Interactive Audio on the Web

Review and Recommendations

Prepared by the
Web Audio Working Group of the
Interactive Audio Special Interest Group

December 2002

Published by
MIDI Manufacturers Association
Los Angeles, CA

Disclaimer

v.9 Printed June 2003

MMA
POB 3173
La Habra, CA 90632-3173

Table of Contents

Introduction

By Steve Horowitz, Chairman, IASIG Web Audio Working Group

The IASIG Web Audio working group operates from a position of enlightened self-interest. We all genuinely wish to see the web grow and thrive (and along with it, the tools that we use on a daily basis). It is our sincere wish that this document will find its way into the hands of like-minded individuals.

Whether you are a composer, sound designer, software developer or hardware manufacturer, this document has been produced for you in the grand spirit of the IASIG: We wish to unite composers and designers with manufacturers in direct dialog for the sole purpose of helping to produce better tools for the creation of interactive audio on the web.

This group began its work in the summer of 2001—a very volatile time for web audio delivery systems. Today the landscape is as confusing as ever. During the writing of this document formats have come and gone. Some formats have never even been given the chance to arrive. The audio side of the web is in a constant state of flux. We hope that this document can be used to aid the composer/sound designer in the field to make a bit of sense out of all this rapid movement. Ideally, it will also give manufacturers some useful ideas for future product enhancements.

Each of the current leading web audio technologies is covered in its own section (in no particular order), providing pros and cons, as well as an overview of the features and capabilities. The text is heavy on technology, filled with no-nonsense facts, written from “one designer to another", with a hands-on point of view. The technologies were chosen based on group discussion. We tried to address the major players on the web right now, as well as including a long list of new and emerging technologies. Multiple people worked on each section, to make sure that nothing said was just “one person’s opinion”, and as much as possible we have checked our facts (and opinions) with the companies that produced these products, as well as with other experts.

A lot of work went into producing this document. I'd like to personally give a big round of applause to the folks that volunteered their time and effort to the cause of interactive audio. (Dramatic music plays here to fade….).

Steve Horowitz
Composer/Sound designer
WAWG Chairman

General Recommendations

On the Internet today, we find audio is being addressed in isolated, vendor-specific circumstances, which has led to proprietary and incompatible systems. Each of the technologies and formats examined herein address one or more particular problems of audio on the Web, but they all have their drawbacks and shortcomings. These include only specific browser support, oftentimes dizzying complexity, incompatible architectures and/or lack of widespread deployment. There is no higher audio-centric consciousness at work to facilitate a consistent set of audio capabilities. It will always be up to manufacturers to distinguish themselves through their own technologies, so standardized implementation will never be the answer.

As audio developers, however, we should not be satisfied to write to an audio-impoverished platform. The solution is to develop and agree on a common vocabulary for what audio needs to do. On whatever platform we find ourselves, we need the ability to start, stop, pause, set volume, set pan, loop, branch and transition from one piece of audio to the next for a clean and seamless audio experience. We need to build and provide for this functionality on multiple platforms, both hardware and software, to deliver a consistent, high-quality audio experience throughout this new high-tech topography.

It is this Working Group's recommendation that an Interactive Audio format and rendering engine for the Web should have the following capabilities:

1. Respond to sound requests by playing appropriate audio media,
2. Construct a continuous soundtrack (music + SFX) from discrete media chunks,
3. Select the media chunks to play either by static playlists or dynamic rules,
4. Mix and/or mute parallel tracks within and across media chunks,
5. Dynamically control the volume, pan, spatial location, and SFX and other DSP parameters of the currently playing media,
6. Provide comprehensive media handling services for data provided locally or anywhere on the Internet,
7. Provide communication links between the rendering engine and the game or host.

As the IASIG's experience in such efforts as DLS and I3DL2 teaches, structured dialogue between manufacturer s/developers and composers is critical for such a specification to enjoy industry-wide support, adoption and consistent implementation. For this to happen, the WAWG feels the IASIG needs to be more proactive in defining recommended audio practices, capabilities, services and standards in the area of Web Audio, and in particular should get more manufacturers involved in these discussions.

Overview of the Leading Technologies

Author: Michael Sweet

FLASH

Flash allows the flexibility of having both embedded MP3-compressed sounds and the ability to stream elements while a Flash (swf) movie is playing back. Volume and panning can be controlled through program control, and there is an envelope editor. With the envelope editor, it is sometimes possible to save space by creating several sounds out of one.

As swf movies are downloaded they get stored in the browser's Internet cache. Flash allows the creation of shared libraries, so you can re-use audio assets from one movie to the next. Since quite a few Flash projects are done in multiple movies this is a useful option. There is a bug when using sounds from shared libraries; shared library sounds must be played from the score timeline, and not from playback control. This limits things like program control over individual panning, and volume control of shared sounds.

Sounds that are started at the same time will remain in sync with one another, so it is possible to have your drums on one track, keyboards on another, etc. Syncing this way can be somewhat unstable on slower machines (Mac G4s below 450 Mhz and PCs below PII). You can also use the envelope editor to create variable mixes, but there is a maximum of 8 envelope points per layer or 'track'. Since each fade in/out takes 4 envelope points, this means something like a total of 2 fades in/out per layer, which is pretty limiting in practice. Starting in Flash 5, it has also been possible to sync sounds using the SoundObject and ActionScript.

These drawbacks to syncing in Flash make "branching" all the more important. In Flash 5, the only way to branch from one section to another is to use the volume controls to do cross-fades. There is no way to sync to a beat dynamically through program control. FlashMX, however adds a new handler called "onSoundComplete". Using this method, one can branch between sections seamlessly and select branches interactively.

There are several methods for syncing audio to visuals in Flash, exemplified by "start" and "stream". "Start" allows the movie to play back at whatever rate it can and show all of the visual frames. "Stream" will skip visual frames in order to maintain audio sync, similarly to QuickTime. Most Flash designers are reluctant to use this method because it reduces the quality of the visuals. Flash does not support meta-event markers or cue points as in Shockwave, which would allow integration between audio playback and other behaviors.

JAVA

Even though Java Sound has been around for almost two years, at the time of this writing most of the sites that are creating Java content still only support the older Java 1 spec. Typically, sites that use Java want to be machine friendly, so they still require games to be produced in older formats so they will play back on as many machines as possible. The Java Sound API is part of Java 2 Standard Edition (J2SE) version 1.3.0 and higher.

Java leaves composers and sound designers some tough choices: Either push for more support of the Java Sound libraries, limit one's creative choices by using only the AudioClip API, or be forced to use the non-standard sun.audio classes.

Java Sound supports a 64 voice engine that will playback AIFF, .au and WAV audio files. MIDI file support also includes SMF and RMF (Beatnik) formats. Volume, panning and pitch can be controlled. Java Sound also supports callbacks from the MIDI or (in a more limited way) audio files themselves. Since a generic sound bank is not included with the standard Java 2 Runtime Edition (JRE) distribution it may also be necessary to download this in order to play back SMF and some RMF files. This download is between 0.35 MB to 4.0 MB, depending on the quality of the general MIDI sound set. Sun is working on a solution to facilitate soundbank deployment, but the details of this solution are not yet available. With Java it is possible to create audio and MIDI files on the fly.

AudioClip is the only audio API prior to Java 2 that is guaranteed to work everywhere. However, it is severely limited in its abilities. AudioClip has only a few methods for playing back sound (start/play/loop/stop). There are no options for panning or volume. Files cannot be streamed-they must be downloaded in full before playback will begin. AU uLaw 8k 8bit is the only supported sound file format. Java is quite a sophisticated programming language so it is possible write your own Java code to convert other sound formats to AU, but the only native playback support is the antiquated AU file. The AU file format yields large file sizes compared to the MP3 format and is quite inferior in sound quality.

The sun.audio classes, which are available in most of Sun's and Microsoft's Java implementations, allow you to copy and paste loops together so you can build a larger piece before starting playback. This allows you some flexibility in creating larger loop-based pieces. However, the sun.audio classes were never an officially supported part of Java's language specification, and both Sun and Microsoft have made moves to discontinue supporting them. Since Sun's version 1.4.0, the sun.audio classes are no longer accessible to applets, and since Microsoft's J#/.NET development platform, they are no longer available as a "third-party" API.

Although Java Sound is a very powerful sound engine, we may not see its potential fully realized. With Microsoft's move to stop shipping Java with Internet Explorer, Java's life in web-based multimedia applications for mainstream Windows users may be short.

QUICKTIME

QuickTime is a good mechanism for delivering a wide array of file formats. QuickTime supports almost all audio file formats and compression methods, included MP3, AIFF, WAV, MIDI formats, DLS, SoundFont and, as of QuickTime 6, AAC and MPEG-4 (see "What's New in QuickTime 6.3?" below).

QuickTime offers a rich API, fully accessible through Java as well as C, allowing its use as the media handler for any number of applications.

There are a few JavaScript controls (although JavaScript control of QuickTime in Microsoft Internet Explorer is currently unavailable on Macs). Unfortunately communication with the QuickTime plug-in through JavaScript is quite limited. There are only a few controls, including play, stop, setposition, nextframe, lastframe.

The QuickTime format allows for custom programming to be built into the movie itself, achieving application-like levels of interactivity within the movie itself. Such movies can be built using third party tools such as LiveStage.

When building interactive QuickTime movies the author has access to sound playback, sound levels and panning. Other controls include simple EQ: treble and bass. There is access to playing notes from the general MIDI sample-based synthesizer built into QuickTime. There is no position specific branching or looping, although it is possible to cross-fade between two audio tracks that start at the same time. MIDI and audio tracks will stay in sync if played from the same QuickTime format .mov file.

QuickTime also supports multiple streams via an Apple- or Unix- based server. Real time encoding of streams is available, as with the RealAudio plug-in. There are many sophisticated parameters for controlling live QuickTime feeds from a server.

Director also has a plug-in to include QuickTime content as part of a Director movie so that it is possible to control a QuickTime movie with Lingo. Flash movies can also be exported in QuickTime format and embedded within a larger QuickTime movie. There are some very interesting pieces that use Flash and QuickTime together.

QuickTime (like RealOne) supports the SMIL standard, but again when cueing events there is a p.ause in between the events, making it impossible to create a seamless piece of music that transitions between two scenes.

Powerful as it is, QuickTime's scripting language is not as robust as those found in Flash or Director. Because of this, and the more limited adoption of the QuickTime plug-in by consumers, it is not usually the first choice for multimedia delivered over the web.

REALONE

RealOne's real power lies in streaming linear music and video over the web. On the interactive side the audio feature set is minimal. Volume can be controlled but there is a limit of one buffered playback stream at a time. Since RealOne will play back Flash 4 content producers have the option of using that engine to complement the advanced streaming features provided with Real.

RealOne also supports the SMIL multimedia markup standard, so it is possible to cue audio events one after another. Unfortunately, there are slight p.auses between events, so building and cueing seamless song elements is not possible. Cross-fading is not supported because it is only possible to control one volume stream.

In addition to SMIL there is a variety of other methods for scripting control of RealOne, including JavaScript, but SMIL has emerged as the umbrella technology.

The primary compression methods used by Real are proprietary, but Real can also be used to play back other codecs including MP3. The true advantage of Real's compression scheme is the ability to encode once for all the bit rates that may be encountered, using Real SureStream technology, although this does result in larger file sizes on the server.

SHOCKWAVE

Shockwave offers a surprising number of audio capabilities. Although it lacks any kind of waveform envelope editing as featured in Flash, it does offer a rich set of Lingo commands to control and manipulate sound. Compression of audio is mostly done through Director's swa (Shockwave audio) format which is basically MP3 with a Macromedia header. No MIDI features are currently available with Shockwave except through the use of third-party Xtras (Beatnik, Bonneville's CPS or Sourceforce's Sequence Xtra) which require an additional one-time download. Xtras that are verified as "Shockwave-safe" download seamlessly into Shockwave's plug-in structure. They only require a Verisign message to install and do not require restarting the browser.

On the most basic level Shockwave can manipulate volume and panning in real time. A sound is played back out of one of its 8 channels, and each of those channels has volume and panning characteristics which can be controlled by Lingo in real time. Unfortunately there are no other DSP features, such as Beatnik's reverb and filters.

Shockwave has four different methods for playing back sound. The most interesting of these allows users to cue up sounds and play them back sequentially. For instance let's say we have a two different drum loops. We can create a playlist in which we have the first drum loop play four times, then play the second loop two times. Then we pass this list to a command called sound(1).setplaylist(mylist) and start the stream. We can also add sounds to the end of the playlist after it's begun playing. This also allows for seamless branching.

For example, assume we've started a playlist which has 10 sounds in it. The game has just finished a level but we still have four sounds in the cue. Shockwave allows us to clear these four sounds and cue up an ending which will start after the current sound ends. This allows a fairly robust mechanism for creating looped based scores.

In addition there is the ability to also break a looping sound, play the remaining portion of the sound and cue up additional sounds after it.

Shockwave does allow you to have two of these tracks working in sync with one another, but on slow machines (pre G3 Mac, and pre Pentium II Wintel) these tracks will slip out of sync with one another.

The second way to play sound in Shockwave is to use the timeline, which has two dedicated channels for sound. The third is via the puppetsound method, which plays a sound from the internal cast of sounds. The fourth is soundplayfile, which streams a sound from a local disk, and so is not very applicable for web use.

Shockwave is also able to pitch sounds in half steps from –64 to +64. This allows the user a little more flexibility when they only have space for, say, two footsteps in the game. Although one might think you could build a MIDI player using this feature, the timing mechanism in Lingo is not good enough to produce accurate playback.

Although Shockwave will download sounds in the background while a movie (dcr) is running, a file must be totally loaded into memory before it will begin playing back. There is an exception to this if you are playing back an external Flash movie inside of a dcr file. In this case the Flash movie takes precedence.

Macromedia Flash

Author: Chris Burke

o Pros:

- Even without supporting plug-ins, Flash provides interactive control via actionscript of most of the functions that are important for creating sonified web applications.
- Flash ActionScript is relatively easy to learn.
- Being part of a successful animation package from an industry leader, Flash audio is in a good position to survive and grow.

o Cons:

- Unlike Shockwave, there is no Xtra implementation. Beatnik can be used to extend Flash audio capabilites but using a second plug-in is often prohibitive, and the necessary ja vascript control is not available in certain browsers.
- Synchronization is limited to sounds started at the same time and is sometimes unreliable.
- Flash has a limit of eight sounds playing concurrently.

o Recommendations:

Being in such a good position in the industry Flash is certainly a competitor for most useful web audio tool. Use it and push Macromedia to give developers a level of control over audio comparable to that available for visuals in Flash.

---

o Describe the platform.

The recently released FlashMX is the latest version of Macromedia's popular web authoring tool. Except where noted, most of the features and techniques discussed here apply also to Flash 5, still the most often targeted version. The animation features and use of vector graphics in Flash make it very useful, but Flash 5 and FlashMX have also introduced audio features that make it an important tool for authoring web audio.

With the introduction of the Sound Object in v.5, Flash authors are now able to trigger and manipulate certain aspects of sounds without having them tied to a timeline. Dynamic control over volume, pan start time and even rudimentary sync capabilities are now possible. When combined with FlashMX's more robust Action Script, sound can become a far more interactive element than was possible with earlier versions of Flash.

o What kind of experiences can this platform deliver over the web? What is and is not possible using the platform?

Using Flash, authors can create animated programs with synchronized soundtracks as well as interactive sound elements. Functional use of sound as buttons, transitions, background loops, etc. has be available for some time. With the new controls listed above, more advanced programs are possible:

- Games with evolving soundtracks in which sound elements can be brought in and out and stay in sync with some restrictions.
- Music remixers with reliable sync and visual mixer-style controls.

- Sound games in which sound elements can interactively manipulated.
- Animations with streaming, linear soundtracks.

All of the above examples suffer from certain limits.

- No equalization or effects, such as filters, echo, reverb. This would be a requirement for a more advanced music remixing application. One can create an echo effect, but the function must be 'built'.
- A limit of 8 simultaneous sounds. This is extendable somewhat with with certain restrictions, but it seriously limits what one can do musically.
- Synchronization is only possible with sounds that are started at the same time. On lower end machines, sync can be unreliable. One cannot load a new sound while others are playing and have it play in sync with those others. Flash does not use midi and relies on the movie's frame rate as a clock. One work-around is to use streaming sound to force a consistent frame rate and trigger 'event' sounds in the same timeline, thereby locking them into sync together. While this is a very useful trick, there are still serious limits to working this way, not the least of which is the attention one must pay to preparing the tracks before importing.
- Sound quality is still an issue, as it is with all web audio apps to some degree. This is partly a bandwidth issue, which makes compression of the sounds necessary. Mixing multiple compressed sounds together in Flash 5 can cause a clicking distortion or general muddiness resulting in, for example, dialog that is not clear enough to hear over background music.

The subject of rhythmic synchronization in Flash is worthy of a closer look. The two methods of rhytmically syncing sounds in Flash are "timeline-based" and "SoundObject-based".

Using the timeline approach, sounds that are started at the same time will remain in sync with one another, so it is possible to have your drums on one track, keyboards on another, etc. In this way, you can use the envelope editor to remix, however there is a maximum of 8 envelope points per layer or 'track'. Since each fade in/out takes 4 envelope points, this means something like a total of 2 fades in/out per layer, pretty limiting in practice. There is also sometimes a problem getting the sounds to start at the same time requiring a simple workaround.

Using the SoundObject approach (supported since Flash 5) sounds started at the same time remain in sync, but syncing this way can be somewhat unstable on slower machines (G4: <450 Mhz and PC: <PII). These drawbacks to syncing in Flash make "branching" all the more important.

In Flash 5, the only way to branch from one section to another is to use the volume controls to do cross-fades. There is no way to sync to a beat dynamically through program control. FlashMX, however adds a new handler called "onSoundComplete". Using this method, one can branch between sections of a remix or choose a fill to end a phrase. A variable can be used in the argument of onSoundComplete, allowing the branch path to be interactively controlled. Given the problems with syncing multiple tracks in Flash, this is a big improvement.

There are several methods for syncing audio to visuals in Flash, exemplified by "start" and "stream". "Start" allows the movie to play back at whatever rate it can and show all of the visual frames. "Stream" will skip visual frames in order to maintain audio sync, similarly to QuickTime. This option actually streams the data from the Internet so the sound can start playing before it resides totally in memory (note: the whole flash movie is streamed not just the streaming audio, and also streaming sound is embedded into the movie) In order achieve this Flash basically cuts up the audio into many pieces and each Flash frame contains some of the streaming audio.Most Flash designers are reluctant to use this method because it reduces the quality of the visuals. Flash does not support meta-event markers or cue points as in Shockwave, which would allow integration between audio playback and other behaviors.

o How stable & reliable is the platform?

Flash is a remarkably stable authoring environment on both Mac and Windows. Since the Player is a browser plug-in, it is susceptible to browser crashes. Flash is considered fairly resource-efficient as far as plug-ins go.

o What issues (technical, usability-related, economic, other) might stand in the way of adoption by users? By developers? By content providers, clients, web hosts? How many users does the platform already have? Is it well established and ubiquitous, or marginal?

Flash is considered about as ubiquitous as any web technology. Macromedia claims that 98.3% of the online audience has the Flash Player. Adoption is certainly among Flash's strong points and a good reason for developers to author with it. This in turn insures a plethora of Flash media on the web, which encourages users to download the plug-in if they don't already have it. The plug-in is free to download and it comes bundled with most browsers.

On the other hand, some clients are still strict about the 'no plug-ins' rule. As efficient as Flash is with file size, sound can add a lot of download time to a page. This is certainly a drawback, albeit one that all web audio apps face to some extent. Some apps are stronger in this area than Flash. Beatnik, for example can create more audio bang for the file size. Streaming-only solutions like Real Media boast much faster delivery but are weak in the interactive area. Flash is strong on interactivity and quite competitive on the file size issue as well. This issue is less important for so-called 'rich media' sites, where users expect to wait a little for a more extensive experience.

o What is the platform company's strategy for web audio?

Macromedia has a mixed record with audio. Some of us remember the Deck fiasco three years ago. Sound Edit has not been updated in years. Of course, many technology companies seem to consider sound to be far less important than visuals. One would hope for something more like Beatnik to be built into Flash, allowing authors to create a richer sonic experience with advanced tools to keep the file size down as well. Macromedia is rightfully proud of Flash's audio tool set. Flash 5 was a big step in the right direction, but it seems clear that few web technology companies are dedicated to bringing audio tools up to the level of the tools they create for the visual side.

o What are the capabilities of the authoring tool in relation to web audio development? What is the quality of the interactivity of the platform?

Flash sound capabilities are still fairly limited to the essentials. While superior web audio authoring tools exist, Flash has the advantage of having their audio tools built into a powerful animation tool. But with other options like Beatnik, Java Sound, and emerging technologies, one would hope that some of these audio advances will find their way into a future rev of Flash.

