Int J Digit Libr (2012) 12:89–103
DOI 10.1007/s00799-012-0090-3

A spatial hypertext-based, personal digital library for capturing and organizing musical moments

David Bainbridge · Brook J. Novak · Sally Jo Cunningham

Published online: 14 April 2012

Abstract We describe the design, development, and evaluation of a personal digital music library application designed to assist musicians in capturing, developing, and managing their musical ideas over time. The target user group is musicians who primarily use audio and text for composition and arrangement, rather than with formal music notation. The software design was guided by a formative user study which suggested five requirements for the software to support: capturing, overdubbing, developing, archiving, and organizing. This led to a spatial hypermedia approach forming the basis for the developed application. Furthermore, the underlying spatial data-model was exploited to give raw audio compositions a hierarchical structure, and—to aid musicians in retrieving previous ideas—a search facility was provided to support both query by humming and text-based queries. A user evaluation of the implemented environment indicated that the target musicians would find the hypermedia environment useful for capturing and managing their moments of musical creativity. More specifically, they would make use of the query by humming facility and the hierarchical track organization, but not the overdubbing facility as implemented.

Electronic supplementary material The online version of this article (doi:10.1007/s00799-012-0090-3) contains supplementary material, which is available to authorized users.

D. Bainbridge (✉) · B. J. Novak · S. J. Cunningham
Department of Computer Science, University of Waikato, Hamilton, New Zealand
e-mail: davidb@cs.waikato.ac.nz

B. J. Novak
e-mail: bjn8@cs.waikato.ac.nz

S. J. Cunningham
e-mail: sallyjo@cs.waikato.ac.nz

Keywords Music composition · User-centered design · Spatial hypermedia · Personal digital music library

1 Introduction

Musicians come up with ideas for songs in many settings. The heavy metal band Deep Purple got their inspiration for *Smoke On The Water* when the Montreux Casino was accidentally burnt down from a flare gun fired off inside the premises during a Frank Zappa and The Mothers of Invention concert. The band saw the smoke from the fire spreading across Lake Geneva from the relative safety of their hotel room, which in effect inspired the lyrics of their work-in-progress [30]. Inspiration for The Bee Gees' *Jive Talkin'* came from the "chunka-chunka-chunka" sound of a car rolling over a bridge crossing Biscayne Bay near Miami.[1]

These examples illustrate that many musical ideas are spontaneous and can occur in any setting.

In this article, we describe APOLLO, a software environment designed to help composers/performers of popular music:

Capture musical ideas in a convenient form, as and where they occur;
Enhance and embellish the idea; and
Organize the ideas over time into an archive.

Based on the results of a diary study, a spatial hypermedia paradigm [2,18] was selected as the framework within which to develop the software environment. Looking ahead, Figs. 2 and 3 show two snapshots of the developed prototype, with its interface and functionality explained in more

[1] Refer to www.songfacts.com/detail.php?id=1793/ for more details.

detail in Sect. 4. In brief, it provides a hierarchically structured set of freeform canvases for creating and manipulating musical ideas (both text and audio recording), augmented with support for searching the hierarchy with either melodic or text-based queries. Time for musical objects on the canvas is represented horizontally, left to right.

The target users are musicians who use music composition software designed for non-professional use, such as Apple's GarageBand. The aim of the project was to establish whether the spatial hypermedia form of interaction could enrich a user's experience over conventional support for composition.

The work presented here builds upon and expands the details provided in [5], and is structured as follows. First, we present the results of a diary study designed to establish how, when, and where musical ideas occur for the target group of users, and what their computer needs are as the ideas are developed over time (Sect. 2). This led to the decision to explore how suitable a spatial hypermedia approach was to support musicians in this activity, and in particular, determine if this approach has any benefits over conventional style tracking editing applications.

In Sect. 3, we review related work, including both commercial and research systems that aid composition. This is followed by details of the software design (Sect. 4). To see whether musicians might prefer this new environment, we describe a formative user study that required participants to complete a range of musical activities using GarageBand and, task-for-task, using the new spatially hierarchical environment (Sect. 5). We conclude with a summary of our findings.

2 Diary study

A qualitative diary study was performed to arrive at a deeper understanding of when, where, and in what settings musical ideas occur. Diary studies are particularly well suited to the exploration of "little experiences of everyday life" [28], those events that occur too infrequently and/or spontaneously to be difficult to reproduce in a formal laboratory-setting—and certainly the lightening strike of musical inspiration fits into this description, as exemplified in the introduction.

In a diary study, participants literally keep a record (conventionally, in a paper diary) of occurrences of the experience under study. These records are then shared with the researcher, usually at the end of the diary period, and usually in a de-briefing session that allows the participant to explore the events with the researcher. The primary advantage of the diary technique is that the diary—when filled in conscientiously by the participant—provides a more faithful description of the activity than one which may be obtained by retrospective methods such as post-hoc interviews [7].

An initial study with three subjects over three days was undertaken as a pilot, before scaling up to a longer-term diary study lasting two weeks with six participants. During this time, participants recorded their ideas in a paper notebook and/or audio recorder (an MP3-player-based dictaphone or mobile phone). Participants were free to choose the form of recording device they were most comfortable with.

The notebook was specifically designed for the experiment. Comfortably fitting in a back pocket, it came with a colored design on the front and hard-backing—this care of attention to detail and quality was done in accordance with established HCI practice, to send the participants the message that the study was of importance. Inside the notebook was a mixture of blank "ideas" pages and printed blank staves, with the former in more evidence.

Participants were instructed to take the diary and voice recorder with them everywhere they went for the duration of the study. Whenever an idea came to them, they had to record it using the notebook and/or the voice recorder straightaway.

The initial trial had used only blank ideas pages, so as not to bias the participant into thinking music notation was required; however, in a post-pilot study debriefing, one subject noted that instead of having to draw their own, "it would have been good if the ideas' pages had printed staves." The notebook design was adjusted accordingly.

All the participants could play at least two instruments, all had band experience, and most were studying or had studied a music-related degree. All of the participants had experience with computers, all used Internet-based and music editing software and the majority used office-suites. Participants spent between 10–28 h a week using a computer, averaging 15.25 h.

The source data (pre- and post-questionnaires, notebooks, and recordings) were analyzed along with interviews with each subject. Background questions were posed to establish their musical background: e.g., "what instruments can you play, and which do you specialize in?" and "do you usually write your own music?" Others probed the issue of capturing musical ideas: e.g., "do you usually record musical ideas," "if yes, in what form does a musical idea generally come to you?" and "would you consider the notepad and recorder a good way for recording idea? (why/why not)."

A total of 60 ideas were captured during the study period: 31 on the dictaphones, and 29 in the notebooks (Tables 1, 2). One participant explicitly linked four paper and audio diary entries—the dictaphone recording a guitar performance of

Table 1 Summary of audio diary entries

	No. of entries	Min. length	Max. length	Average
Audio diary	31	6 s	469 s	70 s

Table 2 Summary of written diary entries

	No. of entries	Lyrics	Chords	Written notation	Comments
Written diary	29	19 entries	13 entries	8 entries	9 entries

an idea, and the corresponding notebook entries consisting of written chords. The average time of an audio recording was 70 s, excluding two outliers that lasted over 5 min (Table 1). All but one of the ideas recorded by the dictaphone involved a guitar (for playing the chords).

Figure 1 presents a sample of a written diary entry. Note that the participant apparently expresses meaning through spatial placement and symbols (in this case, an underline, a box, and an asterisk). As discussed in Sect. 5, idiosyncratic note-taking styles are common in other domains [16]; this diary study provides evidence that styles for scribing musical ideas are similarly personal and free-form. Written diary entries included a range of types of music information:

- lyrics: lyric entries ranged from fragments (e.g., "I'm a contradiction") to extensive entries (the longest was two diary pages, completely filled).
- chords: these entries were brief chord progressions.
- written notation: melodies were scribed most frequently not only in conventional music notation, but also in guitar tablature or a personal notation.
- notes and comments: text comments can set a mood ("a jazzy feeling chord progression"), serve as a reminder of the circumstances evoking the idea ("Have just got back from Christchurch & have seen 2 Jazz bands play!"), and so forth. Some of the comments are comprehensible only to the author (e.g., "– appealing – misleading").

