

OpenCourt: A Developer's Introduction

Fred Buhl
Court Technology Officer
8th Judicial Circuit of Florida
10/30/09
opencourt@circuit8.org

Contents

Introduction.....	1
Background.....	1
The Transcript.....	1
The Problem.....	2
The Solution.....	2
Theory of Operation.....	3
Hearing Room Setup – A Concrete Example	3
Recording Software	5
The OpenCourt Web Front-End.....	7
Recording an Event	7
Playing Back a Recording.....	11
Currently Implemented Features	12
Required Features – Unimplemented	14
Expected Features – Unimplemented	17
Wish List	19
Current Development Work & Code Status	22
Development Milestones	24

Introduction

The purpose of this document is to provide an introduction to the OpenCourt project for a potential developer. I'll provide some historical background, discuss the current problem in our digital court recording systems, and describe current status of OpenCourt development and planned enhancements.

Background

People have been making detailed records of court proceedings for over 2,000 years, with the goal of taking down every word that is spoken. Originally, these records were written in various forms of shorthand. In the 1800s, the first stenotype machines were developed, which, when operated by a skilled court reporter, could rapidly produce a typed phonetic record of the words spoken. (In Florida, for certain types of court events, the presence of a stenographic court reporter is still required.)

Later, reel-to-reel and cassette audio recording systems were put into use, as backup for a court reporter, and for certain events even replacing the court reporter. VHS video tape recordings came after that, which added video to the mix. (When I arrived here in 1998, cassette tapes and VHS recordings were the only recording systems in use.)

In the 1990s/early 2000s, as digital audio recording became widespread, vendors developed digital audio recording systems specifically for courtroom use. Eventually, video was added, which led to the Digital Court Recording (DCR) systems we use today.

For several years, Florida's Office of the State Courts Administrator (OSCA), in conjunction with the 20 Trial Court Technology Officers (including your humble narrator), have developed a set of standards for DCR systems. These standards are revised as new technologies and requirements dictate. Vendors must meet these standards to be acceptable for use by the Florida State Courts System. This ensures that regardless of vendor, minimal requirements for a good digital recording will be met.

The Transcript

All these systems, from shorthand to DCR, have had a transcript as the ultimate goal. A written record of the court event, it has stood as the "official" record for centuries.

To make a transcript from a recording, it's helpful to have a player with a foot pedal control to start and stop a recording, and skip backwards and forwards. Almost all DCR systems have a foot pedal capability to ease transcript production.

The future of the transcript as "official" record is in doubt. Today, an audio/video recording of an event can be burned to a CD or DVD for review. Burning recordings to a CD or DVD also facilitates outsourcing of the creation of the transcript, for those customers who still prefer reading transcripts to watching recordings. Recent rulings by the Florida Supreme Court required that audio/video recordings be made available upon request. [See <http://www.floridasupremecourt.org/decisions/2009/sc08-1658.pdf>]

Certain aspects of a court proceeding are not part of the ‘official’ record—for example, addresses, social security numbers, and children’s names. Video editing software can be used to redact certain information on the recordings. This redacted recording can then be released directly, without transcribing, at a lower cost to the customer (redaction still takes time and effort, however).

The Problem

Because DCR is a small and specialized market, the hardware and software costs are high. (In the Eighth Judicial Circuit, with 50 separate venues, we spent \$2M on our current DCR infrastructure, with \$440,000 of that cost being software). Maintenance costs for the system are also high (currently 12% of installation cost yearly); we are currently running our systems without any maintenance contracts in place.

In addition, DCR software doesn’t always behave well over a wide-area network. The software we’re using chews up large amounts of bandwidth (multi-megabits per second), adversely affecting network performance, and restricting our ability to cover court events remotely.

While inexpensive (and even free) software exists to record audio and video, nothing readily available has the specific feature set we need.