The introduction of the Sound Object, and advances in Action Scripting, have greatly improved interactive control of audio in Flash. Some simple functions could be added that would increase the amount of interactive sound control. But more importantly there should be some attention to the possibility of freeing developers from using loops and chunks. In order for music and sound to be truly interactive on an authoring and user level, the elements need to be of a sufficiently fine modularity. One welcome advance in FlashMX is the addition of the "onSoundComplete" handler described above.

It may be too much to ask for Flash to incorporate an instrument builder and sequencing capabilities as in Beatnik. One would hope Macromedia would build in full support for some similar third party or open-source composition tool that allows scriptability without requiring another plug-in for playback.

o What kind of interactivity are users actually interested in? How does that relate to the capabilities of this platform?

What is sometimes read as a lack of user interest in interactivity may often be frustration with the web as a medium in it's infancy. As developers, it is up to us to define the future of interactivity, keeping an eye on user feedback along the way. With Flash's current audio tool set, web developers can create a limited sound experience and excite a limited audience. In order for that audience to grow, Macromedia will need to keep pace with the interesting developments in web audio coming from the other players by enhancing the Flash audio tool set with more synchronization based controls. It should be noted that one way currently available to extend Flash audio capabilities is the combination of Beatnik and Flash, or "Flashnik". At this date Beatnik has indicated that they will be no longer develop their internet tools, instead making the Beatnik Player open-source. If this means that Beatnik will continue to be updated, Flashnik offers a powerful, if less than optimum solution for advanced web audio apps.

Java Sound

Authors: Jan Martin Borgersen and Jeff Essex

o Pros:

- Scales across a broad range of platforms.
- Robust architecture based on Beatnik technology.
- Good potential to succeed as Java continues to succeed.

o Cons:

- No clear vision of future development from Sun.
- Difficult to integrate with browsers.
- Programmer-centric; no standard authoring tools for content creators.

o Recommendations

Based on our first and second negative bullet points, we are not convinced that Java Sound is the correct vehicle for an interactive **web** audio solution.

---

o Describe the Platform

Java Sound is an API for the playback, recording, and manipulation of sampled and MIDI audio on Java-enabled platforms. Being an API, it merely defines the programming interface for controlling sound in Java. It is not an editing environment or a player application, although editing environments and player applications can be built in Java by using this API. For web delivery, audio content using the Java Sound API must ultimately be in the form of a Java Applet (a small, portable application) running in a Java-enabled web browser.

Sun Microsystems' implementation of this API has been shipping as a core technology in the Java 2 Standard Edition Runtime Environment (JRE) and Software Development Kit (SDK) downloads for Windows, Solaris, and Linux since version 1.3. For it's lowest-level audio rendering, this implementation uses the Beatnik Audio Engine (BAE), which is the same cross-platform audio engine that powers the Beatnik plug-in. It should be noted here that the Java Sound API is not a simple wrapper to the BAE, nor is it a replacement for Beatnik's Music Object API, which allows JavaScript control over the BAE from inside web pages. Instead, this API is intended to allow Java developers to use audio in their applications and Applets by exposing some of the most interesting features of the BAE, and adding quite a bit of new functionality. The API includes support for swappable General MIDI soundbanks, programming control over software synthesizers and sequencers, and an extensible set of file readers, file writers, and audio encodings.

Since Java Sound is a core Java API, it is the natural choice for simple audio programming in Java. However, it is limited by its standard number of supported file readers, file writers, and codecs, and it does not provide any network-level protocol support for streaming audio. For developers with greater needs, Sun also offers the Java Media Framework (JMF) as an optional extension to the Java language. In a broad sense, JMF has similar functionality to QuickTime. It provides an architecture for controlling and synchronizing the playback of various media types (including audio and video), it supports more file formats and encodings than Java Sound alone, and it offers API's for streaming protocols.

It is important to keep in mind the small role Java Sound plays relative to the larger role of Java technology in general. In recent years, Java has succeeded more as a business-oriented enabling technology rather than a media-delivery technology. Audio can certainly play a role in Java applications, but Java's success at this point has not been with entertainment applications. However, it is also true that Java Sound has the potential to have a broad base of support as Java applications become ubiquitous across a range of platforms.

o What kind of experiences can this platform deliver over the web? What is and is not possible using the platform?

Audio experiences using Java over the web will take the form of Java Applets running in web browsers. Since Java support is extremely inconsistent across browsers, this form of delivery is very difficult. It is certainly possible to deliver a Java-based multimedia experience via the web, but the average user will have to jump through several hoops to get Java up and running. Certain Mozilla-based browsers (including Netscape 6 and later) are the only browsers that will automatically support Java Sound after installation; users of other browsers will have to install Sun’s Java Plug-in (http://java.sun.com/getjava) as well as a software Soundbank before they can take advantage of this technology.

Java Sound is useful for interface effects; small sounds can be preloaded into RAM via the “Clip” interface for low-latency performance within Java Applets. It is also useful for playing back soundtracks of MIDI files created for a General MIDI set of instruments, and RMF files created with Beatnik authoring tools. The JMF API’s provide mechanisms to properly stream sampled-audio soundtracks and voice-overlays.

o How stable and reliable is the platform?

Sun’s implementation of the Java Sound API is reasonably stable and reliable for basic audio playback and sampled audio recording. Sun chose the BAE for its implementation platform because it had already proven itself to be stable and easily ported to new platforms. Furthermore, standard sets of General MIDI software soundbanks ensure that MIDI and RMF music will sound the same from one machine to the next. The only obstacle to ensuring reliability is getting both the Sun Java Plug-in and the appropriate software soundbank installed properly on the end-user’s machine.

The current Java Sound release suffers from a few functionality limitations that Sun plans to address in future implementation releases. Most of these will be invisible to end-users of web Applets. One of the chief complaints on the Java Sound Interest mailing list (http://java.sun.com/products/java-media/sound/list.html) was the lack of MIDI In support, which prevented developers from building useful sequencer applications using this API. Sun’s response was that this functionality was not stable enough by the deadline for J2SE version 1.3, and was therefore left out of the original release. It was added in version 1.4.1 for Windows, and will follow for Linux in a future release.

o What issues might stand in the way of adoption by users, developers, clients? How many users does the platform have -- is it ubiquitous?

Until browsers adopt the latest versions of Java in their standard downloads, it is unlikely that Java Sound will be adopted by a broad audience. Support is required at both the OS level (the inclusion of the Java Virtual Machine) and the browser level, and Netscape 6 (and later) is the only browser that currently comes standard with a Java VM that supports Java Sound. Microsoft has completely eliminated Java from its standard installation of Internet Explorer 6, citing download size issues. When users of IE 6 encounter Java applets, they are automatically directed to download the latest Microsoft Java VM, which does not provide Java Sound support.

If users are properly directed to Sun's Java Plug-in website, they can download and install the necessary components to use Java Sound in any browser. The only catch here is that Sun's Java Plug-in does not include a General MIDI soundbank in its standard download (Java Sound will use the host's sound card if it doesn't find a soundbank). The consistency of MIDI music cannot be guaranteed without also directing users to the Java Sound website to download soundbanks (http://java.sun.com/products/java-media/sound/soundbanks.html). Sun claims to be working on a method to facilitate deployment of soundbanks, but this technology is not yet available at the time of writing. Experience shows that requiring multiple downloads and installations is a significant obstacle to broad adoption.

Sun does not currently support the MP3 encoding in its core offering, however third-party MP3 extensions to the API are available for download (http://www.tritonus.org) . Developers wishing to lobby Sun to provide MP3 support are encouraged to vote for this feature request online at http://developer.java.sun.com/servlet/SessionServlet?url=http://developer.java.sun.com/developer/bugParade/bugs/4479349.html .

Despite these concerns, it is important to recognize the strengths of Java Sound. With its roots in the Beatnik Audio Engine, it scales across a range of hardware and OS platforms, it has inherent support for real-time interactive control, and it sports a solid software-based synthesizer and sequencer. The API is very well-documented by both the Java API Docs (http://java.sun.com/apis.html) and the Java Sound Programmer's Guide (http://java.sun.com/products/java-media/sound/) , and it is easily extended.

While Java strives to become a ubiquitous platform, the current reality is that very few people have experienced multimedia via Java. Yet Java Sound enjoys support from a small but dedicated community of developers. Technologies including Xemo (http://www.xemo.org), JSyn (http://www.softsynth.com/jsyn/), and JavaSonics (http://www.javasonics.com) are already making use of the API, and there is even an open-source implementation for Linux (built atop the Advanced Linux Sound Architecture (ALSA)) called Tritonus (http://www.tritonus.org). Without a strong official endorsement from Sun and the browser developers, however, it's unlikely that these efforts will garner a wide audience.

o What is the platform company's strategy for web audio?

As discussed above, it seems that Sun has a very clear strategy for Java -- to create a ubiquitous software environment that can scale across a broad range of platforms. It seems that Sun does not have a strategy for web audio per se, but the capabilities of Java Sound and JMF should ride on the coattails of Java as it encompasses more platforms.

o What are the capabilities of the authoring tool in relation to web audio development?

The Java Sound API does not have a dedicated authoring environment. Content creators can use any set of standard audio and MIDI tools to record and manipulate files. Since Sun's implementation supports Beatnik's proprietary RMF format, creators can use Beatnik Editor in conjunction with other tools to produce RMF content, and can leverage off the existing community of Beatnik developers.

o What kind of interactivity are users actually interested in? How does that relate to the capabilities of the platform?

It is our belief that few users are actually interested in interface audio for the majority of applications. Button clicks and scroll sounds in word processors, spreadsheets and the OS in general are usually disabled because they are distracting and superfluous. Java technology has succeeded in delivering scalable business applications across an enterprise, but has lagged behind native-OS applications for delivering multimedia and games. The opportunities for building a compelling audio experience using Java Sound will be directly tied to Java's future success (or failure) in adoption as an end-user desktop technology for multimedia-rich applications.

RealOne

Author: Spencer Critchley

o Pros:

- Previous versions of the Real Player have achieved the largest user base.
- Tight linking of video/audio streaming plus web browsing for a more integrated user experience.
- Broad and deep authoring capabilities through SMIL.

o Cons:

- Users are aggressively pushed to buy the premium player and content, and must search for the free version. The format also limits the user's freedom to use downloaded content, even when it's been paid for.
- The player is aggressive in taking over MIME types.
- While Real has established the lead in streaming media, adoption of SMIL for interactive authoring has been slow. Flash is still the leader for interactive content, and that lead was strengthened by the incorporation of video into Flash MX.

o Recommendations:

As one of the most widely adopted streaming formats, RealOne is a natural candidate for streaming-oriented presentations.

---

o What is and is not possible using the platform?

With the RealOne Player, Real, like several competitors, aims to enclose all of a mainstream user's digital media experience within its own environment. According to Real, "By combining streaming media, digital downloads, and Internet browsing, RealOne Player provides an all-in-one consumer application for network-based media distribution."[1]

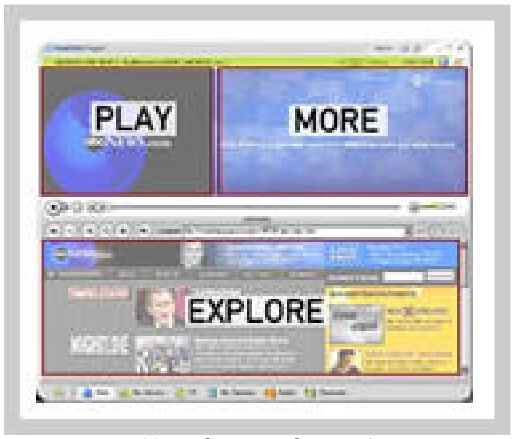
There are two broad paths towards control of media consumption: IP-based (what Real calls "network-oriented") and broadcast. Real is so far following the IP path, as is Apple with QuickTime, and is seeking to reap value from content, production tools, distribution back end (Real Servers), and distribution front end (RealOne). Meanwhile competitor Microsoft is on both paths—Microsoft's Windows Media is an IP-based technology, while so far unsuccessful interactive TV efforts such as Ultimate TV and WebTV) have been on the broadcast path. At some point the paths may converge. As it is, RealOne's 3-window interface (see 2, below) is very similar to the emerging standard for broadcast-based interactive TV.

RealOne takes over the browser function and integrates it tightly with the Player. In addition to playing Real-encoded audio and video, RealOne also wraps around other leading digital media technologies, enabling the playback of Flash 4 movies, Windows Media, MP3's, MPEG video—including MPEG4—and CDs, plus the management of media libraries, including the burning of CDs and transfers to and from portable MP3 players (although downloaded Real format audio files cannot be transferred—see 3, below). Hence a user is able to perform the most popular computer-centric media activities without leaving RealOne.

1 "Introduction to RealSystem IQ Production with RealOne Player Beta", Rev. 12/4/01, p. 1

o What kinds of experience can we deliver over the web?

RealOne's user interface is based on three windows, which are normally connected to form panes of a single window. They are called **Media Playback, Related Info** (also known as **Context**) and **Media Browser.** Real's metaphor for this layout is "play/more/explore":



(©Real Networks, Inc.)

- Media Playback: presents video, slide shows, still images, text, audio, audio visualizations or SMIL (Standard Multimedia Integration Language) presentations. (see 4,v, below).
- Related Info: displays HTML web pages in sync with media playing in Media Playback. It does this in response to clicks on the clip playing in Media Playback, or under the control of the Media Browser.
- Media Browser: a fully functioning, detachable browser, able to display HTML pages and any other kind of web content. It also features a Now Playing button, which calls up a list of media clips, and a tool bar allowing control over My Library, CD, Devices, Radio, Channels and Search.

The player supports bi-directional control among the three windows, using hyperlinks, JavaScript/ActiveX, and/or SMIL. In effect, Real Media clips and related web pages can be displayed in sync with each other, and user choices in one of the three windows can affect what's displayed in the others. It's possible to open only one, two or all three of the windows, all with variable sizes, and to open external browser windows.

The player supports a range of the most commercially promising uses of digital media, which can be categorized as **presentation-oriented** in structure and **consumption-oriented** in intent.

Presentation-oriented: The "play/more/explore" metaphor, along with the underlying timeline-based architecture, is suited to timeline-based presentations plus supporting content.

Consumption-oriented: Commercially promising uses of such presentations are:

- the display of content for purchase (e.g. downloadable music)
- pay per view content (e.g. webcasts)
- free content plus advertising (e.g. Internet broadcasting)
- training
- corporate communications

The consumer's control over the experience is oriented towards choosing what to see or hear at a given time, and select/buy decisions—in effect, channel-changing plus purchasing.

o What Is Real's strategy for web audio?

As described above, Real bases its technology on streaming, downloading and browsing and appears to view audio primarily as a consumable commodity, or as the soundtrack for visual commodities—no doubt this focus presents the most compelling business case from Real's point of view. Features are provided for the consumer-oriented acquisition, organization and presentation of audio, and audio can synched with visual presentations. There is limited native support for interactivity beyond playing, cueing and controlling the volume of clips—multiple clips in the case of SMIL authoring. But other interactive player types, such as Flash 4, can be embedded in a RealOne presentation via HTML and SMIL.

Real has made content deals with many major media companies, such as Disney's ABC, and three of the big five record companies: AOL TimeWarner, EMI, and BMG (via the MusicNet service). Consumers can use the RealOne Player for free, but are encouraged to subscribe to pay content. The Player, both free and pay versions, also displays advertising. Music files are protected by the RealSystem Media Commerce Suite, which prevents transferring or burning to CD, and causes playability to expire after 30 days.

Audio formats supported are RealAudio, MP3, MPEG video soundtracks including MPEG4, Flash soundtracks, and Windows Media.

o What are the capabilities of the authoring tool in relation to doing this type of web audio development?

There two kinds of activity to consider: **encoding** and **presentation authoring.**

For encoding, the RealProducer tool allows the standard data compression and content tagging features (title, author, date, etc) typical of streaming media formats. The codecs used support optimization for music and/or speech, the targeting of one of a list of connection speeds, or adaptation to any speed when the content is hosted on a Real Server. Real clips may be encoded with embedded event cues that trigger the display of other content (see ii and iii, below).

Presentation authoring is based on HTML, SMIL, scripting with JavaScript and ActiveX, and the specialized markup languages RealText and RealPix. XML-based SMIL is the "parent" format for advanced authoring (see vi, below). It's a multimedia markup language, enabling HTML-style authoring using multiple clips of multiple media types, and allowing fairly good control over timing.

Authoring is based on five approaches, which can work together:

i) Embedding a Ram file. The basis for everything else. The .ram file is a text file containing the URL of a .rm file, which is the actual Real media file. This layered embedding—embed a .ram, which refers to a .rm— is used for two reasons: Browsers cannot make direct RealTime Streaming Protocol calls, which are used by Real, and the technique guarantees that Real's player is the one that is launched.

ii) URL Query String Parameters. The URL in the .ram file can also contain parameters, known as query string parameters, which can control playback of the media clip, and can also open HTML pages in the Related Info and Media Browser windows. Query string parameters can also be embedded in a .rm file (see iii, next) or placed in a SMIL file.

iii) Clip-Encoded URLs. When encoding a Real clip, it's possible to embed clip information and HTML page URLs directly in the clip. This is done using the utility *rmevents.exe*. The author creates a text file containing a list of events to be embedded, written in the form of query string parameters, and includes start and end times for each event. It ends up looking something like a video Edit Decision List (see below). *rmevents* is then used to merge the events list with the encoded clip. The clip is embedded in an HTML page (via a .ram file) as normal. When the clip is played, its embedded events list controls playback details and the launch of other HTML pages.

Below is a sample events list that might be embedded in a .rm file. It would open the page browseSite.html in the Media Browser window and leave it open for the duration of the .rm clip (1:53). Six seconds after the beginning of the clip, it would open relatedPage01.html in the Related Info (aka Context) window, and then replace it with relatedPage02.html at 22 seconds in.

```
u 00:00:00.0 00:01:53.0
&&_rpbrowser&&http://www.mywebsite.com/browseSite.html
u 00:00:06.0 00:00:21.0 &&_rpcontextwin&&http://
www.mywebsite.com/relatedPage01.html?rpcontextheight=180&rpcontextwidth
=380
u 00:00:22.0 00:01:53.0 &&_rpcontextwin&&http://
www.mywebsite.com/relatedPage02.html
```

iv) Scripting. Scripting control is via JavaScript or ActiveX extensions. The JavaScript extensions are specific to RealOne and work only when the HTML page is being displayed by the RealOne Player. The ActiveX control for the player can be accessed either within RealOne or from external pages. The JavaScript and ActiveX extensions focus on controlling playback of a clip, and opening HTML pages in the Related Info and Media Browser windows.

Some representative JavaScript/ActiveX methods:

- PlayClip() — Does what it says (both JavaScript and ActiveX).
- AddToNowPlaying() — Adds a clip to a playlist (both).
- RPOnPreload() — Used to preload contextual URLs (both).
- PreLoadURL() — Related to RPOnPreload: specifies which URLs are to be cached (both).
- OpenURLInPlayerBrowser() — Opens a URL in the Media Browser window (both).
- PlayerProperty() — Gets any or all of a list of properties, including available bandwidth (both).
- GetClipInfo() — Gets author, title, etc. information (JavaScript).
- RPOnStateChange() — Gets changes of state, i.e. Play/Pause/Stop (JavaScript).
- RPOnBuffering() — Gets percentage of buffering completed (JavaScript).
- RPOnPositionLengthChange() — Gets current position in the clip, in milliseconds. Called every 0.5 seconds during playback (JavaScript).

v) RealText and RealPix. RealText is a markup language for creating timed text presentations, such as video subtitles or closed captioning. A RealText clip is a text file, saved with the extension .rt, containing the text to be displayed, plus timing and format information.

RealPix is a markup language for creating streaming slide shows, which can contain soundtracks. RealPix files are text files saved with the extension .rp. Both RealText and RealPix presentations can be contained within a SMIL presentation.

vi) SMIL. SMIL is the core technology for advanced Real authoring. According to the RealNetworks Production Guide: "When your streaming presentation contains multiple clips—such as a video and streaming text played together—you use [SMIL] to coordinate the parts. SMIL is a simple but powerful markup language for specifying how and when clips play."[2] SMIL is also used to open HTML pages.

Since it's based on XML, SMIL is easy for HTML authors to learn. Here is a representative excerpt from a SMIL file, calling for the playing of a video clip, the simultaneous opening of an HTML page in the Media Browser window, and the opening of another HTML page in the Related Info (or Context) window 10 seconds after the start of the clip:

```
<body>
        <video src="rtsp://www.mywebsite.com/myclip.rm">

                <area href="http://www.mywebsite.com/relatedPage01.html" external="true"
begin="0s" alt="Page 1" actuate="onLoad" sourcePlaystate="play" rn:sendTo="_rpbrowser">
                </area>

                <area href="http://www.mywebsite.com/relatedPage02.html "external="true"
begin="10s" alt="Page 2" actuate="onLoad" sourcePlaystate="play"
rn:sendTo="_rpcontextwin">
                        <rn:param name="width" value="330"/>
                        <rn:param name="height" value="140"/>
                </area>
</body>
```

o What is the quality of the interactivity of the platform?