Generally the participants felt that a dictaphone and notebook were sufficient for capturing their ideas, with two caveats:

- Several participants noted the dictaphone lacked the ability to overdub—to make a new recording that supplements a previous recording.
- Two participants noted the lack of support for capturing the source of the idea's inspiration—such as could be done through image or video capture—making it difficult to recall the thoughts behind what they were trying to express when they later came back to it.

Thirty-six of the 60 audio and paper diary entries included an indication of the participant's physical location when musical inspiration struck. Thirty-two of those ideas came to the subjects when at home, although it is noted here that this may be because subjects occasionally forgot to bring the notebook with them when they were out and about. This was revealed (obliquely) by one participant commenting on the fact that "using my mobile phone was really good because I always had it on me, as opposed to having to remember to bring an extra device/notebook with me wherever I went." One thing the interviews did establish is that, no matter where the inspiration originally struck, the ideas were further developed at home using a desktop machine or laptop using track-based music production software.

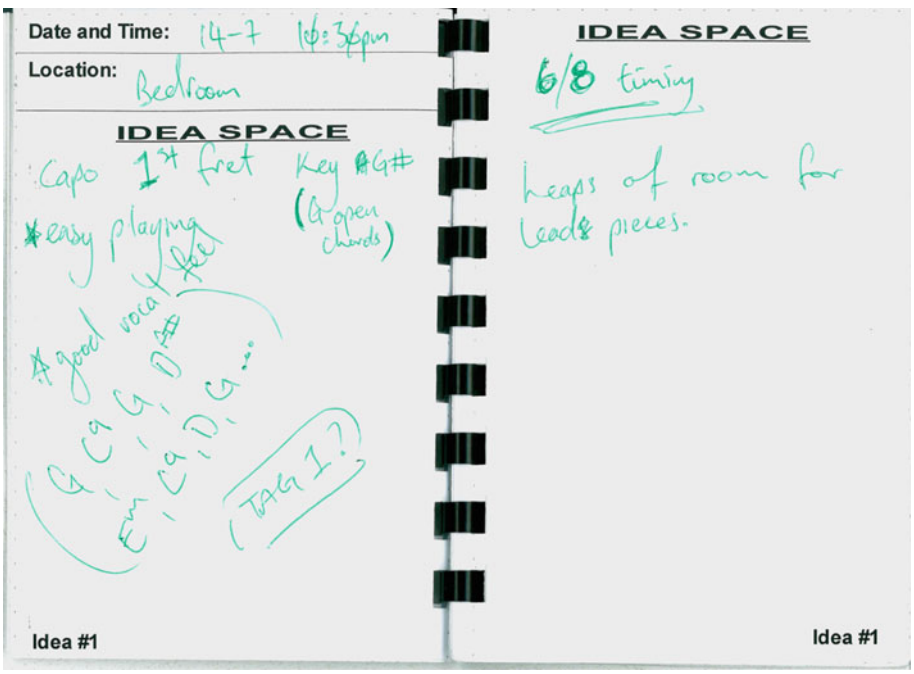

Fig. 1 Example entry from the diary study

Furthermore, the interviews highlighted that

- For capturing ideas "in the wild," there are already many solutions widely available for handheld devices, such as the "LG Real Groovy" mobile phone, which comes preloaded with a suite of music recording applications such as a MIDI keyboard, score editor, and an audio recorder. Even richer graphical applications are becoming available for the iPhone.
- Ideas can consist of multiple pieces of information, such as raw audio, text, and images combined. Therefore, support for directly associating, or grouping different types of media together is needed.
- New ideas strike before previous ideas can be fully realized—and one musical idea can sometimes itself spark another.

Based on the first observation, it was decided to concentrate software development in this project on support for working with musical ideas once they are transferred to the musician's home computer. The second observation underlined the importance of supporting musicians in engaging in their natural idea scripting behavior, rather than forcing a rigid format for information entry. The final observation illustrated that a musician can have many potential compositions on the go at any one time, and that these musical ideas typically occur (initially) in fragments, which are then combined, and/or shared across different emerging compositions—perhaps where only one version will emerge as the definitive work. These qualities suggested that a digital library approach to organizing and managing all of these pieces would work well, rather than assuming that a song is a single entity that can be developed in isolation. The nature of the different media forms involved suggested support for multimedia an important capability, where the ability to freely associate different elements to an idea refined the capability required to that of spatial hypermedia.

Our vision, then, is to support a bricolage style of music composition: creation characterized by combining, modifying, recombining, "fiddling with," extending, and mashing up smaller fragments of material [26]. The user is supported in jotting down these small fragments (musical ideas), quickly and easily shifting and re-arranging them, and in building up more or less complete music pieces. In the implementation, we emphasize speed of interaction, support for browsing and searching over a user's developing fragment collection, and above all an informal interaction design that accommodates idiosyncratic forms of composition—all fundamental to the bricolage style.

We conceive the resulting system, named APOLLO after the Greek god of music, as a personal digital library in which

- documents (musical ideas) can be provided in a variety of forms—audio, text snippets, images, and so forth. Documents can be modified, combined, or deleted by the user at any time.
- metadata (used primarily to support searching) is automatically extracted from the documents. Users include descriptions and annotations as a personal aid to the composition process, rather than as a requirement for efficient document processing.
- text searching is supported over text snippets entered by the user, and "query by humming" search [4] is provided for audio.
- documents are organized for browsing through spatial hypermedia. The user is free to arrange and re-arrange the documents in the space in any way that s/he feels will aid the composition process.

In other words, our aim became to combine spatial hypermedia and digital libraries' capabilities into an interactive environment suitable for a musician to capture their musical idea.

3 Related work

"The human mind... operates by association. With one item in its grasp, it snaps instantly to the next that is suggested by the association of thoughts, in accordance with some intricate web of trails carried by the cells of the brain" [8]. This concept, put forward by Vannevar Bush in 1945, is widely considered to be the origins of the field of hypertext [6]. The term itself was coined by Nelson 20 years later in 1965 [27].

Today the term is used loosely, as there are many accepted definitions of what a hypertext system is. One commonly accepted definition is "windows on a screen are associated with objects in a database, and links are provided between these objects, both visually and in the database" [9]. Hyper*media*—the form we are interested in for this project—is a generalization of hypertext, and in particular, we refer to the form used here as *spatial* hypermedia to emphasize the freeform layout capability that is at the center of the design.

We based our implementation of APOLLO on Expeditee, a modern Java redevelopment of the venerable KMS (Knowledge Management System) [2]. KMS was developed in the early 1980s as a hypertext system to support collaborative creation and communication of information. Expeditee, like its ancestor KMS, is a full-fledged spatial hypermedia system—it permits nearly any sort of "knowledge artifact" (text, images, software, presentations, documents, etc.) to be embedded in a 2D frame. The frames are connected by links, and the user can at any time switch between frames.

A scripting language is included in Expeditee; new functionality can be dynamically embedded in frames and dynamically executed. We expand upon this description in the next section, where we give an overview of the implemented personal music digital library environment.

There are existing software applications aimed at supporting music composition and exploration of musical ideas. In this section, we consider two such commercial systems (GarageBand and Sibelius) and four systems originating in the music research community (OpenMusic, CyberBand, QSketcher, and Mus*ink*).

The fundamental difference between APOLLO and these six systems does not simply lie in its mix of features—in fact, there is considerable similarity in the basic music creation functions in all these seven systems. Where a key difference does emerge is in how APOLLO adopts a generally less structured and prescriptive approach to recording and developing musical inspirations. This approach was inspired by the comments of a participant in the formative design diary study (Sect. 2), who described the development of musical ideas as "like little pieces of artwork."