The Solution

It occurred to me that since Open Source software exists to do digital audio/video recording, it could be modified to meet the needs of the court system. (For many years, I had a Linux system at home running the excellent MythTV package (<http://www.mythtv.org>). Each evening, after spending the day cursing the costs of my digital recording software, I’d go home to watch digital recordings made using free software.) One major stumbling block was that we need four-channel audio for courtroom recordings--the four channels make it easier for transcribers to isolate an individual voice when people are speaking simultaneously.

In October 2008, we realized that we wanted to record events in a hearing room. Not having the required \$15K+ to implement using our existing vendor, we were vexed. I eventually realized **a)** we only record two audio channels in a hearing room, and **b)** there’s free software that can record two channel audio with video.

With approval from my Court Administrator (my boss) and Chief Judge (his boss), I immediately started work on the OpenCourt project. For recording, we are using the excellent VLC media player produced by the VideoLAN project (<http://www.videolan.org/vlc>). I was able to create a web front-end and a network-controlled launcher program which allows court reporters (or even a judge’s judicial assistant) to start and stop the system via a web browser. The system creates audio/video recordings that are compatible with Adobe Flash and Microsoft QuickTime, and are viewable in a web browser (I wrote a simple Flash player application to view and control the recordings.) We recorded our first hearing on March 4th, 2009. Currently, the system is in use at two venues, with plans to expand to another 10 venues in coming months.

The system we've deployed uses inexpensive hardware, with consumer (or "prosumer") grade equipment (Logitech Webcams, M-Audio USB audio interfaces, small-form-factor PCs). By using commodity hardware, I can lower the hardware cost significantly. By using Open Source software, I can lower the software cost to \$0.

Theory of Operation

Let's discuss how OpenCourt works at present.

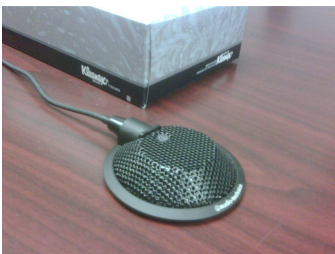
Hearing Room Setup – A Concrete Example

A hearing room is a small room, usually part of to a Judge's chambers. Here's what one looks like:



As you can see, not a very big room. So how do we record it digitally?

First, we need **two** microphones (two are required for hearing rooms by Florida DCR standards). I'm using the Audio-Technica U841a mic:



This is a boundary condenser mic, which is ideal for situations like conference tables. (We could drill some holes in the table to hide the cables, of course.)

These mics won't plug straight into a PC, so you need a "mixer" (really a preamp & audio computer interface)

For hearing rooms, I use the M-Audio MobilePre USB:



(You may notice the "phantom power" indicator; these condenser mics need a 48 volt power connection to work.) The MobilePre powers the mics, takes the signals, and converts them to digital signals which get to the computer via a USB cable.

To get video, you need a camera. For this room, I used Logitech Quickcam Pro for Notebooks:



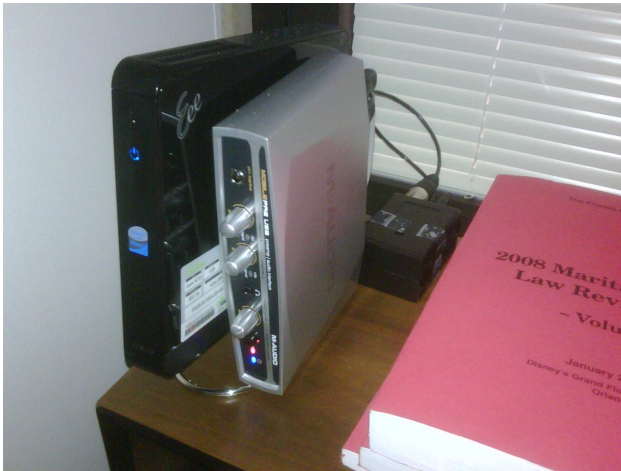
This is a tiny, inexpensive camera (2¼" high!), but it does 640x480 resolution video at 30 frames per second, so it does what we need. I would now spec the Logitech QuickCam Pro 9000, which would be easier to wall-mount (I ended up using Velcro here). Since it's on the opposite side of the room from the computer, I had to daisy-chain a couple of powered USB extension cables to reach. (Belkin 16.4' USB Active Extension Cable, F3U130-16).