In addition to the timing- and linking-oriented capabilities outlined above, SMIL enables some control of visual space, such as layout; some animation of media clips; transition effects and hot spots. It's possible to develop reasonably complex interaction, but the format's strengths are oriented towards enhanced presentations, as opposed to deeply interactive applications such as games. As noted earlier, deeply interactive elements, such as Flash 4 movies, can be contained by SMIL. (Flash movies are first encoded in the RealFlash format.) However, a survey of posts by RealForum developers indicates that making Flash movies work reliably within the RealOne environment is not without problems (see 6, next).

o How stable & reliable is the platform?

In my experience as a new user, RealOne was quite stable, and the performance was reasonably predictable and repeatable. Advanced users report some potentially serious problems (see below).

Based on posts to the RealForum (http://realforum.real.com/cgi-bin/realforum/wwwthreads.pl), advanced applications using multiple clips in different formats can run into timing, display control, reliability and compatibility problems. In particular:

- Flash movies may not behave correctly.
- Clips of various types that are supposed to play in parallel may slip out of sync.
- Compatibility with some versions of Netscape is a recurring problem.
- As usual, JavaScript control of the player under IE on the Mac won't work (once a Mac version of RealOne is available), because of Microsoft's non-implementation of ActiveX on the Mac.

2 RealNetworks Production Guide (Rev. 7/19/02), p. 191

The problems I've found are:

- I find authoring for JavaScript control of the Real Player to be complicated by browser compatibility issues to the point where it may not be worth doing.
- When authoring with clip-encoded URLs, cueing ahead in a media clip does not update HTML pages in the other windows, while cueing backwards does. This works better using SMIL instead.
- Maintaining window dimensions while opening and closing pages and clips can require some fussing.

o What issues (technical, usability-related, economic, other) might stand in the way of adoption by consumers?

The value of the content and new functionality may or may not be compelling to consumers. A generally favorable PC World magazine review of RealOne concluded that Real's content pricing is too high (Dec. 6, 2001, see http://www.pcworld.com/news/article/0,aid,74543,00.asp).

A favorable review of RealOne in PC magazine brought negative reader responses (Dec. 17, 2001, see http://www.pcmag.com/article/0,2997,s%253D1470%2526a%253D20171,00.asp). Readers complained that the new player is larger and slower than previous versions, has downloading problems and other performance issues, and that the value proposition for pay content is poor: Many felt it is too expensive and with too many limitations, such as too-restrictive rights management and the quality issues typical of current generation PC-based media.

Real maintains that adoption by consumers is proceeding rapidly. Developers can of course create content for the free RealOne Player if they choose, with access to most functionality (the graphic equalizer, for example, is reserved for subscribers).

Other issues: Using RealOne requires downloading and installing an updated plug-in, always a source of some inertia. And Real Players tend to be aggressive in taking control of media types on the user's PC, which many heavy media users find irritating and inconvenient, especially given that Real cannot offer media from all suppliers.

o What issues might stand in the way of adoption by developers?

Authoring is fairly difficult, requiring at least an understanding of audio and video encoding, HTML and embedding, and, quite likely, SMIL, JavaScript and/or ActiveX. A variety of tools must be used for advanced authoring, some of them still command-line based. Although the RealProducer tool makes it easy to encode and embed a single media clip, no fully unified visual authoring application, analogous to Flash, exists yet.

There appear to be reliability and compatibility problems in advanced applications (see 6, above).

Developers who want to move outside the "play/more/explore" metaphor and the streaming/presentation-oriented architecture may find other technologies to be better suited to their needs.

Macromedia Shockwave

Authors: Chris Burke and David Yackley

o Pros

- Fairly extensive and robust lingo control of the essential sound functions.
- Support for external code- Xtras. Using the Beatnik Xtra with Shockwave provides more extensive interactive control of sound than most of the other web audio solutions.
- Shockwave has a large installed user base, is part of Director, a successful animation tool and it ships with virtually all browsers.

o Cons

- Shockwave exports relatively large files for download since it uses bitmapped graphics and won't compress sounds to rates of less than 32kbps.
- Lingo is more difficult to master than other scripting languages used in conjunction with web audio.
- Like Flash, Shockwave has a limit of eight sounds playing concurrently and rhythmic synchronization is sometimes unreliable.

o Recommendations

It is somewhat unclear whether Macromedia intends to replace Shockwave with Flash MX entirely. If you are using Beatnik to deliver interactive audio, Shockwave is the best choice for visuals because of its support for the Beatnik Xtra.

---

o Describe the platform.

Macromedia Shockwave is part of the Director/Shockwave Studio suite of tools. Shockwave is the form that Director presentations take on the web. It has become one of the more popular tools for creating interactive media for the web. This paper looks at Director/Shockwave 8 although most of the capabilities were also available in version 7.

Director is primarily a bitmap animation tool with an extensive, extensible scripting language (Lingo). Director movies are compressed into Shockwave files prior to web delivery. Audio contained in movies can also be converted to Shockwave audio (SWA). SWA files are essentially MP3s with a bit of additional information (cue points, etc.) packed into the file header. SWA files can also be created in other applications (notably SoundEdit 16 and BIAS Peak) and can be externally linked and streamed in conjunction with Shockwave movies. In addition Version 1 SWA files can be created with Audioactive Production Studio by Telos Software. Version 1 SWA files do not contain cue point information.

o What kind of experiences can this platform deliver over the web? What is and is not possible using the platform?

Shockwave provides the standard requirements for web audio:

- functional sounds- buttons, transitions, feedback
- background sound- music beds and loops, ambient sound

Frequently seen applications of audio in Shockwave include:

- interactive games
- linear movies or cartoon animations
- web art

Control of sound parameters include:

Sound Cast Members

- specify the startTime and endTime for any sound.
- monitor the playback status of a sound channel.
- loop a sound a specific number of times or infinitely.
- specify specific loop start and end times.
- play, pause and resume sound playback on each sound channel.
- rewind a playing sound back to the startTime.
- set the current time of any playing sound.
- shift the playback rate by semi-tones in D8 and by cents in D8.5.
- use Cue Points in AIFF, WAVE and SWA files to trigger event handlers in Lingo

Shockwave Audio (streaming)

- select a sound channel in which to play
- play
- pause
- resume (by using the Play command again)
- stop playback
- set the preLoadTime
- preLoadBuffer
- use Cue Points in AIFF, WAV and SWA files to trigger event handlers in Lingo

and using the Sound Channel Lingo commands:

- volume- static setting
- volume- fades (up and down)
- pan

Properties can be tracked:

- elapsedTime
- currentTime

Other built-in properties are available:

- startTime
- endTime
- channelCount
- sampleCount
- sampleRate

The maximum number of simultaneous sounds is eight including streaming sounds. Simultaneous streams are limited to the maximum of four set in most browsers by default.

Sounds are mixed at playback using either QuickTime, DirectX or the somewhat less than adequate MacroMix. Shockwave requires a downloadable browser plug-in, but it's bundled with most browsers.

In use, this compliment of interactive parameters and properties can be combined to make some interesting interactive audio experiences. However, they fall short in some areas:

- With the sound cast members (non-streaming), there can be a noticeable lag when the sound decompresses into memory for the first time. However, Lingo can be used to set up preloading.
- Certain built-in properties can only be tracked while playing. This requires some extra code to work around.
- Windows machines without QuickTime or DirectX default to the MacroMix.DLL for mixing the sounds. This can cause some significant latency when mixing multiple sounds.
- For streaming sounds, the sound channel's Pause and Play commands conflict with the SWA member's commands. Pausing the sound channel will pause the sound, but a subsequent Play command will not resume playback. The SWA member's Pause and Play commands should be used instead.

o How stable & reliable is the platform?

Director has been a standard in multimedia production for years and is therefore quite stable. Additionally, Shockwave has a vibrant community of developers behind it and thus many stability issues have been addressed and work-arounds have been adopted. Playback is consistent on most platforms except on WIndows if QuickTime and Direct X is not available. As mentioned above, MacroMix has some latency.

o What issues (technical, usability-related, economic, other) might stand in the way of adoption by users? By developers? By content providers, clients, web hosts? How many users does the platform already have? Is it well established and ubiquitous, or marginal?

This software is already well ensconced in the world of CD-ROM and Web development. Lingo is perhaps a bit more difficult to master than, say, Flash Action Script.

Director/Shockwave Studio has a higher purchase price than many other web audio authoring apps, however it does considerably more than most. Developer adoption rates indicate that there are few effective impediments to using this app. Users have to download and install the free Shockwave browser plug-in or "player" (about 400k). While some clients shy away from any use of plug-ins, Shockwave ranks amongst the most ubiquitous. Macromedia claims there to be 270 million installations of the Shockwave Player and that it comes pre-installed on all new PCs and Macintosh computers.

o What is the platform company's strategy for web audio?

Director/Shockwave benefits from the attention Macromedia has paid to the audio tools. However, Macromedia has what some might consider a mixed record concerning audio products, Sound Edit 16 and Deck being two examples of poorly or non-supported Macromedia audio software. Being part of a successful animation package, Shockwave audio is in a good position to survive and grow unless of course Macromedia decides to replace it entirely with Flash. There have been some advances in recent versions of Director. More file types are now supported and the advent of the semi-tone shift feature has broadened the appeal of this product for audio developers. Macromedia has been quite successful at creating an interactive multimedia standard, or pair of standards with Shockwave and Flash. The plug-ins are now tied together so that there is less to download if a user wants both. Further, Macromedia has made a priority of creating a secure environment for e-commerce applications in Shockwave and they offer free Shockwave licensing for CD-ROM distribution and corporate intranet.

o What are the capabilities of the authoring tool in relation to web audio development? What is the quality of the interactivity of the platform?

Shockwave does quite a bit with audio. Flash 5 is perhaps a bit more ahead in terms of interactive controls. Of course, Flash can be embedded in Shockwave to create advanced interactive games that make use of Flash's sound object, but this usually adds lots of processor overhead and results in poor performance. Without Flash, there are still some interesting uses of audio in Shockwave. While Shockwave's semi-tone shift feature is useful in musical applications, Flash seems to be better at synchronizing multiple sounds. This gives Flash an edge in this area. Overall Shockwave apps are pretty responsive sonically but are somewhat limited for musical uses unless augmented with the Beatnik Xtra.

One advantage Director enjoys over Flash is its support for external code (also called Xtras). Beatnik Xtra, for example, gives Director users access to all the functions of the Beatnik Audio Engine, but without the inherent perils of using a plug-in, particularly messaging via JavaScript in browsers. Also, Xtras download seamlessly when required, so the user isn't forced to endure the process of plug-in installation.

o What kind of interactivity are users actually interested in? How does that relate to the capabilities of this platform?

What is sometimes read as a lack of user interest in interactivity may often be frustration with the web as a medium in it's infancy. As developers, it is up to us to define the future of interactivity, keeping an eye on user feedback along the way. Like Flash, Shockwave enables developers to create limited sound experiences. Macromedia will need to keep pace with the interesting developments in web audio coming from the other players by enhancing the Shockwave audio tool set with more synchronization based controls. As noted above, one can extend Shockwave audio capabilities by using it in conjunction with the Beatnik Xtra. While this has resulted in some unique sound experiences on the web, it involves using something like a second plug-in and is out of the question for many clients. If Shockwave was outfitted with the synchronization and musical strengths of Beatnik without requiring an Xtra, we would have a very powerful web audio application.

QuickTime

Author: Valentin Schmidt

o Pros

- Powerful, widespread, stable and reliable multimedia platform supporting, that supports, in addition to it's own extensible track-based file format, a variety of digital audio formats (including MP3) as well as standard MIDI files, DLS, SF2, AAC, SMIL and MP4
- Integrated high-quality softSynthesizer ensuring similar sound quality across different hardware and platforms
- Although not like Flash or Shockwave, the platform offers some strong interactive features "within" QuickTime movies (e.g. interactive sprite- and Flash tracks)

o Cons

- No consistent JavaScript-interface for browser plug-in across browsers and platforms, and no plug-in methods for direct access to QuickTime instrument tracks and the QuickTime softSynth.
- No compression and 256k file size limitation for custom instruments, and no auto-download/auto-install for soundbank-files (*.dls, *.sf2).
- Limited distribution on Windows machines; huge download.

o Recommendations

Although QuickTime offers quite a few possibilities for web audio, Apple could easily accomplish to make it much more attractive for web audio developers by making some simple enhancements and improvements.

---

o What is and is not possible using the platform?

This question is hard to answer, as QuickTime is a whole universe. My general impression is that QuickTime offers more possibilities for web audio than what you actually find out there, or at least you have to search quite long to find any more advanced application. I hope this report helps to identify the reasons for this.

As a starting point here a list of the various facets of QuickTime:

QuickTime as a File Format

At its core, QuickTime is a patented, extensible track-based file format (which was selected by ISO as the basis for MPEG-4). Each track delivers a different element of content, such as video, audio, interactivity (such as Flash), HTML behavior, and much more. And as new technologies emerge in the digital media space, the industry can develop new track types. A single file can be used for streaming over the web, for downloading from a web server, or for local playback from a CD to both Windows and Mac users. With the increasing number of track types and the further extension of the wired sprite API, QuickTime is maturing into a powerful multimedia format that already matches Director/Shockwave on many levels.

QuickTime 6 supports MPEG-4. It is able to read and write ISO-compliant MP4-files (*.mp4) as well as contain MP4-tracks within a mov file (similar to mp3-support)."

According to QuickTime, the implemented MPEG-4's audio component AAC is superior to MP3 (smaller file size and lower data rate, better sound quality), and might replace MP3 one day as as the digital audio standard for music on the web. Apple has only announced implementation of the video and audio compression algorithms, but none of the systems part of MPEG-4, like BIFS. There won't (and might never) be any support of MPEG-4's Structured Audio components.

Most interesting for web audio are embedded or linked QuickTime-movies containing one or more (streaming) sound or music tracks only, as well as movies containing sound or music tracks in combination with interactive track types like wired sprites or flash tracks.

QuickTime as Multimedia Engine and Player

QuickTime supports more than 50 industry-standard media file types. Once registered as a Pro-Version the QuickTime Player becomes a powerfull authoring tool itself. Since Version 5 the QuickTime Player's interface can be customized by Media Skins embedded inside a media file (.mov). With HotPicks and QuickTime TV channels Apple also implemented some "push functionality" into the Player.

QuickTime as scriptable Browser-Plug-in

Through the QuickTime plug-in, QuickTime's digital video streaming capability is extended to all popular web browsers, including Internet Explorer, Netscape Navigator, and America Online. The plug-in supports over thirty different media types and makes it possible to view over 80 percent of all Internet media. QuickTime also features advanced web streaming capabilities, such as movie "hot spots" and automatic web page launching.

QuickTime as GM compatible and extensible MIDI Software Synthesizer

Included with QuickTime is the QuickTime music synthesizer, a software-based GM music synthesizer which plays sounds using the built-in audio of a Macintosh or Mac OS–based computer or the sound card or built-in audio circuitry of other computers. MIDI files imported into QuickTime are stored as Music tracks. The Player can play both, quicktime movies containing music tracks and external standard MIDI files (in other words, MIDI files can be imported and converted „on the fly").

Beginning with version 2, QuickTime supported the incorporation of custom samples into MIDI tracks but this functionality was never evangelized, and the authoring tool (Atomic Editor) was never officially supported or released (although, until version 4, it was secretely embedded in the QuickTime Player Pro and could be activated by a special key and combination: Get Info→Music Track→Option(Alt) Key + double-click on Instrument→ Edit-Button). Since QuickTime 5 the build in sound banks can be extended by sample banks in Downloadable Sounds (DLS) and Sound Font 2 formats.

QuickTime as Application Programming Interface (API)

QuickTime provides the underlying multimedia engine via powerful APIs. The C QuickTime API includes more than 2000 function calls that QuickTime applications can make. For instance, the QuickTime music architecture (QTMA) provides a set of functions that allows applications and other software to play individual musical notes, sequences of notes, and a broad range of sounds from a variety of instruments and synthesizers.

The entire QuickTime API is fully accessible through Java (QuickTime for Java API). With Java, you can write your own QuickTime-compatible applications, or run Java applets from QuickTime over the web inside a browser. The QuickTime API is implemented as a set of Java classes in Quick-Time for Java. These classes offer equivalent APIs for using QuickTime functionality on both Mac OS and Windows platforms.

o What kinds of experience can we deliver over the web?

Here is a list of some possibilities to use QuickTime (Audio) over the web:

- background music, ambient sounds, to some degree interactive audio environments
- sound effects, response sounds, jingles
- downloadable music files (.mov)
- (interactive) presentations
- enhanced TV
- web radio, custom player, "enhanced jukebox".
- news broadcasts (speech)
- interactive educational programs, web based training (WBT)
- interactive audio toys
- simple games

o What Is Apple QuickTime's strategy for web audio?

Development

Apple focuses on IP-based media consumption and streaming. By implementing MPEG-4 (and its AAC audio component) they probably hope to further extend their market share and maybe gain some advantage over their competitors (Microsoft has not yet announced any plans of supporting ISO-conform MPEG-4, which is not to be confused with Microsoft's own so-called MPEG-4 codec; Real will only support MPEG-4 with the help of a third party plug-in). It's not easy to identify any specific strategy for web audio. The streaming server's support of MP3 (and MP4) audio streaming, and the QuickTime 5 player's support of ShoutCast show that one concept they have in mind is "web radio".

The fact, that QuickTime still does not allow to compress custom instruments or sample banks (like Beatnik's RMF and soon XMF), and that it's MPEG-4 support does not include any concepts like Structured Audio show that Apple does not attach importance to this field of interactive web audio (as long as there is no market for it).

Licensing

- Free QuickTime Player (and plug-in) for end user (moderate price for QuickTime Player Pro)
- Darwin Streaming Media Server open source.
- No licensing fees for the number of simultaneous streams served (via Mac OS X Server or the Darwin Streaming Media Server)

o What are the capabilities of the authoring tool in relation to doing this type of web audio development?

QuickTime Authoring

Most traditional video editing software relying on QuickTime still offers none or only minimal support for the non-traditional media-handling and wired sprite capabilities of QuickTime. The following is a list of tools that support the interactive features of QuickTime at least to some degree.

QuickTime Player Pro

While QuickTime Player Pro is definitely not an authoring tool for interactive movies, it does offer some access to interactive features:

- Tracks can be copied and pasted from one movie into another, including "interactive" track types like Sprite tracks, Flash tracks, Text tracks with HotSpots or Music tracks.
- Sprite media can be exchanged (in existing Sprite tracks)
- The wired actions of a particular Sprite track can be disabled/enabled.
- Some interactive elements like Flash files or HotSpots can be imported and integrated in existing movies
- MIDI-Instruments can be changed, custom instruments can be created from audio samples
- In QuickTime 4 the Softsynth had a secret GUI (Get Info→Music Track→Option(Alt) Key + double-click on Instrument→ Edit-Button), which allowed to shape your own audio samples using Filters, Envelopes and LFOs (this feature seems to have disappeared in QuickTime 5).

Spritz

A Mac-only freeware tool offering a first step into the possibilities of wired actions. Custom events are supported, so sprites can send messages to each other, and a fair assortment of actions are available. The current version is too unstable and has too many rough edges for serious use. Nonetheless, it is good enough for simple interactive projects such as slideshows. There is no interactive access to the QuickTime SoftSynth.

Cleaner 5

The encoding tool Cleaner 5 by Discreet allows you to create interactive presentations with what they call EventStream actions. These events and your original media can then be output to a streaming QuickTime video (and to some degree also to the other leading streaming formats, RealSystem or Windows Media).

EventStream actions:

- Display Text
- Open URL
- Hotspot
- Keyframe
- Chapter
- Replace Movie
- Go to Time
- Pause
- Play
- Web Poster

Cleaner taps into these features to let you add stream navigation, synchronize HTML to streaming media, embed links and interactive hotspots, and more. There is no interactive access to the QuickTime SoftSynth.

GoLive

The web design, production, and management tool Adobe GoLive has an integrated interactive QuickTime editor which allows to import, edit and arange several track types (in multilayer timeline editor, like in flash, director or livestage). Wired sprites can be created by assigning actions to key frames in a sprite track. There is no scripting language like in LiveStage, but those actions can only be chosen from a set provided by the program.

Audio related actions are changing of master volume, track volume and track balance. Unfortunately there is no interactive access to the QuickTime SoftSynth.

Flash

There are several ways to add interactivity to Quicktime movies with Flash:

- First, an interactive Flash file can be created in Flash, exported as swf, and imported and added as a Flash track to an existing QuickTime movie in QuickTime Player Pro or LiveStage. In LiveStage it's even possible to overwrite the ActionScript inside the swf (e.g. the behaviour of Flash Buttons) with QScript Actions.