As we explored this statement, an analogy emerged: with a painter and their notebook, many ideas are quickly sketched out, so why not look to achieve the equivalent experience for a musician? A notebook allows the painter to quickly transfer ideas to the page so the painter's thought processes are not lost. The sketches are rough, but conveyed in sufficient detail so that the artist can envisage exactly how the polished picture will look, even if they come back to it years afterward. The artist's sketches may not be what they want first time round—they might augment a sketch later, annotate it, or flip the page to take a different perspective. And, of course, a visual artist rarely works exclusively on a single piece; the sketchbook contains many ideas that might evolve into different finished works—or that might be abandoned, yet remain in the sketchbook. APOLLO is intended to serve as a musician's equivalent to a visual artist's sketchbook.

We first contrast APOLLO with GarageBand. This is a popular commercial track editing software package. Its data model is for representing one song at a time, and is composed of multiple tracks, laid out vertically as a sequential list. The horizontal axis represents the time-line of the song. The time-line can be scaled to get either a closer view of the audio or an overview of the tracks. The system supports a mix of MIDI and raw audio.

GarageBand aims to allow the user to produce full quality audio songs, at or approaching professional level. The analogy of the painter and sketchbook does not fit well with its data model because users are forced to work with a list structure rather than a blank canvas—the users must lay out the musical ideas sequentially, rather than being able to "play" by moving music snippets about freely. Further, the musical ideas must be entered with an attention to fine details of performance that is not appropriate to the capturing of nascent musical ideas.

Sibelius is another popular commercial product. Its primary intended use is for creating and editing musical scores; audio input is not directly supported, but realistic audio can be generated from a score. The process of transferring a musical idea from a musician's mind to Sibelius, then, involves recording it in formal music notation—a relatively laborious task for many amateur musicians, and virtually impossible for musicians who play by ear. The written scores cannot preserve tone information, which could be a significant aspect of an idea.

The artist's sketchbook allows the artist to annotate drawings, paste in photos and other images, jot down feelings and impressions, and so forth—the process of creating a painting can involve recording text and images that themselves do not necessarily appear in the finished work, but that are important to the artist in remembering and exploring an idea. GarageBand and Sibelius have limited facilities to allow the musician to jot down extra-musical material. Neither allows free-text notes or images to be added to the musical idea. Sibelius allows lyrics to be added to a song (GarageBand does not), but the lyrics are expected to be in final form—Sibelius is ill suited for working through multiple drafts of lyric fragments.

Another distinguishing characteristic of the artist's sketchbook is that it can contain ideas for many different pieces, and the artist can easily move between the records of different ideas/distinct artworks. GarageBand has no explicit link between its music files; each song has a separate workspace unrelated to others by that composer. Sibelius has an "Ideas Hub" feature which can be used to tag, store, and retrieve musical ideas (i.e., lyric text and melodies in written musical notation). The process of capturing an idea requires the users to first scribe the ideas in written notation, chord sequences and/or lyric text; then to bring up the Ideas Hub window; and then to drag and drop the idea into the window. A power-user can speed up the process by selecting an idea's data and typing a control key sequence. This model streamlines the capturing of the creative moments of a musician: the integrated archiving system does not require the users to deal with files (as is the case with GarageBand), and the users do not have to think about naming or organizing their ideas until a later time either (and again, GarageBand forces the user to think at this level from the first glimmerings of a musical idea). However, the Sibelius archiving system only stores written notations or lyric text. Further, users must fill out a form-like dialog to tag their ideas for them to be searchable. To find an idea, the user performs a keyword search to view the best matching ideas in the Ideas Hub window—a far cry from casually flipping through a sketchbook. As a whole, the idea archiving model here is very rigid: users must follow a specific sequence of steps to transfer their ideas

from the mind into their idea collection and then to organize them. With this model, musicians cannot organize their ideas idiosyncratically or spontaneously.

The Ideas Hub has one advantage over the paper sketchbook: once an idea is tagged, the musician can use Sibelius's search engine to locate it and to view related (i.e., similarly tagged) ideas. In practice, however, this feature is likely to be of limited use, as the system cannot search lyrics directly and the one- or two-word user-supplied tags are highly impoverished surrogates for complex musical thoughts.

We turn now to four non-commercial systems—OpenMusic, CyberBand, QSketcher, and Mus*ink*—developed in the research community. This is not intended as an exhaustive exploration of the software contributions of the computer music research world, but rather is an overview of the directions that this research has taken.

CyberBand [29], like APOLLO and GarageBand, is aimed at the amateur musician who might, or might not, be comfortable with formal musical notation. Users can work with either a formal score sheet or audio "blocks" of music. The system provides a catalog of "riffs" that can be used as building blocks for new music, and the user can add new riffs/blocks to the catalog—so that the catalog is the storage point for musical ideas under development, or ideas that can be re-used for different songs. These musical ideas are limited to playable music segments, however, as CyberBand does not support text or image annotations.

OpenMusic [3] is a music composition program implemented as an object-oriented, visual programming language based on CommonLisp. While the system includes a conventional music editor, the full power of OpenMusic is realized by using a visual programming editor to connect together ("patch", in the OpenMusic terminology) library or user-provided modules. Full utilization of the system thus demands the ability to work with formal music notation and to program. While the user may develop personal libraries of re-usable music components, viewed in terms of the sketchbook capability sought in this work these libraries share many of the shortcomings of Sibelius's Ideas Hub as a mechanism: the process for recording ideas is time-consuming and not straightforward (involving coding the idea as a "1patch"), and there is limited support for searching and browsing the set of ideas.

Mus*Ink* [25] and QSketcher [1] explicitly address aspects of sketching as an aid to the composition process. Mus*Ink* seeks to address the disjunct between the rich set of idiosyncratic annotations that composers make to musical scores, and the rigid score notation imposed by computer-supported composition software. Specifically, Mus*Ink* provides a bridge between paper composition and OpenMusic; using a digital pen and Anoto technology, the user can make paper annotations that are reflected back into the OpenMusic digital representation of a musical piece. While Mus*Ink* does allow greater freedom than OpenMusic in defining personal gestures/annotations, this capability comes at the cost of having to formally add semantic definitions to gestures (supported by a Gesture Browser) so that the annotations can be recognized by OpenMusic. This is a powerful and flexible mechanism, but again requires expertise in both programming and use of formal music notation—and so is not intended to be a lightweight mechanism for capturing musical ideas.

QSketcher is the closest in spirit to APOLLO of the systems discussed here. Intended as a tool for scoring films, the emphasis is on removing barriers to inputting ideas, no matter how fragmentary or in whatever form the user wishes to represent it ("as graphical sketches or scribbles, textual annotations, music played on a keyboard, and so on" [1]). The Idea Space for QSketcher is a close analog to the music sketchbook idea, as a lightweight mechanism for recording musical inspiration in the "natural" form for that idea, for that user. In comparing QSketcher to the needs of users targeted in this project, despite its name, the QSketcher interface is more rigid and complex than what is sought for the sketchpad capability. It is also highly tailored to the domain of film production.

The role of the time-line in APOLLO is worthy of more detailed consideration. Its primary purpose is as an aid to the user in coordinating several tracks in a single frame, and a time-line is local to that point in the hierarchy. The user has direct control of playback (visualized by progression across the time-line). Its main uses are for overdubbing and to support visualization of tracks.

This temporal management differs from that in other spatial hypermedia systems incorporating time (e.g., HyperCafe [23]), in which the user may have little or no control over the expression of time-based media embedded in the hyperspatial system, and in which the time-line equivalent may extend over several locales. Indeed, APOLLO'S representation of time is much simpler than that found in full-fledged temporal hypermedia systems, which are based around a master time-line (sometimes referred to as a score or script). The time-line supports temporal ordering in the user's movements between different document components (for example, frames/pages) or between portions of the same document component. A component itself can be time based (e.g., a video or audio clip), with the hyperlink potentially including only a portion of the component [14]. More complex temporal relationships such as synchronous display of multiple time-based components are also possible [15]. These complex temporal facilities are useful in supporting such diverse activities as maintaining edit histories [15], hypervideo development [23], and hypermedia narratives [23]. Given APOLLO'S tight focus on supporting music creation, these more general temporal facilities appear extraneous to the task, and so are not included in the APOLLO architecture.