The computer I used was the ASUS Eee Box:



It's netbook-grade, and just powerful enough to do the job, and I would currently recommend something along the line of a Dell Studio Hybrid if you want to go with an inexpensive small form factor PC—it's a little beefier.

By a clever use of self-adhesive Velcro tape, I put the whole system together thusly:



The small black boxes behind the PC & mixer are the power adapters for the microphones, a slight disadvantage of the U841a mic. These systems are set up with no monitor, mouse, or keyboard. Total hardware cost? \$1,045.48; Even with beefier computer hardware, you're still looking at <\$1200 per hearing room. Total Power consumption is around 15-19W when running, which is less than two compact fluorescent lightbulbs. (All the attached devices are powered via USB from the computer's power supply). The computer is networked via Category V cable to the building's LAN, and has a network drive mapped to some place to store recordings (a file server, or network attached storage).


Recording Software

The system is running Windows XP SP3, along with two other pieces of software:

- 1) VideoLAN's VLC Media Player (0.8.6i) (Free from <http://videolan.org>.)
- 2) The OpenCourt Launcher (developed here)

The version of VLC we're using is old, primarily because the newer version had some stability issues under Windows when used for recording. This has since been fixed, but some other issues remain, and we're still happy with 0.8.6i.

The OpenCourt Launcher runs at startup; it's currently a Windows console app:



```
63 OpenCourt Launcher
SUCCESS: The process with PID 2460 has been terminated.
SUCCESS: The process with PID 3360 has been terminated.
SUCCESS: The process with PID 440 has been terminated.
message is HTTP/1.1 200 OK
Date: Fri, 18 Sep 2009 17:58:53 GMT
Server: OpenCourt-Launcher (Windows XP)
Last-Modified: Fri, 18 Sep 2009 17:58:53 GMT
Accept-Ranges: bytes
Content-Length: 82
Connection: close
Content-Type: text/html; charset=UTF-8
C#0 - OC-ALA-CIU-301<br>OpenCourt v0.01<br>STOPOK<br>Fri, 18 Sep 2009 17:58:53 GMT
Connection closing...
Got message GET / HTTP/1.1
message is HTTP/1.1 200 OK
Date: Fri, 18 Sep 2009 17:58:53 GMT
Server: OpenCourt-Launcher (Windows XP)
Last-Modified: Fri, 18 Sep 2009 17:58:53 GMT
Accept-Ranges: bytes
Content-Length: 81
Connection: close
Content-Type: text/html; charset=UTF-8
C#0 - OC-ALA-CIU-301<br>OpenCourt v0.01<br>READY<br>Fri, 18 Sep 2009 17:58:53 GMT
Connection closing...
E:\opencourt\data>m:\web\opencourt\data\raw\np4box -raw 1 2009-09-18~13-34~ALA-CIU~301.np4
Extracting MPEG-4 AAC
E:\opencourt\data>m:\web\opencourt\data\raw\np4box -raw 2 2009-09-18~13-34~ALA-CIU~301.np4
Extracting MPEG-4 AAC-H264 stream to h264
E:\opencourt\data>REM have to rename these since the long file names don't work well on the repackge command...
E:\opencourt\data>rename 2009-09-18~13-34~ALA-CIU~301_track1.aac 1.aac
E:\opencourt\data>rename 2009-09-18~13-34~ALA-CIU~301_track2.h264 2.h264
E:\opencourt\data>m:\web\opencourt\data\raw\np4box -ipod -fps 25 -add 1.aac -add 2.h264 m:\web\opencourt\data\2009-09-18~13-34~ALA-CIU~301.n4w
AAC import - sample rate 44100 - MPEG-4 audio - 2 channels
AAC-H264 import - frame size 320 x 240 at 25.000 FPS
Import results: 29169 samples - Slices: 117 I 29052 P 0 B - 1 SEI - 117 IDR
Setting up iTunes/iPod file...
Forcing AAC/H264 SAR to 1:1...
Saving to m:\web\opencourt\data\2009-09-18~13-34~ALA-CIU~301.n4w: 0.500 secs Int
erleaving
E:\opencourt\data>REM del 2009-09-18~13-34~ALA-CIU~301.np4
E:\opencourt\data>del 1.aac
E:\opencourt\data>del 2.h264
Got message GET / HTTP/1.1
message is HTTP/1.1 200 OK
Date: Fri, 18 Sep 2009 20:40:14 GMT
Server: OpenCourt-Launcher (Windows XP)
Last-Modified: Fri, 18 Sep 2009 20:40:14 GMT
Accept-Ranges: bytes
Content-Length: 81
Connection: close
Content-Type: text/html; charset=UTF-8
C#0 - OC-ALA-CIU-301<br>OpenCourt v0.01<br>READY<br>Fri, 18 Sep 2009 20:40:14 GMT
Connection closing...
```