- Second, it's possible to import a QuickTime movie in Flash (it is not copied into the Flash file as most other media elements are, but instead It is imported as a link), add some vector animations, navigation controls or user interface widgets, then export the result as a Quicktime movie. For Flash soundtracks, you have the option of moving audio out of the Flash Media Handler track and into the QuickTime audio track. This lets you use QuickTime audio compression codecs that aren't available inside Flash.

iShell

iShell is not an authoring tool for editing interactive QuickTime movies, but an object-oriented authoring environment (especially for creating customized Internet applications that launch from the browser or desktop) that uses QuickTime to handle all media elements. Unlike Macromedia Director it offers full QuickTime 4 support for all advanced QuickTime features-including streaming, QuickTime filters, and all advanced QuickTime effects. The final application is saved in a proprietary format, to run it over the internet the online audience has to download a 1MB (compressed even smaller) custom application (a renamed version of the iShell runtime). When the first file is referenced off the server, the application will run.

iShell does not offer any interactive access to the QuickTime Softsynth. However, the iShell SDK can be used to expand iShell by creating modules for added functionality in both the Editor and Runtime. A programmer familiar with the QuickTime APIs could probably write such a plug-in, e.g. for playing single notes via the QuickTime Softsynth.

Electrifier Pro (discontinued)

Electrifier Pro is a QuickTime compositing tool for assembling simpler, low bandwidth interactive QuickTime movies from pre-existing content. It affords simple interactive sprite editing, but the supported wired atoms are limited. Only a few events are supported, with no ability to create custom messages, variables, logical control structures etc. There is no interactive access to the QuickTime SoftSynth.

LiveStage

At the moment, LiveStage is the only software that can honestly be called a QuickTime Authoring Environment for interactive QuickTime movies. It's a serious authoring tool with a complete scripting language called *QScript. Qscript* is not an Apple technology, it was developed by totallyHip. Nonetheless *Qscript* is intimately connected to QuickTime. Each line of *Qscript* gets converted to 'wired action atoms' (bytecode tokens) by the LiveStage compiler when you export or preview a movie. These 'atoms' are then interpreted by the QuickTime engine as executable instructions when the movie is loaded into a QuickTime aware container. No trace of your *Qscrip*t code remains in the exported movie, only the wired action atoms which it represents.

LiveStage even allows to construct custom handlers, so sprites can send messages to each other, but parameter passing and return values are not supported. (This is another shortcoming in the QuickTime API, and is likely to be added sooner or later).

LiveStage is also the only software offering some interactive access to the QuickTime musical instrument architecture conform to Apples's QuickTime API. The relevant *QScript* actions are:

- PlayNote(Instrument,Delay,Pitch,Velocity,Duration)
- SetControler(Sample,Instrument,Delay,Controller,Value)

Conclusion: What's missing?

- Authoring Tools offering deeper access to advanced QuickTime audio and MIDI features (LiveStage 4?)
- Other wired movie authoring tools that would really compete with LiveStage (e.g. by Apple itself)

Instrument Editors for QuickTime

PolyPhontics

PolyPhontics (http://www.bestsoftwaredesign.com/polyphontics.html) is a Mac-only DLS/SoundFont-Editor which is especially developped to work with QuickTime. It supports the latest SoundFont(SF2) and Downloadable Sounds(DLS) Specifications.

Audio Compositor

Audio Compositor (http://audiocompositor.home.att.net) is an Instrument/DLS/SoundFont-Editor for Windows. Unlike PolyPhonetics, It’s not designed to work with QuickTime in particular, but nevertheless it allows to create and edit soundbanks which can be used with QuickTime.

Other Authoring Tools

Most authoring tools for video editing (QuickTime Pro, Premiere, FinalCut Pro, iMovie,…), Special Effects (AfterEffects, Commotion Pro, Studio Artist,…) or Compression (QuickTime Pro, Cleaner, QDesign Music Codec 2 P.E.,…) support basic audio functions as they relate to video editing (volume, pan, etc.). While all sound editing software on Mac OS (SoundEdit, Peak, …) support QuickTime, it's astonishing that the major sound editors for Windows (SoundForge, WaveLab, …) still don't.

Conclusion: What's missing?

- QuickTime support of sound editing software under Windows
- Software Sequencer (and more Instrument Editors) fully supporting QuickTime Music

o What is the quality of the interactivity of the platform?

The easiest way to use QuickTime for web audio is by embedding or linking QuickTime movies containing one or more sound or music tracks in web pages, as repeating background loops or as progressive downloads (pseudo-streams).

To put several music tracks on one page, rather than having them all attempting to download at once, it's possible to use the poster movie technique. This exploits Quicktime's ability to link one movie to another, so that the second loads in the space occupied by the first when your visitor clicks on it.

The following section deals with more sophisticated ways of using QuickTime.

QuickTime Interactive

Sprites, Sprite Animation and Wired Movies

The most interesting track type concerning interactivity is the sprite track. Sprite tracks with motion paths first appeared with QuickTime 2. Since Quicktime 3 sprites can be provided with interactivity, something which Apple call 'wired action atoms'. System events such as mouse-clicks and keystrokes can be wired to a sprite in a sprite track and trigger more than 70 preset actions, including nested conditional and recursive structures.

Wired movies are QuickTime Movies that have tracks that contain scripts that run in response to a user's actions. Wired movies were originally introduced in QuickTime 3 and have been enhanced in QuickTime 4 and 5. These interactive movies can be played in any Web browser using the QuickTime plug-in, and in all applications as long as they use the QuickTime movie controller API.

Wired movies let you create QuickTime movies that are interactive. User input is translated into QuickTime events. In response to these QuickTime events, actions may be performed. Each action has a specific target, which is the element in a movie the action is performed on. Target types include sprites, tracks, and the movie itself. A few actions do not require a target. Actions have a set of parameters that help describe how the target element is changed.

Typical wired actions--such as jumping to a particular time in a movie or setting a sprite's image index--let you create a sprite that acts as a button. In response to a mouse down event, for example, a wired sprite could change its own image index property, so that its button-pressed image is displayed. In response to a mouse up event, the sprite can change its image index property back to the button up image and, additionally, specify that the movie jump to a particular time. Wired sprite tracks may also be used to implement a graphical user interface for an application. Actions associated with sprites in a sprite track, for example, can control the audio volume and balance of an audio track, or the graphics mode of a video track. The complexity of the **wired sprite API** is only now beginning to be realized. With its appearance QuickTime has started to mature into a powerful multimedia format which already matches Shockwave on many levels.

Embedded movies

Embedded movies are implemented through the track type "movie track". A movie track maintains a list of movies that may be loaded and played within the track. The movie track plays only one movie from the list at a given time. This list is initialized from data in a movie track sample, but the list may be augmented at runtime.

For example, a movie could contain an animation track, perhaps a sprite or Flash track, and an embedded movie track for the video content. In this example, the root movie's time base would control the animation, but the video's rate could be controlled independently from the animation's rate. Wired actions could be sent to the embedded movie when a user clicks on buttons in the animation track. The wired actions could play, pause, and fast forward the video, or switch to a new one. Another way you can take advantage of independent time bases is to allow long audio tracks to be triggered interactively. For example, if you create a game with sound effects and background music that need to be played back at times defined by events that occur in the game, you can use an embedded movie for each audio track. The advantage of using an embedded movie instead of a music track with a custom sound is that the entire sound does not need to be loaded into memory, so it is appropriate for longer sounds. The disadvantage is that you may not control it similar to a MIDI instrument.

One way to use embedded movies is to break projects into components, allowing portions to be reused in other projects, and simplifying the authoring process. In QuickTime, this technique could be used in a Web browser using external movie-to-movie communication. In QuickTime, a single movie can be created that is playable in the QuickTime Player or any application that plays QuickTime movies.

HREF tracks

HREF tracks can make the movie's display area into a clickable (or automatically executed) link that points to different URLs at different times during playback. The QuickTime plug-in has full access to the QuickTime importer, which means you can put a specially marked up text file on a web server - for example with a CGI, ASP or PHP script - and have the QuickTime plug-in interpret it on the fly as a time-based hypertext document, linking to other similar files.

Music Tracks

The QuickTime API (only) provides 2 actions for music tracks:

- kActionMusicPlayNote
- kActionMusicSetControler

At the moment, only the authoring tool *LiveStage* allows to wire those to a user's action. The corresponding QScript actions are:

- PlayNote(Instrument,Delay,Pitch,Velocity,Duration)
- SetControler(Sample,Instrument,Delay,Controler,Value)

Load web pages

With Plug-In Helper, you can make any movie or its video tracks into clickable links that load web pages or QuickTime movies into a web browser, QuickTime Player, or the QuickTime plug-in.

Interact with Flash

Flash tracks can incorporate interactivity supported by Macromedia's Flash, including custom movie controllers. Wired sprite actions can be linked to Flash animation.

Messaging

Inter-movie messaging is a way for QuickTime wired movies to control other QuickTime movies running in the same container (something like 'tell window' in Director). It is handled transparently by the QuickTime Player or browser plug-in without any need for LiveConnect. Wired sprites can also send string messages to their container applications. For example, the QuickTime Plug-in in a browser can invoke a JavaScript routine in the HTML file that has embedded the QuickTime movie.

QuickTime Streaming

QuickTime as Streaming Format

With the release of Quicktime 3, Apple joined Microsoft in the quest to give RealAudio a little competition. The QDesign Music and Qualcomm Purevoice codecs can make the files small enough to stream effectively. Progressive download ("Fast Start") allows users to listen to media as it is being downloaded from a standard web server (such as Apache) to their hard drives. This method continues to work well for short-form media where file size is limited, ensuring high-quality playback regardless of bandwidth. QuickTime has adopted a system of having multiple files encoded at different bit rates from the same source, then one "master" file to act as a pointer. When you set the preferences in your Quicktime player to indicate your connection speed, that information will be sent to the server and you will be given the file that best matches your connection. Not quite as sophisticated as Real's SureStream, it will work without any special server with HTTP.

QuickTime supports RTP/RTSP streaming of video, audio, text, and MIDI. To stream a movie with other media types, such as sprites, you should use HTTP. Using the industry-standard Real-Time Protocol/ Real-Time Streaming Protocol (RTP/RTSP), QuickTime can be streamed from the *QuickTime Streaming Server* (Mac OS X Server) or the *Darwin Streaming Media Server* (see next section) as well as from RealNetwork's *RealSystem™ Server 8 (RealServer™ 8)*.

QuickTime as Streaming Server Platform

QuickTime Streaming Server is the only open source, standards-based streaming server in the industry. Designed for Mac OS X Server, *QuickTime Streaming Server* is also available as an open source server called *Darwin Streaming Server*. Versions are available for Linux, Solaris, Windows NT/2000, and FreeBSD. And because it is an open source technology, Darwin Streaming Server can be ported to other platforms if desired just by modifying a few platform-specific source files. Both are free. And unlike other streaming servers, there's no per-stream charge: QuickTime does not charge licensing fees for the number of simultaneous streams served. This freedom from "server tax" can add up to enormous savings over other media platforms, especially for customers with high-volume media distribution requirements. Although these streaming servers are free, you don't sacrifice performance or scalability. Serve thousands of simultaneous streams from a single server, or scale to much more with multiple servers.

The Darwin Streaming Server includes a *PlaylistBroadcaster* utility that functions like RealServer's G2SLTA: it streams a prerecorded clip to a server, which then delivers the stream as if it were a live event. New features of the QuickTime Streaming Server 4 and the Darwin Streaming Server 4 include:

- Skip Protection which protects streams from disruptions in the Internet, resulting in better-quality playback
- Native MPEG-4 streaming: Standard hinted MPEG-4 files can be served directly, without converting to .MOV files.
- MP3 audio streaming (e.g. for internet radio stations): MP3 files can be served to clients that support MP3 streaming via http, such as iTunes and SoundJam MP.

QuickTime as Streaming Client

QuickTime 5 supports ShoutCast. This allows ShoutCast stations to create their own custom players without relying on generic media players. QuickTime gives you the ability to send the proper bitrate to your listeners automatically, add your own links to the player, or embed the ShoutCast stream directly inside your web page.

Another interesting feature of QuickTime 5 are the Media Skins, which let users embed a custom interface (skin) to the QuickTime Player in each media file (.mov).

Interactive Streaming with SMIL

A SMIL document specifies what media elements to present, and where and when to present them. Each media element is specified by a URL. A SMIL presentation can use any media elements that QuickTime can play, including still images, audio, MIDI, text, QuickTime movies, sprite animations, live streams, VR panoramas and VR object movies. The URL of a media element can point to local or remote media, using any format that QuickTime supports, including file access, HTTP, and RTSP. SMIL can combine streaming movies with local or Fast Start movies without having to edit or combine them in QuickTime Player. A common use of SMIL is to combine an ad banner with a streaming QuickTime movie.

The following are way to enhance SMIL presentations with interactivity:

<u>Clickable Links</u>

You can make any visual media element in a SMIL document into a clickable link by using the <a> </a> tags. You can:

- open an URL in the default browser window, or (with help of SMIL Extensions) a specific browser frame, specific browser window, or QuickTime Player.
- replace the current SMIL presentation in the plug-in or QuickTime Player (whichever is active). The URL must specify something that QuickTime can play.
- jump to a named point in the current presentation (needs SMIL Extensions).

Throwing a Switch

You can automatically present different elements to different viewers using the <switch> </switch> tags. SMIL supports a set of user attributes, such as screen resolution, color depth, maximum data rate, and language. Groups of elements can be listed between <switch> and </switch> tags. QuickTime selects one element from the list based on user attributes, much like QuickTime's alternate track and alternate movie mechanism. QuickTime supports the following user attributes:

- system-bitrate
- system-language
- system-screen-size
- system-screen-depth

"On the fly" generation

Because SMIL presentations are described by text files, they can also be generated "on the fly" (e.g. dependent on user input like a form) using any script language that creates text files (CGI, ASP, PHP, Perl, …).

Movie Tracks

SMIL also provides an easy way to make use of a new QuickTime feature -- tracks in QuickTime movies that are QuickTime movies. These are called movie tracks, and are similar to text tracks, video tracks, or sound tracks, but point to other movies. Movie tracks have their own time base, so they can play forward, play backward, repeat, or loop, independently of the movie that contains them. When QuickTime imports a SMIL document, it creates a movie track for each SMIL media element.

Using wired sprites, you can start or stop a movie track at any time -- while the rest of the movie is paused or plays normally -- which is useful for creating interactive sound or interactive animation. You can put sound or animation in one movie, then create a wired sprite movie that controls the sound or animation using intermovie communication. Then you can stitch the movies together into a single presentation using SMIL.

QuickTime -Plug-in

Through the QuickTime plug-in, QuickTime's digital video streaming capability is extended to all popular web browsers, including Internet Explorer, Netscape Navigator, and America Online. The plug-in supports over thirty different media types and makes it possible to view over 80 percent of all Internet media. QuickTime also features other advanced web streaming capabilities, such as movie “hot spots” and automatic web page launching.

Since Version 4.1 QuickTime enables JavaScript control of the QuickTime Plug-in in Netscape browsers (<6). Version 5.02 provides more than 100 methods to control QuickTime from JavaScript. Although the QuickTime-Plug-in worked in older versions of Microsoft's Internet Explorer, there has never been any JavaScript control. Since versions 5.5 SP2 and 6.0 Microsoft Internet Explorer for Windows no longer supports Netscape-style plug-ins at all, such as the plug-in installed as part of QuickTime 5.0.2 and earlier versions. To restore compatibility, Apple has provided an ActiveX control (version 5.0.3). This ActiveX-control at the moment does not provide the methods of the Netscape-Plug-in (only the 4 methods listed below). However, the next ActiveX-control version of the QuickTime-Plug-in (Version 5.1, in beta state at the moment) will provide them. On the Mac, there is no JavaScript control of the plug-in under IE at all, because Microsoft has not implemented ActiveX on the Mac. And there is still no way to control the plug-in in Netscape 6 (both platforms), as there is neither a XPCOM-version for Mozilla/Netscape 6, nor a Mozilla 0.9.2+/Netscape 6.1+ compatible LiveConnect version.

The following list shows some methods that can be useful for web audio:

- GetSpriteTrackVariable
- SetSpriteTrackVariable
- SetTrackEnabled
- GetTrackEnabled
- SetMute
- GetMute
- GetIsLooping
- SetIsLooping
- GetStartTime
- SetStartTime
- GetVolume
- SetVolume
- GetTime
- SetTime
- SendSpriteEvent
- Rewind
- Stop
- Play

These methods allow some interactive control of embedded QuickTime movies containing sound or MIDI tracks. Even more powerful control is available in combination with wired movies: mouse clicks or rollOvers in the browser window can communicate with the wired movie via the SetSpriteTrackVariable-method and trigger custom handlers inside the movie, for example to play notes. (there is a method SendSpriteEvent which would maybe provide an even better way to achieve this, unfortunately there is no documentation or support available on how to use it).

Conclusion: What's missing?

- Scriptable versions of the QuickTime plug-in for Internet Explorer (ActiveX), Netscape 4.x (LiveConnect) and Netscape 6.x (XPCOM or adapted LiveConnect-Version) on both major platforms that provide exactly the same functions, properties and events.

- Better documentation for the JavaScript control of the plug-in - some functions like sendSpriteEvent are not documented at all.
- Plug-in methods for direct access to QuickTime instrument tracks (like PlayNote, ProgramChange, Mute and Solo of individual channels/instruments, as is possible in Beatnik)

QuickTime Music Synthesizer

Included with QuickTime is the QuickTime music synthesizer, a software-based music synthesizer which plays sounds using the built-in audio of a Macintosh or Mac OS–based computer or the sound card or built-in audio circuitry of other computers. Since QuickTime 5 the build in sound banks can be extended by sample banks in Downloadable Sounds (DLS) and Sound Font 2 formats. Unfortunately, this feature's use for web audio is rather limited, because they can't be installed automatically, but have to be put in the QuickTime folder inside the System folder either manually or by an installer application, and they can't be compressed (not Apple's fault, though, but a feature not included in the DLS standard). The latter also applies for sound samples embedded as custom instruments. As they can't be compressed, and the file size is limited to 256 k, their usefulness is rather limited, too.

On Macintosh computers, the QuickTime softSynth can be used as the default MIDI synth for other applications (e.g. via OMS). On computers playing MIDI files with any MIDI plug-in other than QuickTime, (e.g. with Shockwave plus sequenceXtra, MIDIio Xtra or the older Yamaha MIDXTRA) user interactions that generate MIDI events can use QuickTime for web audio.

Conclusion: What's missing?

- Sound banks compressable with strong compression algorithms like mp3 (maybe this could be integrated in the next version of DLS?)
- Sound banks embedded in .mov file, or automatic download and installation of sound banks (like for example shockwave-safe xtras in shockwave)
- Compressable custom samples (strong compression like mp3)
- Custom samples without limit of file size
- Maybe in the future: XMF-support?
- Realtime generated MIDI events like controllers which affect the behavior of playing MIDI tracks (the Beatnik idea)
- Using the QuickTime softSynth under Windows as the default synth by other applications; its appearing in list of available MIDI-instruments in the system's multimedia-settings

QuickTime API

QuickTime API (C)

The use of the C version of the QuickTime API for web audio can only be indirect, for instance by allowing third party developers to build applications and tools that can be used to improve the process of content creation. For example, QuickTime Music is crying out for an tool similar to the Beatnik Editor (even more after Apple seems to have dropped the "secret" instrument editor in version 5) which would allow you to drop new samples in while composing, edit the sample's instrument settings and then embed them in the final movie. Or an authoring tool for wired movies like LiveStage, but focusing on audio/MIDI…

QuickTime for Java API

The entire QuickTime API is fully accessible through Java. With Java, you can write your own QuickTime-compatible applications, or run Java applets from QuickTime over the web inside a browser.

QuickTime for Java enables Java and QuickTime programmers alike to take full advantage of QuickTime's multimedia capabilities. At its simplest level, QuickTime for Java lets you write a Java applet and run that applet on a variety of platforms. Java can be used in an applet, for example, to make Web pages more interactive so that users will be able to interact with those pages in ways they havn't before. At an advanced level, you can write applications that composite images, capture music and audio, create special effects, and use sprites for animation.

The QuickTime for Java API consists of two layers. There is a core layer that provides the ability to access the QuickTime native runtime libraries (its API) and an Application Framework layer that makes it easy for Java programmers to integrate QuickTime capabilities into their Java software. The QuickTime API is implemented as a set of Java classes in QuickTime for Java. These classes offer equivalent APIs for using QuickTime functionality on both Mac OS and Windows platforms.