4 Implementation

Figure 2 shows a snapshot of the implemented personal music digital library environment. As much as possible, it was developed within the existing capabilities of Expeditee, utilizing its dynamic capabilities (such as agents and actions) associated with media elements, accompanied with a pre-populated set of frames. In this way, APOLLO is like applying a skin for Expeditee, in the same way the look and feel of a media player can be changed by applying a skin.

As the time of implementation, Expeditee itself was a work in progress, and so inevitably some software development was done on the core Expeditee system also. We return to this point later in this section, when we discuss the innovation of light-weight and heavy-weight widgets and the resulting change needed in the canvas repainting algorithm.

The scenario depicted in Fig. 2 is a musician developing their own arrangement of the Michael Jackson hit, *Billie Jean*. It could equally be a tune of their own composition. In the example they have laid in two guitar tracks (upper half of the frame), overdubbed some special effects (just below the mid-line), and entered the lyrics to the chorus as text (at the bottom). A time-line runs along the bottom for controlling when the audio tracks are played.

Drawing upon the capability provided by Expeditee, in its rawest form, the implemented system consists of a large blank canvas (a frame) upon which objects (text, scalable vector graphics, and images) can be placed—wherever the mouse cursor is at the time. For this work, the set of basic media types supported was extended to include audio (see Sect. 4.1 below).

A layering mechanism is provided to allow objects to overlap. Any object can be hyperlinked to a new blank frame, thereby creating a hierarchy of frames. Clicking on the hyperlinked object takes the user to the new frame. When at the new frame, clicking on "white-space" (an empty area of the frame) takes the user back. An object can also (and/or) have an action associated with it, triggering a particular response when clicked upon. No explicit saving is needed; this happens automatically when the user transitions from one frame to another. Indeed, the user is never prompted for a filename to use: content is accessed through browsing the hyperlink structure or through searching over the contents of a set of frames.

Leveraging from its generic hypermedia data-model, APOLLO is primed to launch with an initial frame set that provides, for the musician, built-in online help and functionality for playing and recording audio as well as searching. This functionality is evident in Fig. 2 along the top and right-hand side of the interface. It is implemented as an overlay, and so is persistent from frame to frame.

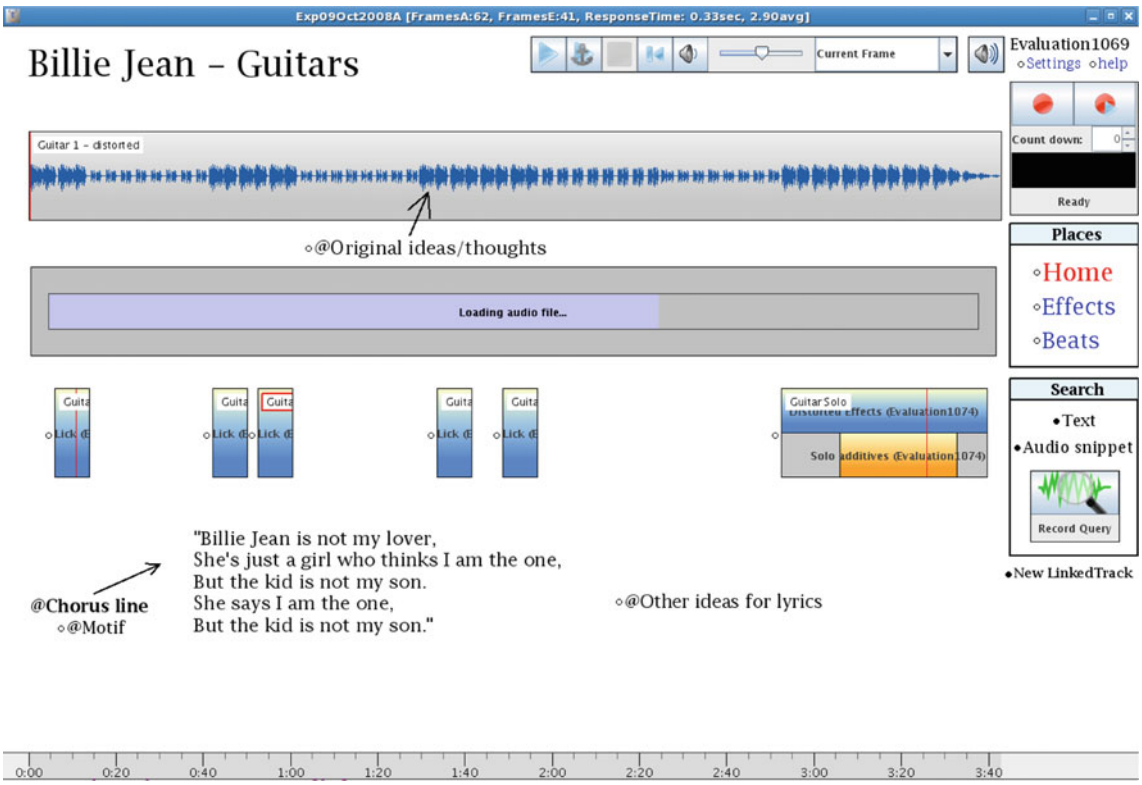

Fig. 2 Overview of APOLLO

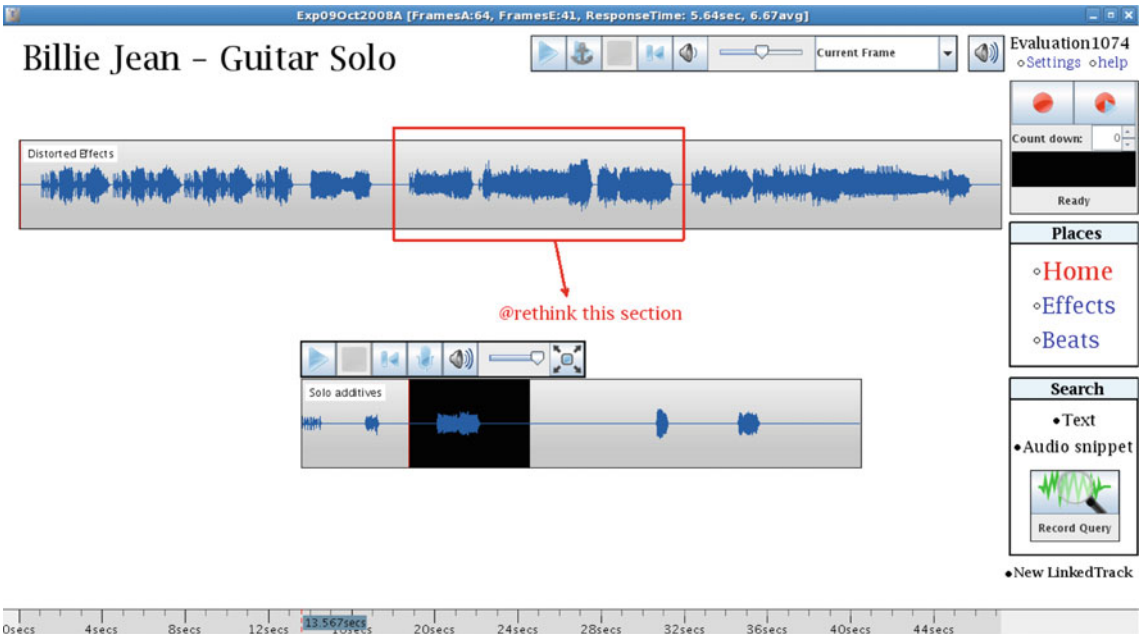

Fig. 3 An example sub-frame in APOLLO

External files, such as images and audio, can be dragged and dropped on the frame, and turn into objects on the frame at that point. This is how the two audio tracks were laid in, with the x-axis to a frame used to represent time in seconds. Text is simply typed in.

Live audio recording is achieved by placing a suitable icon in the overlay (the often seen centered red circle used to signify a record button was used) and an action associated with it. When clicked, the result is for a recording widget to appear attached to the mouse cursor that the musician then moves to the location they want to use it. After making the recording, the widget turns into a visualization of the audio recording.