Which is a little crude, but since no one sees it (unless you connect via Remote Desktop Connection or something similar) it's not a Big Deal.

The OpenCourt Launcher's job is to start and stop VLC recording, repackage and upload those recordings to a central server, and report on its current status.

Any computer running the OpenCourt Launcher is a **Recording Station** – a computer that can make OpenCourt recordings.

Users don't access the OpenCourt Launcher directly; instead they interact with:

The OpenCourt Web Front-End

The Front-End was written in PHP, and is currently running on a Linux server:

The screenshot shows the OpenCourt web interface. At the top, there is a "Recording Control" button. Below it is a calendar for September 2009, with the 3rd highlighted. The date 3 is bolded, indicating recordings. Below the calendar, the text "Recordings for Thursday, September 3rd, 2009" is displayed. A table lists the recordings for that day:

Time	Location	Room	Presiding	Event Type	Case #s	Event Description	
8:29 AM	ALA-CIV	301	SMITHF	DOMESTIC RELATIONS			Edit Download
10:08 AM	ALA-CIV	301	SMITHF	DOMESTIC RELATIONS	06-DR-3307		Edit Download
12:55 PM	BAKER-JAIL	JAIL	WILLIAMSJ	FIRST APPEARANCE		TODD SARK, CHILD SUPPORT WRIT, DID NOT DO 1ST APPEARANCE	Edit Download

(Our goal is to support both Linux and Windows on the Front-End, but it is Linux only at the moment.)

Once a user pulls up the website and logs in, they'll see the OpenCourt main screen, and various options depending on what sort of user privileges have been set for them.

The Calendar control at the top of the page lets them pick any particular date; once they pick the date, they'll see the recordings for that date. (Dates with recordings show up in **bold**).

Recording an Event

The "Recording Control" button on the Front-End is what's used to start and stop recording (you only see the Recording Control button if you have permissions to start & stop recording). When you press it, you'll see:

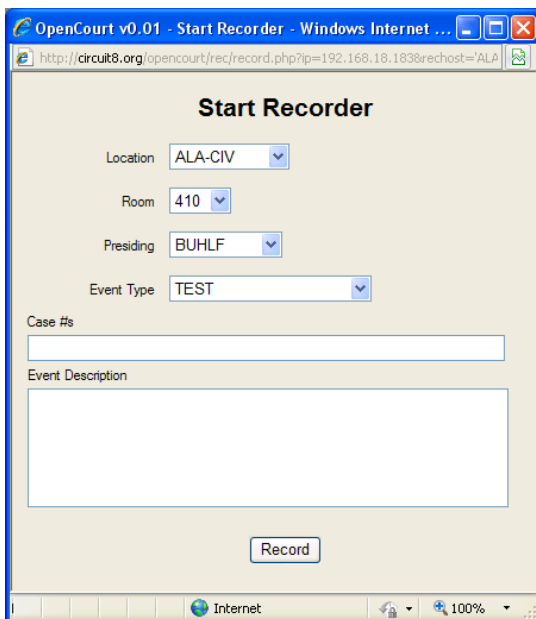
The screenshot shows the OpenCourt web interface. At the top, there is a "Back" button. Below it, the text "Recording Control" is displayed. A table lists the status of various servers:

Server	Status	Actions
ALA-CIV 301	READY	Record
ALA-CIV 410	DOWN	
ALA-CIV TEST	DOWN	Wake
BAKER JAIL	READY	Record Preview

Here's a list of the current Recording Stations and their status. DOWN means the station didn't respond when queried (the **Wake** button will actually wake the Recording Station if it has Wake-On-LAN enabled). READY means the Recording Station is ready to start recording (they'll be a **Record** button for those). If a station is recording, it'll show as RECORDING (unsurprisingly), and a **Stop** button will be displayed. The **Preview** button lets you see a slowly-updating image of the venue (currently set to update every five seconds or so; I call it "Slow-Scan", and it's designed to let someone check a venue while using as little bandwidth as possible).

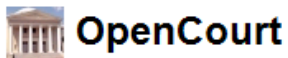
Generally, One Recording Station=One Venue. The nice thing about this is that if a Recording Station fails, it only affects one courtroom, which can be quickly swapped out (you can have a hot spare ready); alternatively, just reschedule the event to another venue. Other DCR systems have recording servers supporting multiple courtrooms, which cuts your computer count down, but means you have a bigger problem when the server fails...

To start a recording, click on **Record**. You'll see this window:



Here, you'll see the Location (usually a building), the room (individual Room #), Presiding Judge (or Magistrate, or Hearing Officer, or whatever—it's not always a Judge). There's an Event Type you can set, a place to enter case #s (which can help with searches & integration later), and a description field where you can put anything you like...

If you press the **Record** button at the bottom of the window, and the window will disappear, the Station will change status to RECORDING, and a **Stop** button will appear next to it.



[Back](#)

Recording Control

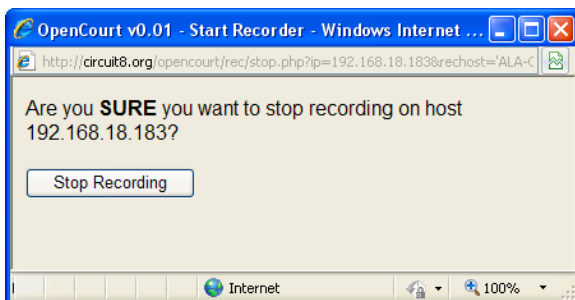
Server	Status	Actions
ALA-CIV 301	READY	Record
ALA-CIV 410	DOWN	
ALA-CIV TEST	RECORDING	Stop
BAKER JAIL	READY	Record Preview

And yes, after pressing Record, the Recording Station will start recording the hearing room.

Behind the Scenes: The Front-End sends an HTTP message to the Recording Station telling it to start recording; it also posts some of the information on the hearing (location, room, etc.) so that this can be used to generate the file name for the recording. The OpenCourt Launcher, running on the Recording Station, receives the message and launches a batch file, which in turn launches the VLC Media Player with the appropriate settings to record the courtroom. The resulting MPEG-4 file is written to the local hard drive on the Recording Station. While that recording's taking place, the OpenCourt Launcher will report back a "RECORDING" status to any Front-End requests.

Current Limitation: You can't see or hear what you're recording until afterwards. We need to be able to monitor the recording while it's being made.

When you want to stop recording, press Stop, you'll see:



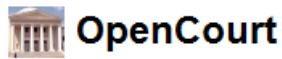
Displaying the Raw IP address isn't helpful there (we can fix that in a later release), but at least it asks for confirmation before it stops recording.