The QuickTime for Java API supports all fully compliant Virtual Machines (VMs). On the Macintosh, it supports Apple's Macintosh Runtime for Java (MRJ) 2.1 or later; under Windows, JDK 1.1 or later. The Java VM used must be fully compliant with at least the JDK 1.1 specification and implementation from Sun Microsystems. QuickTime for Java can also run in an applet, provided that the browser or applet viewer has one of the supported Java VM's chosen. Currently this requires the use of the Java Plug-in on Windows Internet Explorer or for Netscape 4.x browsers because those browsers do not provide a fully compliant Java 1.1 VM. On the Macintosh, you can use the Internet Explorer version 4 browser with an applet tag if MRJ 2.1 is chosen as your default VM. For Netscape Navigator version 4 on the Macintosh, QuickTime for Java applets must be viewed using the MRJ Plug-in.

QuickTime VR

QuickTimeVR with Sound

Both VR Panormas and Object Movies can be inhanced with sound. Adding sound to an object movie is a bit more tricky. You can easily add narration or sound effect to a given view, but for background music, the problem is that the sound is played only when its part of the movie time line is played, and an object movie jumps around in the movie time line in a nonlinear way as the user drags the image. The best way to add background music to an object movie being played in a browser is to embed an audio-only movie in the same Web page. The best way to add music to an object movie being played in QuickTime Player is to use SMIL.

To play a time-based track with the object movie, you must synchronize the sample data of that track to the start and stop times of a view in the object image track. For example, to play a different sound with each view of an object, you might store a sound track in the movie file with each set of sound samples synchronized to play at the same time as the corresponding object's view image. (This technique also works for video samples.) Another way to add sound or video is simply to play a sound or video track during the object's view animation; to do this, you need to add an active track to the object that is equal in duration to the object's row duration.

What's New in QuickTime 6.3

Here are the major new features of QuickTime 6, as described by Apple:

3GPP Support

QuickTime 6.3 delivers extensive support for 3GPP, including video, audio, text, and native .3gp file format support. Because 3GPP is now a part of the core architecture of QuickTime, you can import, export, and play back .3gp files just as you do .mp4 and .mov files. The technologies in QuickTime 6.3 that provide these capabilities are newly enhanced MPEG-4 and H.263 video codecs, a newly enhanced AAC (Advanced Audio Coding) audio codec, a new AMR (Adaptive Multi-Rate) audio codec, a new 3G Text importer and exporter, and a new user-friendly 3GPP export dialog to aid in the creation of 3GPP-compliant files. This support is available via the QuickTime 3GPP Component. Learn more about 3GPP.

Additional Technologies

QuickTime 6.3 also delivers playback of .amr and .sdv files, automatic detection of streaming transport, significant enhancements to DV audio and video synchronization, and improved support for Keynote, iMovie, and iDVD.""

o How stable & reliable is the platform?

QuickTime is a well established and widely spread technology, and can be considered quite stable and reliable. Most shortcomings in reliability (e.g. concerning scripting the plug-in) are due to the multitide of browser flavors and operating systems (the same holds for any other cross-platform technology). In general, QuickTime is probably the cross-platform multimedia technology with the highest congruence between Mac and PC versions. Its playback behavior on Mac and PC is less divergent than Real or Windows Media Player.

o What issues might stand in the way of adoption by consumers?

- On Windows PCs QuickTime is not as widely distributed as Windows Media Player, as it's not preinstalled with any Windows OS (nevertheless a full 90 percent of QuickTime users are on Windows-based systems, and QuickTime 5 alone will have something like 100 million Web downloads in its first year).
- If QuickTime is not yet installed, quite a big download is required (4,4 MB - 9,4MB, depending on components).

o What issues might stand in the way of adoption by developers?

- Until recently, many track types could only be created/edited by small specialized freeware/shareware applications (most of which are Mac only).
- To use the interactive features of wired movies, users have to purchase LiveStage, and a new scripting language (QScript) has to be learned.
- At the moment the plug-in situation (individual versions for different browsers offering different amounts of JavaScript control) makes it hardly possible to use JavaScript control on the web (better situation on an intranet where all users have the same browser installed)
- In combination with Macromedia Director: many QuickTime features are not supported yet, Director's QuickTime support exposes only a tiny fragment of what is actually available.
- Apple's support and documentation is not always satisfying.
- Most sound editing software on Windows PCs (Sound Forge, WaveLab etc.) does not offer QuickTime support, so different programs are necessary for editing and encoding.
- Still no Software Sequencer and/or Instrument Editor fully supporting QuickTime Music available.

Emerging Technologies

Author: Martin Wilde

In this document, I've tried to present some specific information on the large number of newer audio-related technologies available for the Web. Each has their purpose and target market, which is simultaneously a good and bad thing. Good because I truly believe the killer, all-encompassing, final solution for everything audio on the Web doesn't exist - there's just too much diversity in what people want to do for monolithic solutions to succeed. Bad because you have to learn what each thing can do and if it plugs into, extends, replaces or refines the audio experience you want to deliver.

I also wanted to give you a short list of those things I think are some of the most powerful and potentially useful technologies out there. No doubt you all will have your own opinions, which, not surprisingly, may differ from my own ;). But at the risk of getting my head chopped off, for one reason or another, and so you actually see this list (because getting to the end of the paper will be a *huge* accomplishment in and of itself), they are:

-**MPEG-4** for its ability to format, organize, and deliver a myriad of multimedia experiences within the same shell. While intimidating, it's also modular - you can pick and choose which parts of the spec you want to support or work in, and deliver just that piece. You don't have to bring along everything MPEG-4 is with every file, just the part you want. To me, this offers a tremendous flexibility and power, and will contribute to its adoption and use.
- **SMIL and it's Microsoft HTML+TIME counterpart** - because of the coordination and timing capabilities in the delivery of multiple media files within a site. It's easy to learn and powerful to use, and it's already supported by some of the major players (Real and Apple, to name two).
- **VoiceXML**, which brings natural speech to the tasks of the web across a huge variety of platforms - from PCs to cell phones to PDAs and a myriad other dedicated information appliances.
- **MPEG-7** - while I don't include detailed description here, this standard has to deal with what is known as "metadata." This is not the actual audio data itself, but standardized descriptions of the underlying structures of what the sound content contains. Akin to the efforts of such groups as MuscleFish and Yamaha Music Technologies, this is a standard which focuses on the search, filtering and retrieval of audio content based on its descriptive features.

So, without further ado, and in no particular order, sharpen your pencils, have your Visine ready, and prepare yourself…

STREAMING MEDIA TECHNOLOGIES

The idea here is to feed media to the user as it is being viewed. Note that true streaming is done from a streaming server. These are available in several proprietary, commercial packages. But there are several technologies, including QuickTime, which have been released as open source.

"Psuedo-streaming" is when you load some amount of content before starting your presentation and the media are served from a standard Web server. This has the advantage of being more accessible and cheaper than a streaming server. The disadvantage is that your streaming will not be as robust, especially if you have a lot of files to stream to a lot of people.

The power of streaming media comes in streaming formats that can stitch together a wide variety of resources in a multimedia presentation. Here are several:

SMIL

The Synchronized Multimedia Integration Language (SMIL, pronounced "smile") is a recommendation from the World Wide Web Consortium (W3C) that allows for the creation of time-based multimedia delivery over the web. It allows developers to mix many types of media, text, video, graphics, audio and vector based animation together and to synchronize them to a timeline.

SMIL enables simple authoring of interactive audiovisual presentations. It is typically used for "rich media"/multimedia presentations which integrate streaming audio and video with images, text or any other media type. It provides for:

- the definition of timing relationships in it's display
- the spatial layout of objects
- direct inclusion of non-text and non-image media
- hyperlink support for time-based media, and
- adaptation to varying user and system characteristics.
-

SMIL is an XML-compliant format, similar in syntax to HTML but with a lot more rules. SMIL presentations can be written using a simple text-editor, and is currently supported in Flash5, RealPlayer and QuickTime.

W3C SMIL recommendation: http://www.w3.org/TR/smil20/
Streaming Media World SMIL home: http://smw.internet.com/smil/smilhome.html

HTML+TIME

Netscape and Microsoft have decided to withhold support of the SMIL standard. Netscape is taking a wait and see approach and is very busy with their Mozilla browser project. However, Microsoft, Macromedia and Compaq are currently developing another standard that more closely integrates synchronized multimedia with HTML. This standard is called HTML+TIME (for Timed Interactive Multimedia Extensions) and has been proposed to the W3C as a competing standard. The W3C has characterized this standard as "extending SMIL into the browser."

HTML+TIME is targeting the need for a standard means for HTML authors to easily add timing and interaction relationships to arbitrary HTML elements, and to coordinate these with time-based media. TIME describes a set of extensions to add additional timing, interaction and media delivery capabilities to HTML. These are modeled closely along the lines of SMIL, and attempt to reuse terminology.

The timing and interaction support augment current script support for timers and DHTML. Simple timing is supported with a very simple syntax, but more complex timing constructs can also be described.

Microsoft says HTML+TIME is not intended to supplant SMIL and contains several new tags to support specific features described in the SMIL spec. Some differences in the timing syntax are introduced, however. HTML+TIME marks up existing HTML with inline timing info.

HTML+TIME also includes support for asynchronous media loading and dynamic resynchronization of players into a simple concept of sync rules and scope. HTML+TIME defines additional attributes for media elements, and a general mechanism for managing dynamic synchronization of timeline elements.

And finally, it should be reiterated that Internet Explorer does not support SMIL but only HTML+TIME.

HTML+TIME introduction: http://msdn.microsoft.com/workshop/author/behaviors/time.asp
SMIL and HTML+TIME tutorials: http://smw.internet.com/smil/tutor/

MPEG-4

MPEG (the Moving Picture Experts Group) is a working group in a subcommittee of ISO/IEC (International Organization for Standardization / International Electrotechnical Commission) in charge of developing international standards for compression, decompression, processing, and coded representation of moving pictures, audio, and their combination.

In particular, MPEG defines the syntax of low bitrate video and audio bitstreams of synthetic and natural sources, descriptions of their structure and content, and the operation of conformant decoders of these bitstreams. The encoder algorithms are not defined by MPEG. This allows for continuous improvement of encoders and their adaptation to specific applications, within the bitstream syntax definition. Along with the video and audio coding, MPEG also defines means to multiplex several video, audio and information streams synchronously in one single bitstream, describes methods to test conformance of bitstreams and decoders to the standard, and publishes technical reports containing software describing the decoder operation and software describing examples of encoder operation.

There are several different phases of MPEG, denoted by Arabic numerals. The following describes what is provided audio-wise by the various phases:

MPEG-1 (ISO/IEC 11172-3) provides

- single-channel ('mono') and two-channel ('stereo' or 'dual mono') coding of digitized sound waves at 32, 44.1, and 48 kHz sampling rate. The predefined bitrates range from 32 to 448 kbit/s for Layer I, from 32 to 384 kbit/s for Layer II, and from 32 to 320 kbit/s for Layer III.

MPEG-2 BC (ISO/IEC 13818-3) provides

- a backwards compatible (BC) multichannel extension to MPEG-1; up to 5 main channels plus a 'low frequent enhancement' (LFE) channel can be coded; the bitrate range is extended up to about 1 Mbit/s;
- an extension of MPEG-1 towards lower sampling rates 16, 22.05, and 24 kHz for bitrates from 32 to 256 kbit/s (Layer I) and from 8 to 160 kbit/s (Layer II & Layer III).

MPEG-2 AAC (ISO/IEC 13818-7) provides

- a very high-quality audio coding standard for 1 to 48 channels at sampling rates of 8 to 96 kHz, with multichannel, multilingual, and multiprogram capabilities. AAC works at bitrates from 8 kbit/s for a monophonic speech signal up to in excess of 160 kbit/s/channel for very-high-quality coding that permits multiple encode/decode cycles. Three profiles of AAC provide varying levels of complexity and scalability.

MPEG-4 (ISO/IEC 14496-3) provides

- coding and composition of natural and synthetic audio objects,
- scalability of the bitrate of an audio bitstream,
- scalability of encoder or decoder complexity,
- Structured Audio: A universal language for score-driven sound synthesis,
- TTSI: An interface for text-to-speech conversion systems.

MPEG-7 (ISO/IEC 15938) will provide

- standardized descriptions and description schemes of audio structures and sound content,
- a language to specify such descriptions and description schemes,
- goes beyond the domain of audiovisual coding and focuses on efficient search and retrieval, filtering and semantic description of content (metadata).

Unlike most of the rest of the Internet, streaming media has been dominated by proprietary tools and proprietary solutions. Enter the MPEG-4 standard -- a set of specifications that may be used to build products for creation, encoding and delivery of audio/video content over many kinds of networks to a variety of clients (personal computers, wireless receivers, set-top boxes, gaming consoles, web browsers, handhelds, etc.).

MPEG-4 provides the standardized technological elements enabling the integration of the production, distribution and content access. For all parties involved, MPEG seeks to avoid a multitude of proprietary, non-interworking formats and players. MPEG-4 achieves these goals by providing standardized ways to:

1. represent units of aural, visual or audiovisual content, called "media objects". These media objects can be of natural or synthetic origin; this means they could be recorded with a camera or microphone, or generated with a computer;
2. describe the composition of these objects to create compound media objects that form audiovisual scenes;
3. multiplex and synchronize the data associated with media objects, so that they can be transported over network channels providing a QoS appropriate for the nature of the specific media objects; and
4. interact with the audiovisual scene generated at the receiver's end.

MPEG-4 audio coding integrates the worlds of speech and high quality audio coding as well as the worlds of sound synthesis and the representation of natural audio. The sound synthesis part is comprised of tools for the realization of symbolically defined music and speech. This includes MIDI and Text-to-Speech systems. Furthermore, tools for effects processing and 3-D localization of sound are included, allowing the creation of artificial sound environments using artificial and natural sources.

Synthetic audio is described by first defining a set of 'instrument' modules that can create and process audio signals under the control of a script or score file. An instrument is a small network of signal processing primitives that can emulate the effects of a natural acoustic instrument. A script or score is a time-sequenced set of commands that invokes various instruments at specific times to contribute their output to an overall music performance. Other instruments, serving the function of effects processors (reverberators, spatializers, mixers), can be similarly invoked to receive and process the outputs of the performing instruments. These actions can not only realize a music composition but can also organize any other kind of audio, such as speech, sound effects and general ambience. Likewise, the audio sources can themselves be natural sounds, perhaps emanating from an audio channel decoder, thus enabling synthetic and natural sources to be merged with complete timing accuracy.

TTS is becoming a rather common interface and plays an important role in various multi-media application areas. For instance, by using TTS functionality, multi-media contents with narration can be easily composed without recording natural speech sound. Moreover, TTS with FA/AP/MP functionality would possibly make the contents much richer. In MPEG-4 activity, common interfaces for TTS and TTS for FA/AP/MP are proposed. The proposed MPEG-4 TTS functionality; Hybrid/Multi-Level Scalable TTS, can be considered as a superset of the conventional TTS framework. This extended TTS can utilize prosodic information of natural speech in addition to input texts and can generate much higher quality of synthetic speech. The interface and its bitstream format is strongly scalable; for example, if some parameters of prosodic information are not available, it then generates the missing parameters by rule. Still the basic idea for the scalable MPEG-4 TTS interface is it can fully utilize all the provided information according to the level of user's requirements. The functionality of this extended TTS thus ranges from conventional TTS to natural speech coding and its application areas, from simple TTS to AP with TTS and MP dubbing with TTS.

MPEG-4 standardizes natural audio coding at bitrates ranging from 2 kbit/s up to 64 kbit/s. The presence of the MPEG-2 AAC standard within the MPEG-4 tool set will provide for compression of general audio. For the bitrates from 2 kbit/s up to 64 kbit/s, the MPEG-4 standard normalizes the bitstream syntax and decoding processes in terms of a set of tools. In order to achieve the highest audio quality within the full range of bitrates and at the same time provide the extra functionalities, three types of coder have been defined. The lowest bitrate range between about 2 and 6 kbit/s, mostly used for speech coding at 8 kHz sampling frequency, is covered by parametric coding techniques. Coding at the medium bitrates between about 6 and 24 kbit/s uses Code Excited Linear Predictive (CELP) coding techniques. In this region, two sampling rates, 8 and 16 kHz, are used to support a broader range of audio signals (other than speech). For the bitrates typically starting at about 16 kbit/s, time to frequency coding techniques are applied. The audio signals in this region typically have bandwidths starting at 8 kHz.

A number of functionalities are provided to facilitate a wide variety of applications that could range from intelligible speech to high quality multichannel audio. Examples of the functionalities are speed control, pitch change, error resilience and scalability in terms of bitrate, bandwidth, error robustness, complexity, etc. as defined below. These functionalities are applicable to the individual coding schemes (parametric, CELP and t/f) as well as across the coding schemes.

- The speed change functionality allows the change of the time scale without altering the pitch during the decoding process. This can, for example, be used to implement a "fast forward" function (data base search) or to adapt the length of an audio sequence to a given video sequence.
- The pitch change functionality allows the change of the pitch without altering the time scale during the encoding or decoding process. This can be used for example for voice alteration or Karaoke type applications.

Bitrate scalability allows a bitstream to be parsed into a bitstream of lower bitrate such that the combination can still be decoded into a meaningful signal. The bit stream parsing can occur either during transmission or in the decoder.

- Bandwidth scalability is a particular case of bitrate scalability, whereby part of a bitstream representing a part of the frequency spectrum can be discarded during transmission or decoding.
- Encoder complexity scalability allows encoders of different complexity to generate valid and meaningful bitstreams.
- Decoder complexity scalability allows a given bitstream to be decoded by decoders of different levels of complexity. The audio quality, in general, is related to the complexity of the encoder and decoder used.
- Error robustness provides the ability for a decoder to avoid or conceal audible distortion caused by transmission errors.

To allow for smooth transitions between the bitrates and to allow for bitrate and bandwidth scalability, a general framework has been defined. Starting with a coder operating at a low bitrate, by adding enhancements both the coding quality as well as the audio bandwidth can be improved. These enhancements are realized within a single coder or alternatively by combining different techniques. Additional functionalities are realized both within individual coders, and by means of additional tools around the coders. An example of a functionality within an individual coder is pitch change within the parametric coder.

MPEG-4 Structured Audio (MP4-SA) is an ISO/IEC standard that specifies sound not as sampled data, but as a computer program that generates audio when run. MP4-SA combines a powerful language for computing audio (SAOL, pronounced "sail") and a musical score language (SASL, pronounced "sassil") with legacy support for the MIDI format. MP4-SA also defines an efficient encoding of these elements into a binary file format (MP4-SA) suitable for transmission and storage.

MP4-SA is different from standards like the MIDI File Format, because it includes not only the notes to play, but the method for turning notes into sound. As a result, MP4-SA is normative -- an MP4-SA file will sound identical when converted by any compliant decoder.
If the instrument models use algorithmic synthesis instead of wavetables, an MP4-SA file can describe realistic musical performances without using any audio data -- just score data, mixdown cues, and DSP algorithms. In this case, the MP4-SA file is about the same size as a MIDI File, but is a lossless encoding of the audio heard at mixdown.

One implication of this design is that encoding content in MP4-SA is a creative act, not an automatic one. Or more specifically, two creative acts:

- Sound modeling. In MP4-SA, sound happens because a program written in the SAOL computer language outputs audio samples. The algorithms coded in SAOL may model how musical instruments (like a piano or the human voice) create sound, or may process sounds (tasks like adding reverberation or mixing instrument sounds together).
- Sound sequencing. Performance is sound moving in time: notes play in sequence to make a melody, faders sweep across a mixing console to blend a performance, etc. In MP4-SA, the score language SASL, the MIDI standard, and the SAOL language itself are all available to support sound sequencing.

Appropriate choice of MPEG-4 profiles will enable a number of applications that are either not possible today, or are possible only via proprietary technology. These can be deployed today starting from very simple profiles and can be made to grow compatibly using more sophisticated profiles when technology enables and market so demands. Some examples:

- Audio on demand on the Web. MPEG-4 Audio at 16 kbit/s gives a very interesting quality for commercial applications at bitrates available on the Web or on mobile networks today. Audio services at higher bitrates approaching transparent quality can be served on fast-emerging high-speed cable modem or DSL links;
- *Digital radio broadcasting* in narrow band channels, such as digitization of AM radio;
- *Video services on the Web*. The ability to compose even simple 2 D scenes provides interesting possibility for possibly protected services that can be consumed by different cultural environments, targeted advertising etc.;
- *Interactive multimedia on mobile*. The coming 2.5G or 3G mobile services will provide various types of functionalities (point-to-point and point-to-multipoint) at bitrates that may be as high as 2 Mbit/s. The MPEG-4 Video and Audio standards have embedded error resilience designed especially for the mobile channel;
- *Digital Multimedia Broadcasting*. Many types of digitized broadcasting delivery systems exist. MPEG-4 can be easily deployed on such systems to broadcast interactive multimedia services;
- *Electronic Program Guides (EPG)*. The set top box is considered to be a vital gateway for the merger of broadcasting services and other services. The 2D composition profiles offers a very powerful composition tool for EPG today with a well established route for more powerful features tomorrow;
- *Virtual Reality experiences on the Web*. The high-compression capability of MPEG-4 and the ability to download portions of the world only when they are actually needed makes MPEG-4 a very interesting tool to enjoy virtual worlds with streaming content at the typical bitrates of the Web today;

- *Virtual sites on the Web*. The synthetic face and body of MPEG-4 make it possible to create virtual spaces for new exciting experiences on the narrowband Web of today;
- *Interactive local multimedia*. Complex virtual worlds may be stored on a CD-ROM or DVD-ROM, while last-minute updates can be made from the web or a broadcast channel

It should be stressed that an MPEG-4 browser IS NOT required to support ALL the features listed above simultaneously. An MPEG-4 browser can be an inexpensive piece of hardware for the mobile user, a medium-complexity piece of software running on the set top box or a very complex piece of software with a graphic accelerator requiring a top-level performance PC.