Unlike our most common experience of hypertext and hypermedia—the World Wide Web, there is no distinction between viewing information and editing it. Any object displayed on the frame can (at any point in time) be manipulated, for instance, changing its content or position, or possibly deleting it—and this applies to the primed "decorations" on the overlay around the edge. Don't like the provided button in a given place? . . . then click to the overlay frame and move it! Even the overlay functionality can be adjusted, and new functionality introduced through new buttons.

Sections of text can be highlighted by dragging the mouse cursor over them. A full complement of cut, copy, and paste is supported. Furthermore, an analogous set of operations is available for audio. Sections of the audio can be ranged out and cut from the track, or copied and spliced into another track, or used to start a new track. The set of short overdub audio tracks (located just above and to the left of the chorus text in Fig. 2) are examples of this, where the initial recording of an aspirated sound effect of "chick-aah" has been copied and then replicated a four further times.

Moving objects around includes taking them to another frame. Returning to the example in Fig. 2, the musician has chosen to group the overdubbed special effects (guitar solo) positioned to the right as a separate frame. In this case, an overview of this sub-frame appears in the parent frame, showing itself being composed of two audio tracks. Clicking on this overview takes the musician to the relevant sub-frame (Fig. 3). Recall that clicking in white-space will bring them back to the parent frame. In the case of text, a hyperlink to a child frame is indicated by a small circle drawn to the left of the text. Figure 3 also shows they have used the scalable vector graphics capability to annotate the frame (e.g., "rethink this section").

Creating frames and sub-frames is a natural process, and when used over a period of time, the hierarchical structure quickly grows. Indeed Fig. 2, for example, is not the top-level frame (as first implied in our description above), but is part of a larger example consisting of 140 frames (excluding in-built help frames) generated over a series of days. To help quickly locate frames, search facilities for both text and audio queries are provided, again implemented utilizing the existing capabilities of the spatial hypermedia environment.

On initiating a text query, the result is a new (system-generated) frame appearing with the search results displayed in a linear list. Each item is encoded as a text object that displays the title of, and is hyperlinked to, the matching frame. Query by humming is accomplished the same way; only the recording widget is used to input the query before the search

is initiated. Conversion of raw audio to symbolic form, and the subsequent symbolic matching is based on the algorithm presented in [4]. The system-generated frame of results is linked to the existing hierarchy. Thus, not only it is a form of query history provided, but also (like any other frame) it can have its objects manipulated, moved, and annotated—and of course is included in the search space for subsequent queries.

4.1 Light-weight and heavy-weight interactive widgets

For this task, the provided set of media types supported by Expeditee was extended to include audio. This in turn led to the development of the afore-mentioned light-weight and heavy-weight interactive "widgets." In essence, they are a way of marrying (typically) small pre-packaged Java Swing components with Expeditee's native methods for interacting with lines, rectangles, text, and other scalable vector graphics. Inspired by the approach pioneered by Unix, they form building blocks that can be combined on the canvas in Expeditee to implement more complex entities. Their usefulness goes beyond support for audio in Expeditee, but discussion of this is outside the scope of this article.

To seamlessly embed Swing components in Expeditee, there were many issues to resolve. Central to the approach taken was the decision to splice, at the programming level, the base widget class (essentially a JComponent) with Expeditee's fundamental classes for representing dots and lines, which in turn are used for representing rectangles and polygon shapes in general. This not only simplified the task—since it allowed it to be accomplished through an extension rather than a re-implementation—but it also brought usability benefits because users would not have to learn a new set of actions and contexts for a new item.

Figure 4 shows one such example of a widget implemented using the new technique: a metronome. The widget is shown in "enclosed mode" as the cursor was inside the bounds of the widget when the snapshot was taken, just as occurs for a rectangular object drawn in Expeditee. The same visual indications are inherited: going inside the rectangular area results in displaying four filled red rectangles at the corners of the widget. The same actions are inherited too: when the middle button is clicked, all the items within the widget bounds (e.g., the arrow and the "@Using 4 bpm is good for this song!" item) would be picked up. Clicking in a corner allows the object to be resized, and so on.

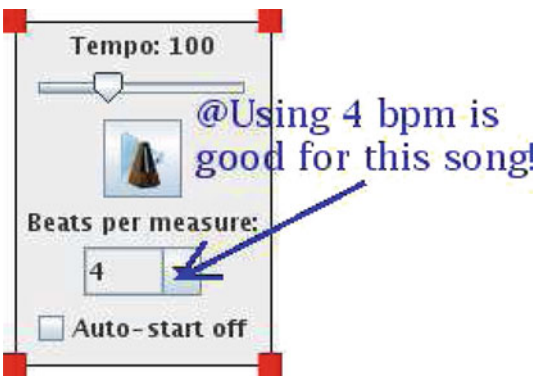

Fig. 4 A light-weight interactive widget used in APOLLO with an annotation

There were many cases that had to be dealt with specially for interactive widgets. For example, disconnecting borders (made up of dots and lines) is an entirely valid operation for a rectangle, but in the context of a widget, this would leave the object in an unstable state; consequently, this action was disallowed. To allow annotations to be placed over a widget, the layering algorithm was refactored. The overall rendering algorithm for a frame was also upgraded from the existing, simplistic approach that would redraw the entire frame any time an item was changed, to one that tracked "damaged areas" and only redrew those when needed. Finally, widgets needed to be persistent: the information present within a widget as a result of interacting with it needed to be retained, and so a mechanism was introduced that allowed the designer of a widget to declare which data fields were needed to keep state. This information is then used (by the software) to determine which data needed to be written to disk.

As noted by Akscyn et al., responsiveness to user interaction is an important requirement in a spatial hypertext environment [2]. Through user studies, they observed that although the average time a user spends at a frame will usually be many seconds, this is typically in the form of pauses followed by rapid bursts of interaction, including navigation to other frames. In the context of this project, the loading of audio tracks (if implemented naively) ran contrary to this requirement. Consequently in APOLLO, this type of object is loaded asynchronously. An example of this can be seen in Fig. 2 where the second of the two main guitar tracks is still loading. Despite this, all the objects on the frame are live, and can be interacted with, including hyperlinks, to move to a new frame.

The feature added to Expeditee that provides this functionality was called heavy-weight widgets. Its implementation is built on top of the light-weight widgets, principally extending it with a globally shared manager, running in its own thread, which handles the loading and saving of large data files. At its core is a queue of files to load and save, tagged with which frame each operation is associated with. Over time, based on the user's operations, the manager can modify the queue to optimize performance. For example, in the case where a user clicks to a frame with some audio tracks (causing these tracks to be put in the queue for loading) but then (with barely a pause) clicks on to a subframe, then these files can be deleted from the queue. Even if one of the files is

partway through being loaded in, the manager has the ability to abort the operation.

4.2 Idea retrieval

As a musician's ideas collection grows, it becomes more difficult for them to locate specific ideas, especially in a large hypertext space environment [21]. It is important for a musician to recover an idea so they can develop it further, or use it for a source of further inspiration. Expeditee already had an in-built agent for providing full text searching. A text search suffices for remembering textual information about an idea, for example a lyric or a name of a song. However, it may not be always the case that a musician can remember textual information: they may only remember a certain snippet of sound, for example a melody line. To help musicians locate their ideas from a melody, a melody search agent was developed for APOLLO.

To search for an idea from a melody, users can press the search button on the melody search widget shown in Fig.5a (which is ever present on the overlay to APOLLO). When a user presses the query button the widget begins recording audio, where a user sings/hums a melody line. When a user is finished recording their melody and is ready to commence the search, they press the "go" button, as shown in Fig. 5b.

At this point, the melody search agent begins, and the current frame changes to a search results frame—this is Expeditee's convention that is used for all search agents. When a search completes, a list of results are displayed, ranked from best matched to worst. Each result is linked to the frame for which the matching tracks reside. Figure 6 displays example results from a melody search; the results are explicitly numbered to clearly convey the rankings. Each result contains the frame-name of the matching track audio widget's parent, along with the track's actual name.