When you click **Stop Recording**, the box disappears, and the Recording Station goes back to READY.

Behind the Scenes: When the OpenCourt Launcher (running on the Recording Station) gets a STOP message via HTTP, it kills the VLC process, which stops recording and causes the rest of the batch file to run. The rest of the batch file repackages the MPEG-4 file on the local hard drive so that it'll be compatible with Flash and Quicktime (the native VLC files

aren't). It then copies the repackaged file to the networked drive the Recording Station is mapped to. The Front-End can see that network drive (currently the network drive is mapped to a NAS via NFS), so that recording will then show up on today's list of recordings. The repackaging & copying process takes a few minutes--we ultimately want to create a Flash-compatible file in both locations simultaneously, so that the repackaging and copying will be eliminated.

You'll then see the event listed on the OpenCourt Web Front-End page for that day:



Recording Control

<< < September 2009 > >>
 S M T W T F S
 1 2 3 4 5
 6 7 8 9 10 11 12
 13 14 15 16 17 18 19
 20 21 22 23 24 25 26
 27 28 29 30

Recordings for Tuesday, September 22nd, 2009

Time	Location	Room	Presiding	Event Type	Case #s	Event Description	
9:55 AM	ALA-CIV	410	BUHLF	TEST		TEST	<div>Edit</div> <div>Download</div>

Playing Back a Recording

To play back a recording, just click on the link under the Time column on the OpenCourt Front-End. The OpenCourt Flash Player will pop-up:



After enough of the files loads, you'll see and hear the recording.

We have some rudimentary controls for now, Play/Pause, Mute, and seeking backward and forwards by certain # of seconds or minutes. The time displayed is the "wall clock" time for the recording; you can seek to a specific time using the slider.

Currently Implemented Features

What we have now is a small subset of the capabilities of a proper DCR system, and thus a subset of current State DCR Standards. Here's the list of features we currently have working:

Record Two Channel Audio and Video

VLC Media Player does this out of the box; good enough for a hearing room, but not good enough for a courtroom. (We need those four audio channels.)

Store Recordings as H.264/AAC MPEG-4 files, 320x200 or 640x480, 30fps, <=384kbps bandwidth

VideoLAN can write H.264/AAC files directly (my preference is to drop the frame rate a bit below 30fps to improve the quality of the individual frames (and allow higher resolution video) given the 384kbps bandwidth limits set by the DCR standards...)

Currently, the files need to be “repackaged” (but not transcoded) to make them compatible with Flash and Quicktime, which means the recordings are not immediately available for viewing on the OpenCourt Web Front-End.

Playback/Start-Stop Control via Web Browser/Flash Player

The Front-End sends commands to the individual recording stations to turn them on and off. Playback is through the same interface.

Playback/Start-Stop Control via iPhone/iPod Touch

A “bonus” feature, tested with my iPod touch. I can start a recording from an Internet café in Timbuktu, and watch it when it's done on the same device. (WiFi recommended, as we're not streaming video, and the AT&T network isn't quite fast enough to make that painless.) When you're watching recordings on the iPod, you don't get the Flash interface—just the normal iPod QuickTime player.

Event Summary Notes

For each recording, you can make a note of which judge, which case #, what sort of event, and any other note you like.

Web Front-End to Browse Previous Recordings

Using an Outlook-style calendar navigator, you can look at recordings on any day you choose.

Context-Sensitive “Last-Use” Summary Values for each Recording Station

If the last recording on Station X was a Judge Smith Arraignment Hearing in Room 222, the next one on that station is likely to share a lot of those characteristics; this keeps you from having to re-select things that didn't change from the last hearing...

Start/Stop Tone via PC Speaker

The recording stations can now make a sound when they start and stop recording; since they're running "headless" without displays, and are started and stopped remotely, this gives feedback to the room occupants that the recording has been started & stopped. (The sound is generated from an embedded speaker, so you don't need external speakers to hear it. The sound itself is based on the fanfare sound used in the Imperial Margarine commercials of my youth.)