There is an additional reason why MPEG-4 is ready to go and can be exploited by anybody. It is possible to download the reference software from the ISO web site. ISO gives MPEG-4 users free license to the software and its modifications for use in products claiming conformance to MPEG-4. This software has the highest ISO status - normative value - i.e. the same level of importance as the traditional text-based description of the standard.

The Wireless Multimedia Forum (WMF) and the 3rd Generation Partnership Program (3GPP) have selected several MPEG-4 technologies for use in the mobile environment. The same has done the Internet Streaming Media Alliance (ISMA) for streaming of audio-visual content over the Internet.

MPEG Official website: http://mpeg.telecomitalialab.com
MPEG Audio is a subgroup of MPEG working on all audio aspects of the MPEG standards.
Official Website: http://www.tnt.uni-hannover.de/project/mpeg/audio/
MPEG-4 Structured Audio homepage: http://sound.media.mit.edu/mpeg4/
MPEG-4 Industry Forum: http://www.m4if.org/

Koan/SSEYO

SSEYO develops both server and embedded Interactive Audio Software platforms with very small footprints. SSEYO's mission is to become the de facto standard for wireless interactive audio across narrow and broad band digital networks to convergent consumer devices. It has developed a portfolio of audio products for mobile. It has patent pending server side ringtone remixing solutions that generate personalized ringtones for existing mobile handsets, as well as comprehensive interactive audio platforms for the Internet and higher end mobile devices, such as wireless PDAs, where non-voice high quality interactive audio can be delivered in a very small space.

SSEYO Koan software has a number of constituent parts:

- The SSEYO Koan Interactive Audio Platform (SKIAP). The SKIAP provides full support for services normally required of an audio playback / rendering system. In addition, it provides many other features required for delivering enriched audio solutions over low bandwidth connections within cost effective processing and memory requirements. The default configuration of Koan is as a full-featured interactive audio engine (the SKIAP), but Koan components may be deployed in any number of “mix and match” configurations. Note: The SKIAP is also available in the form of a Plug-in for an Internet browser, with a comprehensive API that allows dynamic population of webpage content entirely through using Koan audio vectors (see below). The SKIAP is written using portable C++, to run on a large number of platforms, with a particular view to suitability (in terms of both portability and high performance) for running on embedded systems. Supported Operating Systems currently include Windows 95/96/NT4/2000, MacOS 7.3+, Intent 1.1, Window CE, Symbian, Linux, etc.

- The SSEYO Koan Synth Engine (SKSE). This is a fully featured, real-time configurable software synth capable both of sample-based and analog (oscillator)-based synthesis through arbitrary networks of effects such as reverberation, filtering, and modulation. The SKSE is a hybrid digital/analog synth with no inherent limit as to the number of voices. Each voice can be sourced separately from a combination of sample data (in formats such as MP3, WAV or DLS) and/or built-in oscillators. The SKSE supports a number of independently specified effects units per voice as well as global (piece-level) effects. Individual effect units in a Voice or piece-level effect chain (referred to as an "Effect Module") can be enabled, disabled, added and deleted in real time through the Koan APIs while the audio is playing. The SKSE allows multimedia developers to break beyond the limitations of MIDI. The SKSE is written using portable C++ and runs on all platforms to which the SKIAP is ported. The SKSE is capable of generating its audio at any audio sample rate supported by the local hardware / operating system (e.g. from 2KHz through to 48KHz and beyond), with either 8-bit or 16-bit sample data widths, and can be run in either mono or stereo : all of these features are software configurable.

- The SSEYO Koan Ringtone Engine. This is used for composing both monophonic and polyphonic ringtones. The technology can be deployed either on a client or on a server, and is implemented in a very compact Java engine. This technology allows SSEYO to tap-in to developing trends in personalizing audio for personal communication devices, whether or not a local Koan engine is embedded.

- Koan Vector Audio (KVA). A portable, scalable, easy to use interactive audio platform that optimizes performance within very small memory, processing and bandwidth requirements. A Koan audio vector is plain text that is easily added to a webpage, Flash movie or email and starts playing immediately. Koan audio vectors can be as small as 10 bytes, and contain note information like MIDI, but also information on how to create the sounds, which drives Koan's powerful built-in software synthesizer. No audio samples or audio sample set is required… vectors can include links to audio samples located remotely or on the operating system or platform. KVA is available for servers, games consoles, cell phones, PDAs, desktop, and set top applications.

- The SSEYO Koan Music Engine (SKME). This is a fully real-time programmable rule-based musical event generator. It creates fully polyphonic harmonized audio output from a parametric description of music and sequence structures. It can be considered an “expansion” technology in that it can generate on-the-fly a wealth of music and audio content from settings of the relevant Koan parameters and rules. These parameters are supported within the Koan audio vector format.

SSEYO's Koan technology has already been ported to (and is commercially available now for) a range of operating systems including Windows 95/98/2000/ME/NT4, Mac OS 8/9, Windows CE (ARM, MIPS etc. for Pocket PC and other platforms), Tao's intent™ (for a wide range of operating systems including Linux, EPOC, WindRiver, QNX etc. to run on hardware including ARM, MIPS, Hitachi, Intel, Motorola and other devices), and a native port to ARM. Future embedded system Koan ports expected to be completed in the near term include embedded Linux, EPOC, and QNX. Future desktop ports in the medium term include desktop Linux and MacOS X. Koan Server technology is available now in the Java language for any Java implementation for client - or server-side deployment, across a wide range of operating systems and hardware configurations.

Website: http://www.sseyo.com/

ISMA

The Internet Streaming Media Alliance (ISMA) is a standards body comprising such tech heavyweights as Apple Computer and Cisco Systems that has released a specification for streaming MPEG-4 video and audio via the Web.

ISMA recently announced that it has developed and published its first specification. ISMA 1.0 will let consumers install one plug-in for streaming audio and video, rather than a raft of programs each specific to a single format, on devices ranging from cell phones to personal computers. The specification is the latest effort to create open standards in streaming media. Microsoft's Windows Media, RealNetworks' RealPlayer and Apple's QuickTime each hold a piece of the market. Although Apple joined Cisco, IBM, Kasenna, Philips Electronics, Sun Microsystems and other tech companies in founding the nonprofit group last year, market leaders RealNetworks and Microsoft have yet to join.

ISMA 1.0 has two versions. Profile 0 helps wireless and narrowband networks stream audio and video content to devices, such as cell phones or PDAs (personal digital assistants), for limited viewing and listening. Profile 1 is devised for broadband networks and targeted to more powerful devices such as set-top boxes and personal computers.

Apple is hoping the standard will help its QuickTime, which trails Windows Media and RealPlayer, gain popularity. The computer maker is expected to release a new version of QuickTime based on the MPEG-4 format. ISMA 1.0 could free content creators and distributors from depending on a single vendor for streaming media technology. Without an overarching standard, companies must choose one format or encode the same audio and video for several players. With ISMA 1.0, companies would only need to encode the content once to stream it over all compliant players. Microsoft, however, says it is unimpressed with the quality and application of MPEG-4; rather than join ISMA, the software giant has chosen to focus on an upgrade of its own technology, Windows Media 8.

From their web site: "Just as the adoption of standard mark-up languages has fueled innovation and the explosive use of today's World Wide Web, the goal of Internet Streaming Media Alliance is to accomplish the same for the next wave of rich Internet content, streaming video and audio. The Alliance believes that in creating an interoperable approach for transporting and viewing streaming media, content creators, product developers and service providers will have easier access to the expanding commercial and consumer markets for streaming media services. To date, the prohibitive costs associated with rolling out streaming video services that support all current, disparate formats has kept many potential service providers and other adopters from taking full advantage of existing market opportunities. The emerging class of Internet appliances stands to benefit from a single standard as these devices often cannot afford to have multiple streaming media players installed to view differently formatted video content from the web.

Standards for many of the fundamental pieces needed for a streaming rich media over IP solution do exist. The ISMA adopts parts or all of those existing standards and contributes to those still in development in order to complete, publish and promote a systemic, end-to-end specifications that enables cross-platform and multi-vendor interoperability. The first specification from the ISMA defines an implementation agreement for streaming MPEG-4 video and audio over IP networks. The Alliance´s ongoing work to augment the specifications will include adopting methods for digital rights management, reliable quality of service as well as other relevant technologies."

Website: http://www.isma.tv/

JAVA TECHNOLOGIES

jMusic

jMusic is an open source programming library written for musicians in the Java programming language. While still in its infancy this project hopes to develop a library that is simple enough for newbie programmers but sophisticated enough to enable composers to accomplish real work, whatever form that may take. jMusic is designed to be used as a compositional medium, therefore it is primarily designed for musicians - not computer programmers.

As a library of classes for generating and manipulating music, jMusic provides a solid framework for computer assisted composition in Java. jMusic supports composers by providing a music data structure based upon note/sound events, and methods for analyzing and working with that musical data. jMusic can read and write MIDI files, audio files, and its own .jm files. jMusic is designed to be extendible, encouraging you to build upon its functionality by programming in Java to create your own musical tools and instruments. jMusic is 100% Java and works on Windows, Mac OS, Linux, Solaris, or any other platform with Java support.

jMusic is a tool for instrument building as well as music making. Java applications can be written using jMusic components. These components include a musical data structure with associated modification and translation classes as well as some graphical user interface elements. With jMusic you can create your own composing tools and environment. Interfacing jMusic with other music software is facilitated by easy importing and exporting of MIDI files and audio files.

Website: http://www.academy.qut.edu.au/music/newmedia/jmusic/

JSyn

JSyn is a Java API for synthesizing audio. This means that Java programmers can use JSyn to add sound to their programs. The API is the set of classes and their methods that are used directly by the programmer.

JSyn allows you to develop interactive computer music programs in Java. You can run them as stand-alone applications, or as Applets in a web page using the JSyn plug-in. JSyn uses native methods written in 'C' to provide real-time audio synthesis for Java programmers.
The JSyn Software Developer Kit (SDK) enables programmers to create Java applications and Applets that uses the JSyn API. It includes compiled classes, example programs and programmer guides.

JSyn is based on the traditional model of unit generators that can be connected together to form complex sounds. There is also a graphical patch editor and sound designer for JSyn called “Wire.” It allows you to connect unit generators together graphically, and hear the results in real-time.

JSyn consists of many layers. In addition to the JSyn API, there are additional elements to interface the high-level Java code with the underlying ‘C’ code, manage the audio synthesis units, an engine to do the actual audio synthesis, and a host audio driver at the output.

Website: http://www.softsynth.com/jsyn/

JavaSonics

JavaSonics provides high fidelity audio I/O services for multiple platforms. If Java Sound is available, then it is used to implement the JavaSonics API. If Java Sound is not present, then a custom native library is used.

You can use JavaSonics to develop Java applic ations with high fidelity audio, and deploy them on a wide variety of platforms including Java 1.1 and 1.3 Virtual Machines.

Feature Set:

1. Streaming output of 16 bit linear PCM audio data.
2. Streaming input from a microphone or LineIn.
3. Support for mono or stereo streams.
4. Mixing of multiple audio output streams with independent control of gain and pan.
5. Uses Java Sound if present. Uses native plug-in if Java Sound is not available.
6. Load AIFF or WAV samples from an InputStream.

The JavaSonics API is similar to, but different from, the samples sound portion of the Java Sound API. The JavaSonics API was designed so that it could be implemented using either the Java Sound API or some native code that is specific to JavaSonics. The goal here is for people to be able to write applications using the JavaSonics API and then be able to run them on platforms that either do or do not support Java Sound. So the JavaSonics API had to be limited to only doing things that can be implemented using Java Sound.

The Java Sound API is not used because applications should be able to choose between the native implementation, or the Java Sound implementation of the JavaSonics API. There may be instances where Java Sound does not support a particular device or audio feature. But the native JavaSonics engine could support these pieces so the application may prefer to use native JavaSonics. In general, the APIs are similar enough in concept that an application can be quickly ported from one to the other if needed.

JSyn and JavaSonics are efforts from SoftSynth.com.

Central website: http://www.softsynth.com/
JSyn site: http://www.softsynth.com/jsyn/
JavaSonics site: http://www.javasonics.com/

XEMO

Project XEMO is an open source, modular software environment for the development and delivery of interactive music, audio and sound applications, and it is written in Java.

XEMO stands for eXtensible Ele ctronic Music Object architecture. Its goal is to explore the vision of an integrated composition environment for the analysis, notation and composition of music, and to develop basic building blocks and services that would enable innovative music software applications to be written. All parts of a XEMO application are designed as independent modules which are loaded into the XEMO Integrated Composition Environment (ICE). Modules can be mixed and matched to build an ICE.

Project XEMO consists conceptually of three layers: a platform layer, a middle "services" layer, and an application layer. The platform layer consists of the NetBeans platform, and includes remote update support for all parts of the application, workspace & window management, definition and management of projects and utilities for search, and an integrated cross-platform help system. This layer also provides the internal architecture needed for the development of modules, and defines the formats for remote publishing and update of modules. The middle "services" layer provides building blocks to enable musical application building. It's core elements are APIs for musical notation, representation of musical structure and MIDI based playback & performance. And finally, the application layer consists of the implementation of integrated applications, such as notation editors, composition tools, performance and playback sessions, and other creative applications yet to be realized.

NetBeans is an open source integrated development environment for the Java language. Currently Project XEMO uses the default Java Sound implementation supplied by JDK 1.3[tm] for all sound output. XEMO runs on Windows, Linux, and Mac OS X.

XEMO website: http://www.xemo.org/

X-Smiles

X-Smiles is a Java based XML browser. It is intended for both desktop use and embedded network devices and to support multimedia services. The main advantage of the X-Smiles browser is that it supports several XML related specifications and is still suitable for embedded devices supporting the Java environment. X-Smiles is not an HTML browser**,** though in the future it may support XHTML.

X-Smiles is a non-profit project started by the Telecommunications Software and Multimedia Laboratory at Helsinki University of Technology. It was first conceived in a student software project in 1998-1999. Since it has been developed by the staff of the laboratory in the GO project (http://go.cs.hut.fi/). X-Smiles was released as open source in the beginning of the year 2001.

The main objective of the X-Smiles project is to deliver a pure Java XML browser, capable of displaying documents written in various XML languages. The X-Smiles browser should support at least:

- XSL Transformations (http://www.w3.org/TR/xslt.html/)
- XSL Formatting Objects (http://www.w3.org/TR/xsl/)
- Synchronized Multimedia Integration Language (http://www.w3.org/TR/REC-smil/)
- Scalable Vector Graphics (http://www.w3.org/TR/2000/CR-SVG-20001102/index.html/)

Another main goal is that X-Smiles should be suitable for small embedded devices supporting Java. This goal includes porting X-Smiles to Kaffe VM (http://www.kaffe.org/ - an open-source implementation of a Java virtual machine and class libraries). Currently Kaffe supports Java 1.1, while some X-Smiles features are based on Java 1.2.

An important secondary goal of X-Smiles is to redesign the browser core to allow mixing of different (supported) XML languages in a single document. Another secondary goal is to allow rich multimedia content, such as video and audio, and streaming of it.

One secondary goal is to have some kind of form support in X-Smiles browser to allow interactive services. One possibility is the implementation of Xforms (http://www.w3.org/MarkUp/Forms/), which is a future form description language currently under development.

Helsinki University of Technology: http://www.hut.fi/English/
X-Smiles site: http://www.x-smiles.org/index.htmlWildTangent

Wild Tangent

WildTangent is a company trying to enable online entertainment over a narrow bandwidth connection. It is primarily trying to do this via it's "Web Driver" technology. The Web Driver provides a platform for the development of high quality, high performance, compact 2D and 3D content for the Internet. It consists of a high level API for Java, JavaScript, and other COM enabled languages such as C, C++ and Visual Basic, and has a powerful graphical engine underneath. The API provides both 2D and 3D support, allowing the creation of everything from simple 2D or 3D content such as product visualizations for e-commerce, through to full-fledged 3D games. All of these can run within a web page, simply by visiting a web site.

The Web Driver consists of a downloadable component of under 1MB, and includes an updater mechanism that provides a way to distribute new versions with additional functionality and maintenance fixes. The plug-in includes Java and JavaScript interface layers, for use by Java/JavaScript enabled Internet Explorer and Netscape browsers. While the Web Driver was designed for the Internet, it also has a COM interface that can be used in standalone applications as well, using languages such as Visual Basic, C, and C++ .

Audio support within the WebDriver is limited to a set of basic calls to load, get and set information, start and stop playback and change the frequency of a sound. All .mid and .wav (and .wav ADPCM) files are compressed into WebDriver files (.wwv) produced by their own proprietary utility, "WildCompress."

WildTangent also has a toolset called "Multiplayer." Like the Web Driver, this toolset is an abstraction layer which provides access to underlying services, in this case multiplayer networking functionality. The multiplayer API gives access to multiplayer functionality that is provided by a range of possible multiplayer modules. By supporting an extensible modular approach, the system can make use of a variety of network transports, communications systems, and lobby systems, and implement additional multiplayer functionality in the future.

In their latest release, the API is available in Java within the MS IE browser. They claim future releases will expand this language support to include C/C++, JavaScript, VB, and VBScript, and additional languages, which can interoperate with a COM interface.

Website: http://www.wildtangent.com/

MUSIC COMPOSITION AND SYNTHESIS

Common Music

Common Music (CM) is an object-oriented music composition environment from the Stanford University Center for Computer Research in Music and Acoustics (CCRMA). It produces sound by transforming a high-level representation of musical structure into a variety of control protocols for sound synthesis and display. Common Music defines an extensive library of compositional tools and an API through which the composer can easily modify and extend the system.

Common Music is implemented in Common Lisp and CLOS and runs on Macintosh, SGI, and PC.

Website: http://www-ccrma.stanford.edu/software/cm/cm.html

SequenceXtra

SourceForce is a Swedish development outfit says they focus “on the combination of music and computers in every possible way.” They contract to do Macromind Director multimedia applications and object oriented Lingo programming. They also offer developing services in HTML, JavaScript and C++, and have experience in developing Xtras for Macromedia Director. Also offer educational training services. sequenceXtra is their first software product. It is a MIDI Xtra for Macromedia Director which makes it possible to do some basic, musical interaction.

sequenceXtra features:

- Realtime recording from any MIDI keyboard
- Music input with computer keyboard or mouse
- Non destructive quantization
- Real time editing of tracks, parts and notes

Website: http://www.sourceforce.nu/home.htm

Staccato Systems (Analog Devices)

Based on extensive research derived from work done under Stanford University's Sondius Program, Staccato Systems has developed an "Audio Rendering Technology" that allows one to create, deliver and control high-quality sounds with a lot of interactivity. Using a process that describes the properties of a sound rather than storing the sound itself, physically modeled sounds are extremely controllable and have the added advantage of taking up far less storage space than static wave files. This affords a high degree of real-time control over how the sounds react. In addition, the cross-platform Audio Rendering Technology allows for quick and easy modification of content giving the designer the ability to efficiently "tweak" and re-purpose audio without re-programming.

The Staccato Audio Rendering Engine, SynthCore, is a flexible, multi-platform, host-based synthesis engine that combines and allows physical modeling, process modeling, wavetable software synthesis, effects, and other synthesis techniques.

Staccato Systems offers products for two distinct groups of developers:

- The SynthCore Software Developer Kit is aimed at sound designers and game programmers, and
- SynthCore OEM, targeted at manufacturers of CODECs, sound cards, internet appliances and other hardware that would benefit from the cost effectiveness, extensibility, and improved sound quality Staccato provides.

The SynthCore Software Development Kit is a set of APIs, authoring utilities, and synthesis algorithms that allows the creation and control of high-quality interactive sounds for PC and console games. SynthCore's rendered sounds are stored as Downloadable Algorithms (DLAs). DLAs are files that contain only the compact instructions that SynthCore needs to render the sounds on demand. This small file size allows for extremely efficient file management. The SynthCore SDK allows you to create sounds that are totally tuned to a game or level. A sound's sliders get parameterized and saved as simple text-based presets that can be handed off to the game programmer.