To provide the melody-matching capability, the MELDEX melody-matching algorithm [4] was repurposed to fit the spatial hypermedia environment. Figure 7 presents the logic behind integrating the melody search algorithm into APOLLO. The search takes a query (sung/hummed by a user)

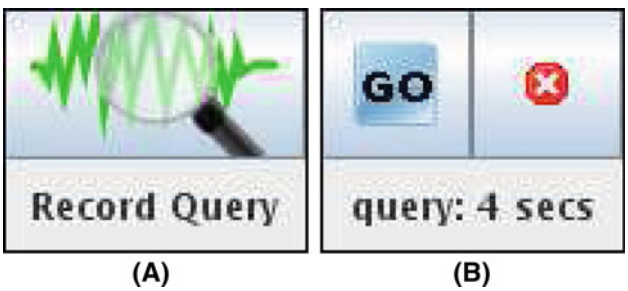

Fig. 5 Using melody search: **a** Initiating the audio recording; **b** Initiating the melody match

- 1: Evaluation 49 (Melody)
- 2: Evaluation 54 (Vox)
- 3: Evaluation 71 (VOX)
- 4: Evaluation 12 (Fretless bass)
- 5: Evaluation 30 (English horn part 1)
- 6: Evaluation 80 (Bass)
- 7: Evaluation 29 (Saxiphone part 1)
- 8: Evaluation 73 (Light part 2)
- 9: Evaluation 92 (deep-purple - smoke c
- 10: Evaluation 39 (Drums 1)
- 11: Evaluation 69 (Guitar 1 - Distorted)
- 12: Evaluation 86 (Abba - Waterloo)
- 13: Evaluation 91 (Staying Alive)

Fig. 6 The result of a music search

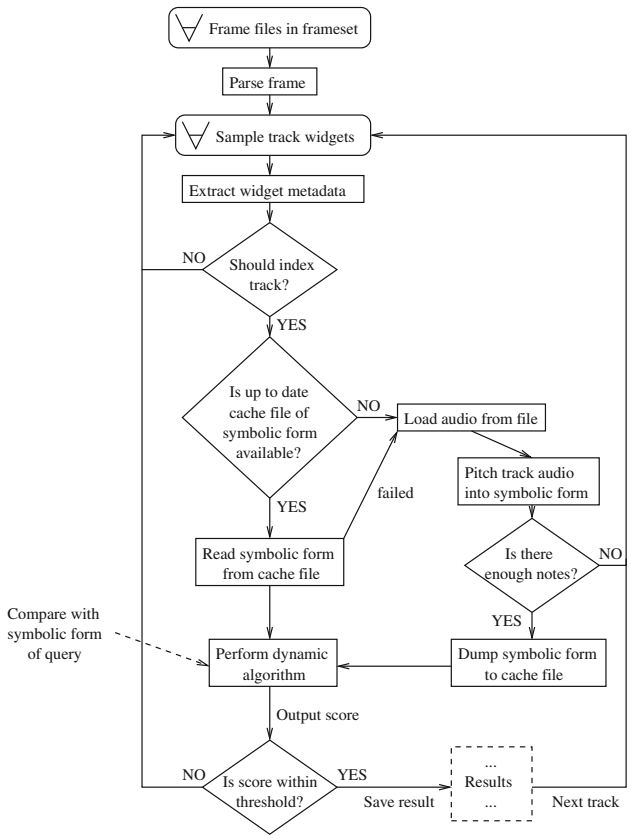

Fig. 7 Melody searching algorithm in APOLLO

and uses a monophonic pitch tracking algorithm to create a symbolic form of the audio (i.e., musical notes and rests). Next it searches all track widgets in the user's frameset. For each track widget, it compares their symbolic representations using a (minimum edit distance) dynamic programming algorithm. The dynamic algorithm calculates a "score" which describes how well the two melodies match. The score

together with the track's metadata are recorded in a results collection. Caching the symbolic representations was used to help speed up the process for future searches. Once the process is finished (in Fig. 7), the results collection is sorted by score (from best matched to worse) and displayed, as shown in Fig. 6.

With melody matching only being monophonic-to-monophonic, tracks containing polyphonic audio and percussive—for example, drums or distorted guitars—are potentially problematic. To compensate for this, a feature was implemented that allowed the user to mark audio tracks that were to be excluded from the search. This is accomplished by pressing CTRL+I, resulting in a visual "ignored" indicator in the top right of a track widget. This feature proved to be particularly adaptable, although admittedly placing a higher level of onus on the user. For instance, if a user wants their ideas to be indexed for melody searching but all the recordings are polyphonic tracks, then they could choose to sing a motif to represent each melody line. The relevant motif could reside in the frame that captures the idea (with mute selected so it does not play with the rest of the tune), or in a separate frame which links to the idea frame to save space. When the user conducts a melody query it searches across the monophonic motifs, but when a song is played back the full polyphonic version is heard.

5 User evaluation

Centered around this implementation, the answers to four key questions were sought:

1. Would a musician prefer to capture musical moments in APOLLO or conventional track editing systems?
2. Would a musician prefer a hypermedia environment or a file system for archiving and organizing musical ideas?
3. How much more usable and natural would musicians find the hierarchical model for laying out tracks in comparison with the conventional model?
4. Would a musician prefer searching for their ideas within a hypermedia environment or using a file system's search facilities?

The evaluation study was split into three real-world scenarios. For each scenario, the participant would play the role of the musician, and would complete the same tasks in both APOLLO and GarageBand. As part of the role-playing, participants were asked to imagine that they usually would use APOLLO/GarageBand for completing such tasks. The tasks were broken down into a series of guided steps. Each participant was de-briefed after each scenario.

A total of 11 participants took part in the study, all of whom usually wrote and/or recorded their musical ideas. User evaluation testing of a prototype of the level developed here is generally conducted with a relatively small number of participants (in the range of 5 to 10), as this study size has been found in practice to be sufficient to identify a design's most significant problems [20]. When this number is exceeded, the cost of expanding the study begins to exceed the potential benefits.

All participants in this study had been, or (at the time of the evaluation) were involved with, performing in a band. Eight of the participants were either using, or had used, GarageBand before. On average, participants spent approximately 20 h per week using a computer; of those hours, an average of 5.6 h was spent using track editing-related software.

5.1 Scenario one: capturing ideas

The following scenario was described to the participants: "You have been practicing some songs that you have to perform in a week's time. All of a sudden you get an idea for a song: a simple melody. Simple, but precious. You decide to quickly record/scribe your idea before it is lost." All participants found this a realistic scenario. In this scenario, participants attempted to capture and enrich their ideas using both systems. The intention of this scenario was to help them answer the first question posed: would a musician prefer to capture musical moments using a hypermedia environment?

Expressiveness. All but one of the participants preferred APOLLO's environment for capturing musical ideas. The common reason given for preferring APOLLO was that it was quicker to get ideas out, mainly because APOLLO stored text with the audio recordings as opposed to having to use separate applications for the audio and text (e.g., GarageBand plus Notepad). Several participants explicitly noted that they found it annoying to have to use a different program for storing lyrics when using GarageBand, and then to have to manually manage the association between the audio and lyric files. This multi-step, multi-application process for accomplishing a common task is "confusing" [Participant 5] and "stink" [Participant 4], and significantly disrupts the composition process.

Two participants spontaneously picked up on aspects of the sketchbook interface metaphor (Sect. 3) for APOLLO's design: "it's like drawing on paper" [Participant 10]; "I liken it to canvas, you're not restricted" [Participant 10]. The direct manipulation, minimalist interface arising from the sketchbook approach encourages exploration and play, possibly leading to greater engagement with a given musical idea: "it gives an opportunity to play around with ideas" [Participant 2]; "it's real easy to keep changing and tweaking . . . it seems more drafty" [Participant 7].

For mixing media together, some example usages proposed by participants were for associating lyrics, notes for memory cues, or chord/progression information with audio.

All but one participant (who was unsure) thought that vector graphics were useful for annotating/highlighting parts of an audio track. Participant 6 was particularly enthusiastic about the potential to individualize the composition process, supported by APOLLO's annotation facilities; when asked whether the ability to add lines, text, and boxes anywhere within a frame might be useful, he enthusiastically responded, "Heck yeah, because in GarageBand you can't personalise it a lot." Music composition is a highly individual process, and one design goal of APOLLO was to support composers in their preferred approaches to composition.