Preview Slow-Scan Video of Venues via Axis Network Camera

Court Reporting wanted to see video from a remote site so they'd know when to start and stop recordings. I wanted to make sure they didn't chew up any bandwidth while doing it. I currently send a 10,000 byte snapshot back every five seconds or so, which is enough to let you tell whether anyone's in a room or not, without causing any noticeable spike in bandwidth. (I call this "Slow-Scan"; catchy, no?) The refresh rate is configurable, and my plans are to make it easily configurable via venue.

This only works if you have an Axis Network Camera deployed; I don't have this working for USB webcams as yet.

ICMS (Integrated Case Management System) Integration

By adding a case # field to the Event Summary Notes, I can tie OpenCourt recordings with ICMS, our home-grown case management system, so when you pull up a case, you can access any recordings for that case...that bit needs some custom code to work with other case management systems, but the "hooks" are there...

Editing of Event Summary Notes

Allows you to start a recording and fill in the details later.

User Access Control (Viewing, Recording, Editing) tied to Venue

A given user may have permissions to see some venues, control recording on others, and edit notes on still others). I still lack a web front-end for administrative control.

Export/Edit Recording to File/DVD (start/stop, redacting audio & video)

Since the files are standard H.264/AAC, you can use free video editing software to do this. I added a "Download" button to facilitate getting the file to a client PC for editing.

Perfectly Synced Conversion to Flash/iPod Compatible Files

I had some synchronization issues on my first venue (Eee Boxes, while nicely inexpensive, just don't have enough CPU). My current recommended hardware (Dell Studio Hybrid) has no such problems.

Required Features – Unimplemented

These features are generally required by current State DCR standards but are as yet unimplemented.

Record Four Channel Audio & Video – CRITICAL FEATURE

This requires either modifying the VLC Media Player [or a similar program] to support input from multi-channel capture devices, or (preferably) developing our own recording program that has this capability.

Notes:

1. Developing our own shouldn't be too difficult, since the libraries for audio capture, H.264/AAC encoding, and MP4 packaging are Open Source and readily available; indeed, they were used to develop VLC.
2. According to http://www.kaourantin.net/2007/08/what-just-happened-to-video-on-web_20.html, four-channel audio will get mixed down to two channel in the current Flash player, although supporting multi-channel audio is targeted for change in the next major revision of Flash. We may only need four-channel control for transcriptionists; the existing player will work fine for most users. We may have to do an ActiveX control or (worst case) desktop client to get around this limitation for transcribers.

Realtime Viewing of Recording Venue with Audio – CRITICAL FEATURE

We need to either **1)** stream the video from the recording station as we encode it, or **2)** read from the file as it's being created and stream it or download it "live". The second option is preferable because it **a)** allows the recording stations to be on a non-accessible network **b)** reduces CPU resources on the recording stations, **c)** verifies that the recorded file you're creating is OK. Whether we can do either with the current version of Flash is debatable; we may have to go to an ActiveX control (or, even worse, a Windows client of some sort) to do this. That player could be used just by court monitors, as opposed to other users, who might still use the stock Flash client.

Preview Slow-Scan of Venues – WebCam (DirectShow)

I want to provide Slow-Scan video via USB attached webcams as well as Axis Network Cameras. (One difficulty: it appears that only one program at a time can open a webcam via DirectShow; to do Slow-Scan while recording from the same webcam at the same time may require resource sharing via threads, or developing some sort of shareable shim...another possibility is having the recording program write out a snapshot image at intervals to kick back to any Slow-Scan viewers...)