The SynthCore Audio Rendering Engine is an OEM product that provides a software-only audio synthesis solution to PC manufacturers. SynthCore 1.2 for the PC is a scalable all-software MIDI synthesizer supporting the General MIDI (GM) and Downloadable Sounds (DLS) standards. It is a flexible, low-cost, sound card independent solution that provides high quality wavetable playback through the PC’s sound output via Microsoft’s DirectSound interface. This product can also be used with a sound card to supplement the system’s specifications by adding additional wavetable voices, DLS support, and algorithmic synthesis. This component can be assigned as the default MIDI device, so it is the synthesizer used by Windows when a MIDI file (file type .MID) is played with the Windows Media Player, a web browser, or other MIDI sequencer application. SynthCore 1.2 also provides high quality software reverb and chorus effects. The XG lite™ wavetable bank (GM compatible) is included with SynthCore 1.2. Additional sound sets may be loaded in the DLS format.

Staccato is a subsidiary of Analog Devices, Inc.
Website: http://www. audioforgames.com/

PortAudio

PortAudio is a cross platform, open source audio I/O library. It lets you write simple audio programs in 'C' that will compile and run on Windows, Macintosh, Unix(OSS), SGI, and BeOS. PortAudio is intended to promote the exchange of audio synthesis software between developers on different platforms, and was recently selected as the audio component of a larger PortMusic project that includes MIDI and sound file support.

Website: http://www.portaudio.com/

PortMusic

PortMusic is a set of APIs and library implementations for music. It is open source and runs on Windows, Macintosh and Linux. Currently,it has libraries that support Audio I/O and MIDI I/O.

Website: http://www-2.cs.cmu.edu/~music/portmusic/

ACCESS AND NETWORKING

Pulse Sonifier

Pulse is a 3D animation and rich media presentation solution for the Web. In June of 2001, they acquired a company called Sonicopia to integrate their animation technology with interactive audio. The technology is now called "Pulse Sonifier," and it claims to be "the first technology designed to enable easy 'sonification' of Web sites of all sizes." (I imagine Beatnik would have a bone to pick with that statement…).

To audio-enable a site using Pulse Sonifier, a user browses the comprehensive libraries of bandwidth-optimized vocal cues, musical scores, audio logos and navigational sound effects packaged with the program. Applying any of those sounds to a site in real-time, the user can preview the resulting sonification of each site element as they proceed. Sounds are organized by type, theme, length, and more. You can also mix and match individual sound elements from different packages to create custom sets suited to your individual taste. No technical expertise in the areas of HTML, JavaScript or client-server relationships is required, and it does not require any knowledge of audio composition, optimization and formatting.

The target of this technology are users and organizations creating web site, but also ISPs and Web-hosting companies where Pulse is being pitched as a value-added service for their customers, in the form of a co-branded, turnkey sonification solution.

Pulse also offers integrated solutions with dynamic data using Rhetorical rVoice text-to-speech technology. This technology integrates into industry-standard interfaces such as VoiceXML, SAPI, JSAPI, Nuance NuTTS and the Pulse rich media data components.

These solutions can be integrated into the enterprise on a wide range of server systems including Linux, Microsoft Windows and can be deployed to a wide range of Pulse-supported Microsoft Windows, Apple Macintosh and emerging wireless web mobile platforms.

Website: http://www.pulse3d.com/index.asp

Microsoft DirectPlay8

Microsoft DirectPlay is a media-independent networking API that provides networking services at the transport protocol and session protocol levels. DirectPlay's media independence means that DirectPlay sessions can be run on TCP/IP networks, IPX networks, and over directly connected modems and serial cables. DirectPlay's media independence also allows it to be extended in the future to support new standards and protocols. This means that DirectPlay applications can improve over time, and track evolving network standards, without additional development by the application developer.

The Microsoft DirectPlay API provides developers with the tools to develop multiplayer applications such as games or chat clients. A multiplayer application has two basic characteristics.

- Two or more individual users, each with a game client on their computer.
- Network links that enable the users' computers to communicate with each other, perhaps through a centralized server.

DirectPlay provides a layer that largely isolates your application from the underlying network. For most purposes, your application can simply use the DirectPlay API, and enable DirectPlay to handle the details of network communication. DirectPlay provides many features that simplify the process of implementing many aspects of a multiplayer application, including:

- Creating and managing both peer-to-peer and client/server sessions
- Managing users and groups within a session
- Managing messaging between the members of a session over different network links and varying network conditions
- Enabling applications to interact with lobbies
- Enabling users to communicate with each other by voice

The current trend toward team-based multiplayer games makes player-to-player communication an essential part of game play. Historically this has been confined to text-based communication, where players type out the messages to their teammates. Although suitable for slower, turn-based games, text-based communication is at best an inconvenience for real-time games. Not only does it put slow typists at a disadvantage during game play but also it is a significant break in the reality that games attempt to create for the player. An obvious solution to the problem is the use of speech as a means for communication. It requires no training and increases the immersion of the game itself.

The Microsoft Windows platform provides all the tools required to provide real-time voice conferencing to video game developers, but it requires a significant amount of effort on the part of the game developer. This, combined with the cost and difficulty of obtaining the rights to compression technology capable of handling extremely low bandwidth situations, has prevented the widespread use of voice in games. Microsoft DirectPlay 8.1 provides the game developer with a robust real-time voice conferencing system that requires minimal effort to use.

The DirectPlay Voice API provides developers with the tools to add real time voice communication to their applications with a minimum of development work. DirectPlay Voice includes extremely low bandwidth codec technology that can be used royalty free in your applications, as well as many features that allow you very fine grained control over the voice communication experience. DirectPlay Voice provides the following services:

- A selection of voice codecs from 64 kbit/sec down to 1.2 kbit/sec.
- Peer to peer, forwarding server, and mixing server topologies
- Completely flexible control of who hears whom within a session
- Automatic voice detection to control network transmission
- Automatic gain control to automatically set the microphone input volume
- Intelligent, configurable, adaptive queuing to automatically adjust to different and changing network conditions
- Integration with DirectSound, allowing for the application playback effects on a voice by voice basis such as 3-D positioning, reverb, etc.
- Capture focus to allow multiple DirectPlay Voice applications at once in the system
- Integration with DirectPlay for session and protocol level services

Website: *http://msdn.microsoft.com/library/en-us/dx8_c/directx_cpp/Play/DPlay.asp/*

VoiceXML

VoiceXML (VXML) is a markup language that allows a user to interact with the Internet through voice-recognition technology by using a voice browser and/or the telephone. Using VXML, the user interacts with a voice browser by listening to audio output that is either pre-recorded or computer-synthesized and submitting audio input through the user's natural speaking voice or through a keypad, such as a telephone.

VoiceXML is an XML format markup language that utilizes existing telephony technology to interact with users over the telephone through speech recognition, speech synthesis, and standard Web technologies. It is designed for creating audio dialogs that feature synthesized speech, digitized audio, recognition of spoken and DTMF key input, recording of spoken input, telephony, and mixed-initiative conversations.

VoiceXML's main goal is to bring the full power of web development and content delivery to voice response applications, and to free the authors of such applications from low-level programming and resource management. It enables integration of voice services with data services using the familiar client-server paradigm. A voice service is viewed as a sequence of interaction dialogs between a user and an implementation platform. The dialogs are provided by document servers, which may be external to the implementation platform. Document servers maintain overall service logic, perform database and legacy system operations, and produce dialogs. A VoiceXML document specifies each interaction dialog to be conducted by a VoiceXML interpreter. User input affects dialog interpretation and is collected into requests submitted to a document server. The document server replies with another VoiceXML document to continue the user's session with other dialogs. VoiceXML gateways also retrieve VoiceXML files over the HTTP protocol from any standard Web server.

Support for VoiceXML is nonexistent in most Web development tools that you might be using now like Dreamweaver and BBedit. However, you can use an XML tool like XMLSpy to develop and validate VoiceXML documents. There are also several VoiceXML editors available from independent providers like Voice Studio from Cambridge VoiceTech and V-Builder from Nuance that are shaping up fast. Support from big vendors is on the horizon however. IBM is one of the few vendors that has integrated

VoiceXML into its code editor for Web Sphere. This isn't surprising though since IBM is one of the leading VoiceXML platform providers.

VoiceXML forum: http://www.voicexml.org/
VoiceXML Planet: http://www.voicexmlplanet.com/

PROS AND CONS OF SELECTED EMERGING TECHNOLOGIES

SMIL

Pros

- Provides for the definition of timing relationships, spatial layout of objects, direct inclusion of non-text and non-image media, hyperlink support for time-based media, and adaptation to varying user and system characteristics.
- Currently supported in Flash5, RealPlayer and QuickTime
- Easy to edit: similar to familiar HTML, uses straight text

Cons

- Not supported by Netscape or MS.
- No native MIDI player with it, sampled clips only. (Question - why can't I use QT to play MIDI files?)
- No basic control of audio clips, just play (no vol, pan, pause).

MPEG-4

Pros

- A set of specifications that may be used to build products for creation, encoding and delivery of audio/video content over many kinds of networks to a variety of clients.
- Facilitates a wide variety of applications which could range from intelligible speech to high quality multichannel audio, and from natural sounds to synthesized sounds.
- Free: users have free license to the reference software and its modifications from the ISO web site.

Cons

- No widely available content creation tools.
- Confusing and intimidating - sheer size and plethora of profiles and options makes it hard to get a handle on what it does.
- More a packaging mechanism than a multimedia player.

JavaSonics

Pros

- API that provides high fidelity audio I/O services for multiple platforms.
- Can mix multiple audio output streams with independent control of gain and pan.
- Runs on platforms that either do, or do not, support Java Sound.

Cons

- MIDI not supported.

- More a teaching and research tool focus than a commercial endeavor.
- Java not supported in all browsers; widespread deployment problem.

JSyn

Pros

- API for synthesizing audio, based on the traditional model of unit generators that can be connected together to form complex sounds; real-time audio synthesis for Java programmers.
- Contains a graphical patch editor and sound designer that allows you to graphically connect unit generators together and hear the results in real-time.
- Provides time-stamping to allow scheduling of control events for rock solid timing.

Cons

- Java not supported in all browsers.
- No MIDI support; synthesis not always appropriate for web audio.
- Web content authors are not programmers, and SDK has to be licensed for commercial purposes.

Post Mortem: Beatnik (and the Beatnik Xtra)

Author: Michael Sweet

Note: *After this review was completed in early 2002, Beatnik (the company) announced a refocus of their business away from the web and on to mobile communications, placing Beatnik's (the product) future as a web technology in jeopardy. Still, the authors felt that even though Beatnik for the web is not widely used, and probably would not be used in the future, it is an important stepping stone for other possible newer web technologies, and should be included in this document, along with this disclaimer:*

"Developers considering working with the Beatnik web audio system should be aware that the company no longer provides any support for the product, and that the current browsers are not supported, and will probably never be supported, and that the code is not being maintained. While there is a rich musician and developer community, the product itself should be regarded as discontinued, and you basically use the system at your own risk."

OVERVIEW

Beatnik has many advantages for Internet audio authoring. Using standard MIDI files, it has a small footprint, a general MIDI sample set built in, and the ability to embed custom samples. Compression of audio is primarily via MP3 or ADPCM. MIDI and audio data are stored in Beatnik's RMF file format, and played back by the Beatnik Audio Engine contained in the Beatnik plug-in (it may also be included in standalone self-playing files). Beatnik supports 16 instruments playing at the same time in each of up to 16 instances of the player plug-in.

Interactivity with Beatnik is quite rich. Volume and panning control, muting, soloing, meta-event callbacks and complex looping are some of the possibilities. Unfortunately there is no way to break a loop other than to stop the sequence. For instance, if you are looping bars 3 to 4, it would be nice to have a command that let the sequence break the loop and move on to bar 5 when you completed a certain event in the game, or Flash movie. The interface also allows the ability to set the position of playback.

It is possible for two or more RMF files that start at the same time to stay in sync with one another, but this is sometimes a difficult task because Lingo or JavaScript (the primary control interfaces) can be one of the last things a browser polls for data.

Response to meta-events, which are triggered by markers in the RMF/MIDI file, is not accurate enough to branch from one sequence to another or set a new position and have it feel seamless. For instance, on reaching the end of level one of a game you may want to play an ending at the end of the current bar. Beatnik will likely play a skip if you try to set the position to the end using a meta-event.

However, it is possible to employ what is sometimes called "vertical remixing" to achieve a branching effect in Beatnik. Different sections or branches are represented by additional sets of tracks, muted at start, which unmute at the appropriate time to begin a new branch. Meta-events can be used to check whether it's time to unmute, or user interaction can set up the new mute positions. It's also possible to use Beatnik's advanced volume controls to perform a cross-fade from one sequence to another.

Communication between Flash and Beatnik can be quite frustrating, and JavaScript incompatibilities between the browsers make it hard to use this feature at all. Meta-event callbacks to Flash from Beatnik through JavaScript can be especially difficult.

Clients are often reluctant to require another plug-in unless absolutely necessary. This can make it more difficult to sell Beatnik unless there is a need for a feature that only Beatnik can provide. Considering Beatnik’s rich API this is sometimes the case.

During the development of this document Beatnik has dropped support of the Web based player. Although it is still possible to create content for Beatnik enabled web pages it's no longer possible to purchase any web development tools or get support directly from Beatnik.

If you are interested in using Beatnik, many web and audio developers may still have copies of the Beatnik Web development kits, and you may approach them if you want to utilize this technology. In addition Java integrated some of the technologies that Beatnik developed into their Java Sound packages and more information can be found out about this and Beatnik's RMF music file system from the Java Sound SDK.

BEATNIK (AND THE BEATNIK XTRA)

Authors: Dafna Naphtali and Luis Bergmann

o Pros:

- The use with Director, as a Beatnik Xtra, provides a stable platform without the plug-in´s browser incompatibility.
- The platform offers strong interactivity features that aren´t found in other web audio solutions, like the MIDI and RMF playing capabilities and their manipulation.
- The recent move of the player into open source may bring new life to this technology.

o Cons:

- Lack of compatibility with the latest browsers.
- Since there isn´t any open source player available at this moment, the platform has reduced commercial usefulness in projects that aren´t Director based.
- The recent lack of support by the Beatnik company.

o Recommendations:

Everyone in this technology's developer community is either waiting for the open source player or somehow working on getting it out.

---

o Describe the platform.

The Beatnik platform consists of:

- Beatnik Player - A browser plug-in that allows users to play Beatnik audio content. The Player is also available as an Xtra for Director and Shockwave.
- Beatnik Audio Engine (BAE) - A software audio engine for synthesis, mixing and interactive control of audio elements.
- Beatnik Editor - Authoring software for creating files in Beatnik's format.

Size and Format

The Beatnik Player is a fairly large plug-in (2.18Mb). Once it has been downloaded and installed users can play **.rmf** (Rich Music Format) files, as well as MP3, WAV, AIFF, AU and MIDI files. RMF is Beatnik's proprietary file format. It's a hybrid format that incorporates a standard MIDI file along with custom audio samples that are triggered by the MIDI data during playback.

Audio and Sound Synthesis

The player contains a software sound generator consisting of a multitimbral wavetable synthesizer with two built-in sound banks. The first bank is a General MIDI compatible sound-set that can be used to play either MIDI or RMF files. This capability makes Beatnik very powerful over low-bandwidth internet connections, allowing the user to download only a small file that plays the player's internal sounds. A second bank, the 'Beatnik Special Bank' is an offshoot of a General MIDI sound set. It provides alternative settings of synthesis options (for example, the electric piano is enhanced by a panning tremolo). The third sound bank is the 'Custom Instrument Bank' that can be used to create custom instruments with the built-in samples or any audio sample imported in the RMF. The power of the Beatnik interactivity relies on the capability of realtime control of the various parameters by MIDI events using JavaScript, Director's Lingo or Flash ActionScript code. Since the RMF is built around a MIDI file, the user can interact with the Beatnik Audio Engine, changing the song's global tempo, pitch and volume, as well as individual track parameters, like pan, volume, controller data and the capability of mute and solo MIDI channels.

The Beatnik Audio Engine

The Beatnik Audio Engine (BAE) is the core of the Beatnik platform, being also part of the technology inside the Beatnik Player. It includes a software-based synthesizer, a 64-voice stereo mixer that can playback the internal sounds and custom samples, and an effects processor and sample rate converter. It provides playback capabilities for various formats, and is a powerful and interactive player for sound, voice and music. The applications built on the BAE are supported by numerous platforms, including Windows, MacOS, BeOs, Windows CE on PocketPC, Java, Liberate, WebTV, Internet Explorer and Netscape browsers; and processors, like ARM, StrongARM, TI DSP, Pentium, PowerPC, Sparc, MIPS and 68040. Besides desktop computer applications, the BAE is one of the emerging technologies for audio in mobile devices.

Tools for production

The last item of the Beatnik system is the Beatnik Editor, the authoring software for creating RMF files. This tool combines MIDI sequences and custom digital samples in the resulting RMF format. The MIDI score needs to be created in a MIDI sequencing program and imported in (as a Standard MIDI file) to the Beatnik Editor, that compiles it into the finished file. The custom sound bank can then be used to edit and store imported audio samples that will reside within the RMF file. Using custom samples can significantly increase the size of the RMF file, however. This increase can be minimized by applying MP3 compression to samples in Beatnik Editor. The MP3 compression encoding rate can be set differently for each sample, allowing for greater flexibility and the scalable high-fidelity encoding needed for each instrument.

The Beatnik Editor can be linked to an external sequencer, allowing the instruments to be played in real time. "Real time" is a bit of a misnomer. There can be significant latency between the time that a note is played on a MIDI controller until the note is actually heard because of the delays inherent in software synthesis. It can, however, be very useful in the production environment, as it makes it possible to sequence MIDI data while hearing the Beatnik's instruments as they are intended to sound in the final file. The Editor has a built-in effects processor capable of producing some presets for reverb and chorus.

A limitation of this feature is that there is only one effects processor that affects all MIDI channels, so the only independent control over a channel's reverb is via the effects send which can be set on and off independently.

The Beatnik Editor can also be used to create banks of triggerable samples that can be played by programming, adding even more to the interactive capabilities of this platform. This method makes it possible for individual notes or sound effects to be played by the user from any built-in or custom instrument. When an RMF file is created containing no digital samples the result is a very small size file that, although very similar to a MIDI file, retains all the interactive possibilities of this format.

o What kind of experiences can this platform deliver over the web? What is and is not possible using the platform?

Beatnik can deliver a wide range of audio experiences on the web. The possibilities vary from site sonification, like mouseover audio cues and background music, to advanced and very sophisticated interactive experiences using JavaScript control, among them the popular remixer applications. Beatnik is not really an interactive environment in and of itself. It's a MIDI-compatible software synthesizer that is accessible from within various web-related interactive authoring environments. The RMF only contains the MIDI and the audio information created by the developer. Beatnik's interactive capabilities must be driven via programming.

All the interactive programming is handled by the Beatnik's Music Object API, a small JavaScript file that allows the control of the Beatnik Player by user defined JavaScript messages. This software acts like a link between the Beatnik Player and the HTML page that will control it. In the case of Beatnik Xtra for Director, messages are sent directly to the Music Object from Lingo. The advantage of this approach is that browser incompatibility is not an issue.

According to Beatnik's documentation, the Music Object API can be divided into four classes: instance methods, instance properties, static methods and static properties. Some of these methods can be used with all of Beatnik's supported file formats, some with the MIDI-based formats and some only with the RMF file. Some examples of the most important methods that define the possibilities are:

- fadeTo - fades the overall instance to a new volume level, defined by a specified amount of time.
- playNote – lets you play a note on a MIDI channel, designating the note number, velocity and note duration.
- setVolume – set the overall playback volume of the Music Object instance.
- setChannelMute – used to mute or unmute a specified MIDI channel.
- setChannelSolo – used to solo or unsolo a specified MIDI channel.
- setController – set a value to a designated MIDI controller on a MIDI channel.
- setTempo – lets you set the overall playback speed of the instance.
- setTrackMute – used to mute or unmute a specified track.
- setTrackSolo – used to solo or unsolo a specifies track.
- setTranspose – lets you transpose the overall song's tuning.

In addition to these and other realtime song, player and track control methods, there are also a number of playback state methods, designed to return boolean values (including getChannelMute, getController, getTempo, getVolume, etc,) that can further increase the developer's possibilities for interactivity.

There are, of course, limitations to this technology. Since Beatnik is designed for interactivity, there are no streaming capabilities available. The streaming media market, well represented by others technologies like Real, QuickTime and Windows Media, do not seem to be the target of the company; however, this capacity would certainly expand Beatnik's interactive power. The primary limitation of this platform is more related to browser incompatibility and the installed base of users of the plug-in, as we will see later in this document. As mentioned earlier, however, Beatnik Xtra for Director is not affected by browser issues. Although the Xtra must be downloaded for use with Shockwave, the process is relatively seamless, and is much less complicated than installing a standard plug-in.

o How stable & reliable is the platform?

Beatnik is a very stable platform, with a strong and efficient authoring software for development. It is also very well documented and has an active community or online developers providing support. However, its stability is closely tied to the OS and, most important, to the browser running this plug-in. Considering Internet Explorer's recent move away from plug-ins, using Beatnik may not be an option in the near future.

o What issues (technical, usability-related, economic, other) might stand in the way of adoption by users? By developers? By content providers, clients, web hosts? How many users does the platform already have? Is it well established and ubiquitous, or marginal?