Studies of "active reading" annotation behavior with text documents underscore the importance of providing flexible, non-prescriptive, multi-option annotation tools; annotation tools that impose structure or restrict forms are counter-productive in that they interrupt the flow of the task [17]. Annotation styles are idiosyncratic, freeform, and mutable [16,11]. Comments from participants bear out these findings from annotation studies over text documents, ranging from anticipating limited use of the annotation facilities (e.g., "text, for notes and lyrics—not sure about arrows/vector graphics" [Participant 8]) to enthusiasm for the entire set ("underlining would be good, and arrows, lines, and boxes over work that needs reworking [Participant 3]).

Time-line in APOLLO. Seven of the participants agreed that the audio placement in APOLLO should have vertical lines for auto-snapping tracks to beats. In general, they thought that the time-line should represent a time signature instead of seconds or minutes. The consensus from these participants was that the audio track placement was too imprecise, especially for participants who were casual users of conventional track editing systems. This point highlights a potential point of conflict between the design goals for APOLLO: supporting both the initial "sketching" of an idea and the further development of that idea. While inexact track placement is appropriate for casually jotting down a sudden inspiration, at some point in the composition process, the musician must be able to accurately manipulate the tracks.

Auto-resize of the time-line was viewed uniformly positively. The effect is relatively subtle, and five participants had not noticed it until it was pointed out in the de-briefing—evidence that its interpretation is straightforward and intuitive. Two of the participants felt that auto-resize would be helpful in "viewing the 'big picture"' [Participant 1] and useful to "put things into perspective" [Participant 9].

5.2 Scenario two: organizing partially developed ideas

In the second scenario, the following description was given to the participants: "You have been working on a backing track for a song for some time now, and you have just come up with some ideas to complete your work in progress." Ten participants found this to be a realistic scenario, and one did not think they would have enough ideas to bother with organization. Analysis of pre- and post-interviews of this scenario was undertaken, seeking to answer the second and third questions, focusing on the aspects of organizing their musical ideas and the hierarchical model for tracks, respectively.

Before the participants started their work on the tasks, the spatial hypermedia concept was explained to them, and a demonstration of the software given. In relation to Question 2, all participants but one found the frames and links concept both easy to understand and useful for moving around ideas. They generally liked the freedom that APOLLO gives from having to think about files or folders. One participant liked how APOLLO only used a single window, keeping the interface uncluttered: they described the environment as, "it avoids dealing with windows, it's like opening doors rather than windows" [Participant 10].

However, many of the participants mentioned that they would probably find APOLLO more difficult to use as they added more links to their frame-sets ("yep, I understand it...but then again you wouldn't want to make too many links" [Participant 7]). A natural response is to adopt the strategy of visually and spatially clumping related work together ("easy to get lost in it...I like to see all the information in one go, as much as possible" [Participant 11]; "I like to keep snippets all in one view" [Participant 6]). These comments evoke the "lost in hyperspace" problem experienced by users of complex hypermedia systems [24]. One participant specifically noted that he "did not like the spatial feel in APOLLO" [Participant 5] These problems are not entirely unanticipated; the searching capability had been included precisely to counter such issues (and is evaluated below). And, of course, the APOLLO user is interacting with a set of personally created linkages in a personal document collection—and so is not likely to experience the more extreme disorientation experienced in a novel, unfamiliar hypermedia system [24].

In relation to Question 3, three participants did not explicitly say whether they found the linked tracks useful or not. Three other participants liked the linked tracks: they found the concept of focusing on a single track at a time to be useful, particularly as an encouragement to explore and play through ease in "grouping ideas and playing around with them" [Participant 3]. Five participants did not like the concept. They felt the whole process of having to use links was "tricky" [Participant 5], involving "a whole lot of thinking" [Participant 5] to manage the frame real estate as it is filled in. Link management was seen as being "like a big puzzle" [Participant 11], requiring mental attention that distracted from the task of music composition. Furthermore, these five participants generally preferred to see as many tracks as possible in a single view, favoring vertical scroll-bars over linking because audio tracks were more easily accessible.

Three participants thought that using linked tracks for sample re-use ("stamping" out copies of the sample on the frame) was valuable. They commented that it was quicker to try out new effects with APOLLO than with GarageBand, which forces the user to manually duplicate and position a sample to try out new sounds.

5.3 Scenario three: retrieving ideas

For the final scenario, the following description was given to the participants: "You recall an idea that you previously worked on a couple of months back, but you cannot remember where you stored it. You want to find it to extend it a little further, or use it for further inspiration." Nine of the participants found this scenario realistic. Two said that they would usually remember the names of their ideas.

To complete the scenario, participants were asked to locate existing ideas in three different ways: browsing, text searching, and (for APOLLO only) melody searching. Given the time constraints of the evaluation, previously prepared set of frames was used as a surrogate for what a user might have produced over a longer period of time. This form of evaluation is known to have its limitations, as the user lacks the familiarity they would otherwise have, had they been their own files. To help offset this, this task was intentionally scheduled as the last of the three scenarios, giving the subjects some lead time to build a better understanding of the provided resource.

Browsing. Six participants preferred browsing in APOLLO's environment rather than using the file system. They preferred APOLLO because it was quicker to check whether a link was pointing to the idea they were trying to locate; this contrasts with having to wait on loading a GarageBand file before they could verify if the file was the one that they were looking for.

Another common reason for their preference was that they could find an idea in many ways—for example, by mood or by genre—but only if they kept their idea set "organised properly" [Participant 6]. That, of course, is the catch; it can be difficult to motivate oneself to put effort into organizing and adding text tags to musical ideas when the payoff for this immediate effort is ease of browsing and searching in the distant future. Procrastination over making the effort to "organise properly" is common with personal file structures [19], personal photo collections [22,12], and personal music collections [10]; it is likely that APOLLO's users would encounter the same difficulties in motivation in maintaining their musical ideas.

Photo collection management software can finesse the problem of user reluctance to invest time in metadata creation by automatically augmenting photos with context data (for example, with GPS location data and timestamps) [13]. One advantage of a file-based search system, such as that supported with GarageBand, is that the user has access to system cues such as filename, filesize, and date of last access. Creation and recent access history dates are particularly appropriate for locating a music idea, given their potential for reminding the user of the context of personal events and activities that sparked the idea; the equivalent facility could be easily added to APOLLO.

Searching by text. Three of the participants thought both systems were equally as good for performing text searches against their ideas collection. Seven of the participants preferred using the file system for text searches, primarily because Spotlight search (provided as standard on MacOS X) was more accessible and interactive: Spotlight continuously searches as keywords are entered. We note that the participants' preferences were based mainly on usability aspects, which was not the focus of the study. However, two more substantive points were made in favor of searching text in APOLLO: participants found it faster to check the results in APOLLO rather than having to wait for files to load to verify whether they were the correct match; and because none of the participants used a narrowed file search, they noted that APOLLO was superior in that only their idea collection would be indexed (by default) as opposed to having to wade through a list of irrelevant files found elsewhere on the system.

During the third scenario, the participants commented on how they would find it easy to become lost among the frames and links. However, it transpired they were unaware of the "home" link (a link on the APOLLO overlay which directs to the user's home frame). Making this more prominent, in a revised version of the interface, would be easy to achieve.

Searching by melody. Nine of the participants found the melody search to be useful. One key advantage of query-by-humming is that the user does not have to context switch between melody and text when constructing a query—the query and the information needed are both in the same format (audio). The process of conventional text search or browsing for a musical idea distracts from the idea itself: "by the time you find an idea doing it the manual [text-based search or browse] way, the inspired idea (to go with it) would have been lost" [Participant 2]. These participants personally related to the scenario, mentioning that it is a common problem.

Most of the participants found the idea within the top 10 search results (the stronger vocalists found a match in one or two clicks). One participant was not sure whether they would use it. The remaining participant [Participant 9] believed that it would be their "last resort" when wanting to locate an idea—not because of any shortcoming in the support for creating a melodic query, but because this participant experienced difficulty in interpreting the melodic search output ("every other method was easier to look around the results listing than the melody search shows"). In working through Scenario 2, Participant 9 recorded several queries before finding a match, even though the correct match was always listed within the top 5–8 search results returned for their queries.

