and lower satisfaction when restricted to using either browsing or searching alone. When they are allowed to use both techniques, their efficiency, accuracy, and satisfaction improved substantially. With this insight in mind, we have been building the Jabber-2 user interface to support both browsing and searching, and to make it as seamless as possible for the user to switch between these two modes.

The user can always browse the time-line directly, clicking on recognized words, seeing and hearing the stored record at that time location. Similarly, the result of a query is to take a user to the point (or points) on the time-line where the search result was found, thus engaging the user in a browsing task.

Furthermore, the Jabber-2 user interface supports the ability to zoom in or out on the time-line view. Consider the difference between Figures 3 and 5, for example. In Figure 3, the time-line is zoomed to a level where individual recognized words can be read. In Figure 5, individual words can not be read, but patterns of activity or areas where a search returned large numbers of results (as indicated by colored buttons) can more easily be detected. By allowing the user to zoom out, we are transferring the burden of browsing and searching from a cognitive task to a pattern-matching task.

5.3. Themes

As already mentioned, one of the most powerful means of supporting a user’s information seeking needs is to free the user from the bonds of keyword-based retrieval. Thus, one of our main foci in this research has been to support topic-based retrieval.

To give a concrete example of how this works, consider the following set of sentences:

1. With a diet consisting of meat in addition to berries, the family ursidae belongs to the order carnivora.

2. The Brown Bear ranges across the coniferous forests of North America, even into the territory of the closely related Alaskan Brown Bear.

3. The market value of common metals, including copper and zinc, is continuing to drop due to a prolonged bear market.

4. If you can bear to hold on q longer, don’t sell your shares yet, as the nickel market is expected to turn from bear to bull.

5. The penny coin is our smallest unit of currency, equal to 1/100th of a dollar. The next most valuable coin is the nickel.

Suppose you issue the query bear, wanting to retrieve sentences relevant to the animal sense of bear. A keyword search will correctly hit the second sentence, incorrectly hit the third and fourth, and incorrectly miss the first. Similarly, a query for nickel (in the sense of the metal) would produce false hits and false misses. For example, the third sentence ought to be returned for this query, since it is discussing common metals, even though it doesn’t actually use the word nickel.

We can support these types of queries in Jabber-2, however, through the use of our concept clustering algorithms. We can also use the notion of concept similarity to suggest ways of reformulating queries. Using this technique, we can (but do not currently) suggest reasonable alternatives in the face of an empty hit list. (e.g. “I can't find any mention of iron, but perhaps the metals copper, zinc, and nickel may be of interest.”)

To support this functionality we need to identify the sense of a word as it appears in the original input and relate it to similar words. In the five example sentences, we would like to see the following senses grouped together.

Brown Bear, Alaskan Brown Bear, ursidae

metal, nickel, zinc, copper

nickel, penny, coin, currency

bear, bull, market

We want to distinguish between the stock market senses of bear and bull and the animal senses of these words. We want to distinguish between nickel as currency and nickel as a metal.

Here are the themes we get after running the example sentences through Jabber-2’s concept clustering function:

1. currency: penny, dollar, nickel, coin, currency
2. metal: nickel, copper, zinc, metal
3. brown_bear: alaskan_brown_bear, brown_bear
4. investors: bull, bear
5. activity: market, share, turn, hold

Notice that we are not only able to cluster related concepts, but also attach a meaningful label to each. These labels summarize the constituent words and indicate themes present in the original source.

Note also that we are able to disambiguate different senses of the same word. We distinguish between the metal and currency senses of nickel, and do not consider bear to be an animal. Both bear and bull are properly cat-
5.4. Query Synergy

Any single multimedia indexing technique is likely to be flawed (as, for example, our temporal idiom recognition only gives us 70-90% accuracy) for any given information retrieval requirement. Similarly, the best speech recognition systems can only claim about 95% accuracy.

This gives us what might be inadequate accuracy if each technique was used by itself. However, combining heterogeneous techniques can ameliorate these flaws. Each indexing technique offers different strengths, and using them together allows a user to express queries that could not be expressed via any single technique. We must not only give users many “open doors”, but we must allow a user to combine these opportunistically.

The foundation of our indexing, however, rests in the theme-based browsing and querying. This kind of retrieval, by itself, ameliorates the deficiencies of speech recognition because it indexes by themes, and themes are inherently redundant. So, relatively low speech recognition accuracy may be withstood while still providing useful indexing for the user. Since we unify querying and browsing in Jabber-2, our objective in responding to queries is to get the user to the general neighborhood of their request, on the assumption that, once there, they will browse around. Also, whether querying or browsing, we make use of a human’s innate pattern recognition abilities to guide the user to “interesting” parts of the stored information.

5.5. Relationships with UI Design Principles

We need to examine the relationship between Jabber-2, as a research prototype system, and the principles of user interface design described in Section 4. Let us first recap the design principles:

- Center the application about the user.
- Make the interface visible.
- Provide a unifying theme.
- Make object interaction direct.
- Provide many open doors.

We center the application around the user, provide a unifying theme, and make the interface visible through the time-line view. This view unifies the various streams of multimedia data, and allows the user to access and replay these streams. The time-line view is the basis for both a browsing and a querying interface. Object interaction is direct in the sense that all buttons on the time-line are “alive”. Finally, we provide many open doors in the senses that we provide complementary indexing techniques, and the we provide many views onto the indexed information.

6. Future Directions

Our future work with Jabber-2 will strike out in several directions. We need to undertake more and more complete user studies, to evaluate how users capitalize on the various retrieval and indexing strategies that we have provided. Thus far we have completed several studies, both formal and heuristic (e.g. [5], [6]), but as the Jabber-2 system evolves, further study is needed to evaluate (what are believed to be) technological improvements.

We would also like to integrate more powerful speech recognition capabilities. We have been using an off-the-shelf speech recognition engine, ICSS (IBM Continuous Speech System) but its restricted capabilities are currently limiting the size and scope of meetings that we can process. Speech recognition technology is maturing rapidly, and we believe that this direction will be a simple progression for us.

Although Jabber-2 has all of the machinery to support query reformulation, we have not yet implemented it yet. We believe that the results of a query can always be used to form more queries:

- by reusing individual words (by selecting them from the query result in some fashion);
- by asking our concept clustering software (using WordNet) to find more clusters of words “like” the
by using WordNet to suggest related search words in the event of a miss.

Finally, we recognize that our zoom capabilities for the time-line view could be improved through the use of fisheye views—areas of distorted magnification on the time-line, on analogy with the way that a fisheye lens on a camera magnifies objects in the center of the view and reduces objects toward the periphery.

The idea is that a time-line has multiple “fish-eye” points of focus, one per “hit” (query result). Each fish-eye focus point contains the word which matched the query at a large magnification, along with a small amount of surrounding context. We show the context by using decreasing font sizes for the surrounding recognized words. Any of these matching words are, of course, selectable, since we are just popping the user into a special non-linear “browsing” view of the time-line. This will aid further in the unification of browsing and searching that is one of the prime objectives of the Jabber project.

7. References