The Beatnik system is not an easy technology to master. The process of creating a MIDI file, importing it to the Beatnik Editor together with its audio samples and the final export as a RMF file requires some experience, as well as the JavaScript or Lingo knowledge needed for the subsequent manipulation of the RMF and the Music Object API by programming. However, the various components of the platform are well documented, and with enough time and the practical projects a developer can become quite skilled in the mechanisms needed for authoring in the Beatnik's environment. With all its interactive power, the platform seems to be very well accepted and is respected by the web developer community, especially the audio professionals.

From the user's perspective, the platform is suffering from compatibility problems with the newest popular browsers, Internet Explorer 6 and Netscape 6. This seems to be the greatest issue that might stand in the way of adoption by the average consumer. In a commercial project for the internet, a new medium that are still very visually oriented and is still discovering its audio's capabilities, this obstacle can add a new dimension to the deficit of the plug-in's popularity.

o What is the platform company's strategy for web audio?

Beatnik seems have moved beyond the web to focus on the mobile devices market. In the desktop arena, the plug-in will work only in Netscape or Internet Explorer browsers below their versions 6.x, leaving a great number of users dealing with incompatibility issues. In February 2002, Beatnik announced that they would release the Beatnik plug-in to the open source community. At the time of this writing, it's too early to tell whether this approach with succeed in keeping the plug-in available for web use.

o What are the capabilities of the authoring tool in relation to web audio development? What is the quality of the interactivity of the platform?

The Beatnik Editor, the platform's authoring tool, in its new 2.1 version, was released in June of 2001 and is the only software capable of producing an RMF file. The most common method of using the editor is by importing MIDI files and custom audio samples and internal patches. RMF files can also be created either by converting other imported formats (MP3, MIDI, AIFF, AU). Beatnik Editor is a very robust, full-featured tool for building custom instruments from audio samples. The synthesis engine gives the user control over volume, pitch and filter envelopes; multiple LFOs; filter resonance; and mapping of multiple samples across specific key ranges.

We believe that this platform is the greatest technology for interactive experiences over the web. The quality of the interactivity is superb, since the platform has so many features designed for interactivity that open huge possibilities for the designer's creativity and programming.

o What kind of interactivity are users actually interested in?

It seems that many users are interested in simple remixers because they have not been exposed to much else. Other kinds of interactivity currently available in examples of Beatnik and Beatnik hybrids are: game style music that changes depending on location of the mouse in an image map or similar idea, buttons or mouseover sounds that change from visit to visit (or reload to reload of a web page) or rollover to rollover, and various kinds of created musical instruments, which often just amount to remixers with a fancier interface. This seems to be because the underlying structure of most programming for RMF files has to do with it's use of a standard MIDI file as it's storage medium, which lends itself to remixes, turning tracks on and off and transpositions or changes of tempo.

The flexibility to create other kinds of interactive environments such as algorithmic or user driven compositions – for fun or commercial purposes – comes in the use of the banks of triggerable sounds, which can be somewhat more independent of the time based storage system of a MIDI file. Other kinds of systems like Koan or MPEG-4 may not have this and may be more flexible in this regard.

Post Mortem: HTML+TIME DirectMusic Player

Authors: Steve Horowitz and Alexander Brandon

Note: *This summary was written before Microsoft's decision to stop supporting the DirectMusic player in version 6 of Explorer. HTML+TIME is still useable but not with DirectMusic. Still, the authors felt that the HTML+Time DirectMusicPlayer was a significant technology, and should be included in this document, along with this disclaimer:*

"Developers considering working with DirectMusic for the web, should be aware that current browsers are not supported by the player, and the code is not being maintained. You basically use the system at your own risk."

o Pros

- HTML+TIME DirectMusic player provides a stable platform for interactive music delivery over the web.
- DirectMusic is a known commodity with a solid track record of use in games.
- Use with Flash is possible without the need to download another plug-in.

o Cons

- This will only work within the Internet Explorer browser 5.5.
- The tools to help compile and audition games and presentations for web distribution are complex and clumsy.
- Microsoft does not seem to have any interest in promoting this technology.

o Recommendations

HTML+TIME provides a useful multimedia scripting language and audio interface within the Internet Explorer browser. What is needed most of all is:

- A player for DirectMusic content.
- A new tool to help compile and audition games and presentations.
- A strong web strategy from Microsoft. HTML+TIME could be a useful tool for delivering games, presentations and interactive web pages. In fact, it has the potential to corner the market. A push from Microsoft toward software developers and creators of interactive web content would be a logical next step.

---

o Describe the platform.

IE5.5 is the only supported browser, and *Windows* the only supported OS. *DirectX* is the set of multimedia API's for Windows that include DirectMusic and HTML+TIME. *DirectMusic* is a DirectX technology for playing MIDI and audio interactively. *DirectMusic Producer* is the authoring program for DirectMusic content. *HTML+Time* is the scripting route for using IE's enhanced multimedia functionality (essentially Microsoft's custom implementation of SMIL).

HTML+TIME allows you to place media within a set time nexus. Built upon DirectX this allows for consistent placement and playback of DirectMusic segments, with synchronization of audio and visual materials.

o What kind of experiences can this platform deliver over the web? What is and is not possible using the platform?

DirectMusic Producer gives you the audio resources to author MIDI and DLS banks and place then interactively in runtime documents. Some key benefits of using DirectMusic content in Web pages are the following:

- DirectMusic content (music or sound effects) can be created such that it dynamically changes each time it is played.
- DirectMusic content sounds the same on all computers regardless of the hardware used by the computer.
- DirectMusic content can be contained in individual components called segments. These segments can be layered on top of one another, providing a flexible and powerful model for content composition.
- Choose from one to 80 performance channels (channels) rather than the one through 16 channels offered by the MIDI format.

HTML+TIME defines a schedule, or timeline, for all the affected elements to follow. The document timeline starts as soon as the page loads and continues to progress as long as the browser renders the page. You can specify that the timeline will be tightly synchronized with other elements or the entire document. By default, media files that are not ready to play when scheduled will slip along the timeline and begin playing as soon as they become available. You can pause and resume the document timeline using methods from the object model.

HTML+TIME can be used to control various aspects of playback directly from the Web page. Here is a list of examples of things that HTML+TIME can control when playing DirectMusic content.

- Control the timing, duration, and activation/ending of DirectMusic segments.
- Create transitions between the playback of segments.
- Control the volume, speed, and mute during playback.
- In Web pages, control like this can be accomplished by playing the DirectMusic content inside of a t:AUDIO element
- Attributes/Properties like t:DUR
- Methods like beginElement endElement
- Events like onbegin and onmediacomplete

The problem is that this interactivity comes at a price: relatively large sound banks are needed to create realistic scores via MIDI. This is not a problem for disc-based games, but downloading large wavetable (DLS) samples via the web is just not practical on the dial-up connections still vastly prevalent to many users. This severely limits the effectiveness of a DM file compared to even a 10 second pre recorded compressed Flash or Shockwave file.

o How stable & reliable is the platform?

We have not yet seen anything authored this way. However, IE 5.5 and 6 are stable browsers, and DirectMusic has been used in many PC games.

o What issues (technical, usability-related, economic, other) might stand in the way of adoption by users? By developers? By content providers, clients, web hosts? How many users does the platform already have? Is it well established and ubiquitous, or marginal?

We believe the option of having media playing through the browser without plug-ins is a strong incentive for developers and producers. At this early stage, however, the user base doesn't exist yet.

With very little real world sample pages or code to look at it is hard to determine how difficult it might be to author complex structures. The degree of headache to the programmer will determine if HTML+TIME is going to be a viable platform.

The fact that DirectMusic is already used in game production (PC and X-Box) could be a big plus as well. Microsoft really needs to make the next move and evangelize the product.

Another obvious complication is that this technology is only available on Windows. Companies would have to be willing to turn their backs on Macintosh users. Also, as a proprietary Microsoft solution, it's unclear (or unlikely) that this technology will be available for the next generation of platforms (mobile devices and set-top boxes) that may be running Linux or Java.

Another barrier to entry for DM, like all other platforms for use on the web, is the fact that its user base is mostly on low speed connections. Getting any amount of audio at all into 1/2 of the homes in the US is an achievement in itself.

On the developer end, DMP has always been a bit confusing with a fairly steep learning curve to achieve truly unique results. In addition, new terminology must be learned in order to write music, and these terms aren’t necessarily in line with current accepted sequencer terminology.

o What is the platform company's strategy for web audio?

With respect to DirectMusic, Microsoft’s strategy is very unclear. Certainly DM was not designed for web audio, but it is actually somewhat better suited for web use than some other programs simply because of its flexibility and small footprint (MIDI).

However, it is understandable that Microsoft, and the rest of us as well, may need more proof that web audio is a profitable and viable market before investing a lot of effort in it.

o What is the quality of the interactivity of the platform?

The effectiveness of DirectMusic (DM) over the web has yet to be determined. No currently active websites have used DM to any degree. But DM has proved itself in games, and seems have a good track record after 4 years of research and evangelism.

The capabilities and future of HTML+TIME are discussed in some detail on the MSDN web site:

http://msdn.microsoft.com/library/default.asp?url=/workshop/author/behaviors/reference/time2/time_dmusic_ovw.asp

HTML+TIME 2.0 is based on the HTML+SMIL language profile:

http://www.w3.org/tr/2000/wd-smil-boston-20000622/html-smil-profile.html

SMIL 2.0 is the successor to SMIL 1.0. HTML+TIME 2.0 is the successor to HTML+TIME 1.0. Note HTML+TIME 2.0 is available in minimal installations of Internet Explorer 5.5 and Internet Explorer 6.

o Conclusions

We believe that traditional console game technologies and web-based online game technologies will begin to look more similar as the platforms merge for online play. Also, the bandwidth issues that separate online from console and PC technologies will be less of a factor every year as the pipe grows wider for more people interested in online interactive experiences.

We think this platform could offer audio designers a chance to reach a large number of users and a chance to tailor the interactivity along the way. In the long run, it may also provide a link between the console game world and the web world if it simplifies the task of porting games between the XBox to the web.

Imagine if we were to bring a very simple idea to an online producer, let's say a "name that tune" game. Using PCM files (wav or aiff) would be prohibitive due to the large file size needed for so many tunes. So MIDI would seem to be the solution. GM is available, and the piano sound might not be so crucial to the flow of game play that we would have to worry about the differences in GM sounds from PC to Mac. However, It might be nice to have a little bit of video tied into the overall presentation, as well as a DLS bank that could interactively trigger player responses in runtime.

Easy so far, right? Well, what are the options for authoring such a game on the web within the browser?

- **Flash** -- Forget it, can't play MIDI files in Mac IE due to JavaScript limitations
- **Director** -- Can be built using the Beatnik Xtra (140k download or the new SequnceXtra GM MIDI player also about 150k download). This is becoming more and more compelling as a gaming platform. But iit would be better to avoid the downloads if possible.
- **Beatnik** -- Works great but we would have to use the plug in conjunction with Flash (often called Flashnik). It's a huge battle to get producers to go for two plug-ins instead of one. Also Beatnik might not survive the transition to open source and add support for IE6 or Netscape 6. Oops this seals the deal, won't work.
- **Real (using MP4)** -- (Just adopted in DECEMBER 2001) (see other team report) Might be great, this is also a very new standard of structured audio.

Considering the shortcomings of the options above, you can see the potential appeal of HTML+TIME. I can write my MIDI files in my favorite sequencer, import my DLS sounds and MIDI file into DirectMusic Producer and deliver the experience through a browser without plug-ins. That is a very powerful inducement for most of the producers working on the web today.

About the Authors

LUIS BERGMANN

Luis Bergmann is a composer and sound designer who works with top Brazilian media companies on internet and television productions. His online and offline works include Coca-cola, Volkswagen, Honda, Peugeot, Nokia, IBM, and Renault, to name a few. His internet credits also include audio programming and implementation on Macromedia's Flash and Director. Luis Bergmann works at Lua Nova, a top Brazilian audio production company, where he is the creative director and runs the audio department, Lua Web. He is also a producer for several music projects on the company's record label, Lua Discos.

JAN MARTIN BORGERSEN

Jan Martin Borgersen is a senior web consultant and an interface development lead with Razorfish, Inc, in San Francisco, California. His client list includes the Cisco Systems Web Framework Group (www.cisco.com), Interactive Audio Special Interest Group (www.iasig.org), and Ventro/Chemdex. Jan joined Razorfish in February 2000 after working at Sun Microsystems as a software engineer on the Java Sound API. Prior to Sun, he worked at IBM on various application development tools, including NetObjects BeanBuilder and VisualAge for Distributed Smalltalk. His career focus has been to understand how users and developers interact with distributed computing environments, and to provide the technology to make those interactions more efficient, more effective, and more fun.

ALEXANDER BRANDON

Alexander Brandon has been writing game music since 1994. He has worked on over a dozen titles and several of the highest ranking computer games in recent years including "Unreal", "Unreal Tournament", and "Deus Ex". He is currently director of audio on "Deus Ex 2" at Ion Storm, headquartered in Austin, Texas. He has written articles for Gamasutra and Game Developer Magazine, has hosted roundtables and spoken at GDC (www.gdcconf.com), is on the steering committee of the Interactive Audio Special Interest Group (www.iasig.org), and is on the board of directors (acting membership director) of the Game Audio Network Guild (www.audiogang.org). Alex focuses on advancing interactive audio on the aesthetic, technical, and popular fronts, in that order, and has had some success in all three.

CHRIS BURKE

As the head of Bong + Dern (www.bongdern.com), Chris has written and produced music, sound and interative audio works for film, TV, the internet and radio. His "Kill This Language" and "Peppermint Lounge" (Altoids.com) were amongst the early, advanced applications of interactive music on the web. In 2000 MTVi hired Bong + Dern to create the interactive remix of "I've Seen It All" by Bjork, which became the first 'webeo' and won Site of the Day from Shockwave.com. Chris has created sound design for over 70 CD-ROMs and scored 7 feature films. He has four record releases including "Idioglossia" and Sire Records' release of his band Glorified Magnified's "All Wave Super". These projects lead to collaborations with producer Don Was and Rage Against The Machine's Tom Morello. Chris lives in Brooklyn, NY and is currently performing live laptop music under the name 'glomag'.

SPENCER CRITCHLEY

Spencer Critchley, a partner in Palo Alto Media Group Inc., is an award-winning producer and creative consultant with experience in digital media, film, broadcasting and the music industry. He has directed creative groups or led audio production at BeVocal, Beatnik, Silicon Graphics, Silicon Gaming and CCC/Viacom. Among the projects he has worked on are the redesign of BeVocal's voice applications interfaces, the multiple award-winning Choosing Success CDROM for CCC, the SGI/Time Warner/ATT interactive TV system, and web content for David Bowie, Moby, Britney Spears and Yahoo! He has also served as a freelance writer/broadcaster for the Canadian Broadcasting Corporation, National Public Radio and others, winning awards for investigative journalism. Spencer has been a composer/producer for Warner-Chappell Music, and was composer and director of audio post production for the documentary film "Blink", which won an Emmy at the 2001 ceremony.

JEFF ESSEX

Jeff Essex, Creative Director of audiosyncrasy, creates music, sound effects and voiceover for multimedia. He is credited on over 50 CD-ROM titles, including products from Disney Interactive, Virgin Sound and Vision, Mindscape and 3DO. A veteran of MacroMind and Macromedia from 1990-93, he served two years as Technical Support Lead for Macromedia Director before starting his own production company. For the past five years he has worked primarily in online entertainment and audio interface design with clients including Walt Disney Internet Group, Nickelodeon Online, Shockwave.com, Beatnik, Sun Microsystems, Apple Computer, Red Sky, @Home, Jet City Studios and Silicon Graphics. He has broad experience in helping clients navigate the thicket of multimedia audio options, and has brought sound to web sites with Flash, Beatnik, QuickTime, Shockwave and 3D positional audio. He is an active member of the Interactive Audio SIG. His book, *Multimedia Sound and Music Studio*, (Random House/Apple New Media) is the definitive guide to multimedia audio production. It won the prestigious Computer Press Award for Best Advanced How-To Book of 1996.

STEVE HOROWITZ

Steve Horowitz is a Grammy award winner who's work includes music and sound design services for some of today's top media companies. He was one of the engineers on "True Life Blues, The songs of Bill Monroe" - Grammy winner for best bluegrass record 1996 featuring: David Grisman, Vaser Clemens & Peter Rowen. His Nickelodeon online Credits include: The Crimson Chin (TV episodes & online Webisodes produced by Butch Hartman), and games for Jimmy Neutron, Dora the Explorer, Blues Clues and more. Horowitz serves as the president of The Code International INC and Chairman of the Interactive Audio Special Interest Group. He is currently running the Audio Department at Nickelodeon online and serves as music director for the MTV reality series "I Bet You will".

DAFNA NAPHTALI

Dafna Naphtali is active as a singer, sound artist, composer, and audio programmer/engineer. She performs and composes using her own custom Max/MSP programs for sound processing of voice and other instruments -- in venues and festivals in NY, Germany, Canada, Holland and Russia - with her various projects and with the digital chamber punk ensemble What is it Like to be a Bat? that she co-leads with Kitty Brazelton. For these projects she has received commissions and awards from NY Foundation for the Arts (Fellowship in Computer Arts 2001), NY State Council on the Arts, Meet the Composer, American Composers Forum and STEIM. She was Project Manager and did research and production for Audio Technologies at MCY.com (on-line music distribution startup '98-'01) and Chief Engineer at New York University's Music Technology program ('96-'98). She regularly gives workshops at various universities (Columbia, School of Visual Arts Computer Arts program and NYU's Interactive Telecommunications Program and University of Nijmegan in the Netherlands). She teaches, programs and consults about Max/MSP: at Artist in Residence programs at Harvestworks (not-for-profit media arts center in NY) and at Engine 27 (a new experimental 16-channel sound gallery in NY). She also teaches at Bard College (starting Spring '03) and New York University -- where she earned her BM (voice) and MM (Music Technology). Dafna can be heard on the Tellus CD - José Halac "Dance of 1000 Heads" and on a forthcoming release from What is it Like to be a Bat? on John Zorn's Tzadik label.

MICHAEL SWEET

Composer Michael Sweet has won numerous Awards for his audio work including the BDA Promax Award 2000 for Best Sound for a Network Package (HBO Zone) and the Best Audio Award 2000 at the GDC Independent Games Festival.

Educated at Berklee College of Music, Michael's early digital artistry has led him to ground breaking experimental work in the creation of interactive scores for art exhibitions. His collaborative sound installations have traveled around the world, including the Millennium Dome in London and various galleries in New York, Los Angeles, Florence, Berlin, Hong Kong, and Amsterdam.

Michael is currently the creative director of his own Silicon Alley audio workshop AudioBrain. On the Internet Michael's interactive audio can be heard on many award winning games and web sites, includes Sesame Workshop's MusicWorks, Shockwave's BLiX & Loop, RealArcade's WordUp, and many games for the Cartoon Network. In contrast, Michael's linear audio work in broadcast can be heard in many commercials and network identities, including HBO Zone, Comedy Central, CNN, General Motors, Kodak, AT&T and the X-Files.

VALENTIN SCHMIDT

Valentin Schmidt was born in born in Heidelberg in 1968. Between 1987-1995 he studyed physics, psychology and informatics at the universities of Göttingen, Hamburg and Berlin. From 1991-1997 he spent time organizing underground techno events in the urban-industrial playground of post wall eastern Berlin. Since 1995 he has been working as multimedia conceptor and programmer, with focus on e-learning and interactive audio toys.

MARTIN WILDE

Martin has immersed himself in the field of audio programming for over two decades. His formal training in computer music and psychoacoustics has uniquely prepared him for the challenges of delivering audio across the wide spectrum of today's technologies. Martin is a musician who brings a wealth of technical understanding and experience to bear in all his professional endeavors. He has designed, developed and shipped sound and music software engines for computer game titles on a multitude of platforms. Martin holds two United States patents, a Masters degree in Computer Studies in Music, and is the chairman of the new AES Technical Committee on Audio for Games. Martin is an active member of the AES and the Interactive Audio Special Interest Group. He regularly publishes articles on game audio, most recently in the AES Journal and on the Sonify.org website. Over the years, he has made numerous presentations to the AES, the Game Developers Conference, the Interactive Multimedia Association, and the International Computer Music Association.

DAVID YACKLEY

David Yackley is a Program Manager for DirectX Audio at Microsoft. He has been active in interactive music for over a decade, as a consultant for the Blue Ribbon Soundworks (1990-95) and with the DirectMusic team (1996-present). A graduate of the Eastman School of Music, David is a composer, arranger and keyboard player. He lives with his wife Tracey and children Laura, Will and Caitlin in Redmond, WA.