The problem is related to the text search usability deficiencies noted above, rather than representing a fundamental issue with the melodic search function itself.

Two participants noted that the melody search would be particularly useful because they tended to label their ideas as something "random named/abstract named" [Participant 7] and/or "generic". Again, creating descriptive metadata is a task that is often postponed or avoided altogether, even after repeated frustrating experiences when trying to re-locate an item: "I get an idea, I save it as something generic ("my idea #3") then I don't know which idea is which because they have all the same names" [Participant 2]. Two participants believed that if they scanned through a listing of all their ideas (viewing the names of their ideas), they would recall the name of the idea they would be looking for.

6 Conclusion and future work

Collectively, the findings on the four posed questions for the evaluation study indicate that hypermedia systems are a suitable environment for musicians in managing their musical moments. The work touched on new ground with the concept of hierarchically structuring audio tracks that form a complete song. The user group was divided over whether they saw this ability as useful. For subjects who preferred the flat 1D list of tracks, it is worth noting that this is a subset of the hierarchy approach. It would be quite instinctive, therefore, for users who preferred to organize their tracks this way, to do so within APOLLO. An additional possibility to explore in this area would be to experiment with a tree-view of audio tracks (expanding and collapsing nodes in the tree when clicked upon).

Most of the participants mentioned how valuable they would find the ability to have text tracks for lyrics, so that the text lines up with the audio. This throws open a window worth pursuing for discovering the different ways to merge the time dimension with text for viewing and authoring.

In general, musicians found the melody search useful for locating existing ideas. The text search result information displayed by APOLLO was found to be of inferior quality to that provided by the in-built text search provided by MacOS X, Spotlight. This would be straightforward to rectify. Evaluation work also highlighted the need for APOLLO to (optionally) support a snap-to-grid feature for adding objects on the time-line, and that the time-line should be represented more abstractly as a time signature, rather than in seconds. This has been implemented, but as of now, not tested through additional evaluation.

Finally, efforts in keeping APOLLO's responsiveness quick turned out to be extremely worthwhile. It became a key reason as to why the musicians preferred a hypermedia environment over a file system for locating their musical ideas, whether it be browsing an organized structure or browsing a list of search results. This contrasted sharply with the file-system equivalent, where musicians found the navigation process in a file system too slow for keeping their inspiration moments fresh—having to wait on load times while opening and closing applications to verify a match.

References

1. Abrams, S., Bellofatto, R., Fuhrer, R., Oppenheim, D., Wright, J., Boulanger, R., Leonard, N., Mash, D., Rendish, M., Smith, J.: QSketcher: an environment for composing music for film. In: Proceedings of the 4th Conference on Creativity and Cognition, pp. 157–164, Loughborough, UK (2002)
2. Akscyn, R.M., McCracken, D.L., Yoder, E.A.: KMS: a distributed hypermedia system for managing knowledge in organizations. Commun. ACM **31**(7), 820–835 (1988)
3. Assayag, G., Rueda, C., Laurson, M., Agon, C., Delerue, O.: Computer-assisted composition at IRCAM: From PatchWork to OpenMusic. Comput. Music J. **23**(3), 59–72 (1999)
4. Bainbridge, D., Nevill-Manning, C., Witten, I., Smith, L., McNab, R.: Towards a digital library of popular music. In: Proc. of the 4th ACM Conference on Digital Libraries, pp. 161–169. ACM, New York (1999)
5. Bainbridge, D., Novak, B.J., Cunningham, S.: A user-centered design of a personal digital library for music exploration. In: (submitted to) Joint ACM/IEEE Conference on Digital Libraries, p. 10 (2010)
6. Bardini, T.: Bridging the gulfs: from hypertext to cyberspace. J. Comput.-Mediat. Commun. **3**(2) (1997)
7. Bolger, N., Davis, A., Rafaeli, E.: Diary methods: capturing life as it is lived. Ann. Rev. Psychol. **54**, 579–616 (2003)
8. Bush, V.: As we may think. Interactions **3**(2), 3546 (1996) Reprint from original 1945 article
9. Conklin, J.: Hypertext: an introduction and survey. Computer **20**(9), 17–41 (1987)
10. Cunningham, S., Jones, M., Jones, S.: Organizing digital music for use: an examination of personal music collections. In: Proceedings of the International Symposium on Music Information Retrieval, pp. 447–454, Barcelona, October (2004)
11. Cunningham S., Knowles C.: Annotations in an academic digital library: the case of conference note-taking and annotation. In: Proceedings of ICADL, pp. 62–71 (2005)
12. Cunningham, S., Masoodian, M.: Metadata and organizational structures in personal photograph digital libraries. In: Proceedings of the Asian Conference on Digital Libraries, p. 458467. Hanoi (2007)
13. Frohlich, D., Kuchinsky, A., Pering, C., Don, A., Ariss, S.: Requirements for photoware. In: Proc. CSCW, p. 166175 (2002)
14. Hardman, L., van Ossenbruggen, J., Mullender, K.S., Rutledge, L., Bulterman, D.C.A.: Do you have the time? composition and linking in time-based hypermedia. In: HYPERTEXT '99: Proceedings of the tenth ACM Conference on Hypertext and hypermedia: Returning to Our Diverse Roots, pp. 189–196. ACM, New York (1999)
15. Hsieh, H., Shipman, F.: Activity links: supporting communication and reflection about action. In: HYPERTEXT '05: Proceedings of the Sixteenth ACM Conference on Hypertext and Hypermedia, pp. 161–170. ACM, New York (2005)
16. Marshall C.: Annotation: from paper books to the digital library. In: Proceedings of the Second ACM International Conference on Digital Libraries, pp. 131–140. Philadelphia, Pennsylvania, USA, July (1997)

17. Marshall, C.C., Brush, A.J.B.: Exploring the relationship between personal and public annotations. In: Proceedings of the 4th ACM/IEEE-CS Joint Conference on Digital Libraries, pp. 349–357. Tucson, AZ, USA (2004)
18. Marshall, C.C., Shipman, F.M.: Spatial hypertext: designing for change. Commun. ACM **38**, 88–97 (1995)
19. Nardi, B.: A Small Matter of Programming : Perspectives on End User Programming. MIT Press, Cambridge (1993)
20. Nielsen, J.: Why you only need to test with 5 users. Alertbox (2000) www.useit.com/alertbox/20000319.html.
21. Rivlin, E., Botafogo, R., Shneiderman, B.: Navigating in hyperspace: designing a structure-based toolbox. Commun. ACM **37**(2), 87–96 (1994)
22. Rodden, K., Wood, K.: How do people manage their digital photographs? In: Proceedings of CHI, pp. 409–416 (2003)
23. Sawhney, N., Balcom, D., Smith, I.: Authoring and navigating video in space and time. IEEE Multime'd. **4**, 30–39 (1997)
24. Theng, Y., Thimbleby, H., Jones, M.: Lost in hyperspace: psychological problem or bad design? In: APCHI 96, pp. 387–396. Singapore (2006)
25. Tsandilas, T., Letondal, C., Mackay, W.: Musink: composing music through augmented drawing. In: Proceedings of the 27th International Conference on Human Factors in Computing Systems, pp. 819–828. Boston, USA (2009)
26. Turkle, S., Papers, S.: Epistemological pluralism: styles and voices within the computer culture. Signs: J. Women Cult. Soc. **16**(1), 128–157 (1990)
27. Wardrip-Fruin, N.: What hypertext is. In Proceedings of the 15th ACM conference on Hypertext and Hypermedia, pp. 126–127. ACM, New York (2004)
28. Wheeler, L., Reis, H.: Self-recording of everyday life events; origins, types, and uses. J. Pers. **59**(3), 339–354 (1991)
29. Wright, J., Jameson, D., Oppenheim, D., Pazel, D., Fuhrer, R.: CyberBand: A hands-on music composition program. In: Proceedings of the ICMC, pp. 383–386. Thessaloniki (1997)
30. Zappa, F., Occhiogrosso, P.: Real Frank Zappa Book. Touchstone, New York (1989)