Event Notes – Creation with User-Specific Hot Keys

Event Notes are time-stamped text notes attached to a recording; they act as a searchable index to different parts of the recording. For many events, we'll have a "Court Recording Monitor" listening in and making these text notes. They can be used to locate past recordings (by searching on a case #, defendant name, and the like) and to seek to specific parts of a recording (a specific defendant on a "cattle-call" docket, for example). Each Court Recording Monitor will want a set of "Hot Keys" predefined with things like "JUDGE", "PROSECUTOR", "DEFENDANT" so they can hit one key and insert the appropriate text.

Event Notes – Viewing/Seek During Playback

The event notes entered by the Court Recording monitor will be visible when looking at a recording. You should also be able to click on a specific event note to advance to that part of the recording.

Event Notes/ Summaries -- Search via Database

Users will be able to search the event notes and summaries for specific text, and by time and date. This data will be stored in a relational database, although separate XML files will be made as a backup in case of database corruption.

Event Notes/Summaries – Attachments

Users will be able to attach files to the Event Notes and Event Summaries, to provide a more completed digital record of the event.

Archiving and Archive Media Management / DVD+R, Blu-Ray

We need the system to burn the recordings, notes, and summaries to DVD+Rs or Blu-Ray disks for archiving. If something is removed from the system, or data is lost, we need mechanisms to find old records and reload them. (I have some code I wrote that included this functionality for FTR Gold; I'll be adapting it for OpenCourt...)

Grooming of Local/Remote Filesystems based on available disk space

As disks fill up, we need to move or delete the contents (deleting only when we know files are properly archived). (Adapting old code I wrote for FTR Gold again...)

Simultaneous Fault-tolerant Recording to Multiple Locations

The recording program should write its output file to two locations simultaneously—ideally, one local drive and a network share or optical media. This should happen in a "fault-tolerant way", so that if one method has problems, it doesn't affect the other. **Note:** I'm thinking separate threads with separate output buffers, with some sort of "catch-up" capability when

the problem gets resolved—and a “give up for now and try again next time” option if the problem has been going on for a while. (It’s been my experience that certain kinds of disk I/O errors seem to slow down the whole machine; this may make it difficult to achieve in practice.)

Start/Stop Recording Control via Schedule (with Rules & Exceptions)

We need to be able to schedule recordings for a given venue at certain start/stop times...making rules much like Outlook appointments (for familiarity’s sake

Backup single channel audio support

Many of our existing centralized recording systems have a “backup channel”, a single channel of audio coming from the courtroom mixers; Backup channels from up to 24 venues are then stored on a separate “backup audio server”. This way if any of the “primary” recording stations fail, there’s a “backup” recording as well. This recording is only single channel audio, and no video, so it doesn’t present too many challenges. I’d like to make it voice-activated, so we’re not dependent on anyone turning recording on or off, but perhaps it could be configurable to mirror the operation of the “primary” stations for that venue (when the primary is on, the backup is on, with both off likewise).

In-Room Playback

There are times when a judge may request audio playback of an earlier recording while in a courtroom (even an earlier portion of the current recording). The recording station could play back that audio, since it’s already hard-wired to the mixer for recording, we could add connections for playback—some commercial systems use that approach. Other systems use a separate PC in the courtroom with a “playback client” on it which might be remote-controlled, or controlled from the Judge’s bench.

Standalone Frontend for Portable Systems

OpenCourt may be run from a laptop or netbook; in those instances, web-based control isn’t available. I have an early version of a “standalone frontend”, but haven’t worked with the newer VLC launching stuff I’ve been doing.

Seal/Un-Seal Records

We need to be able to keep recording from being viewed, in accordance with DCR Standards.

Log All Start/Stop Event, User, Room Use, Etc.

We need to be able to know who did what, and when. This can also help us determine how heavily scheduled a given room is, if we use OpenCourt to record every event...

Reports for Start/Stop User/Room, Viewing, Etc.

Once we start logging all these events, we need reports that will let us see the data...

Pause/Resume Recording

We need to be able to pause recording for a moment to say, and then resume the same recording, rather than stopping recording completely and starting a new recording—the latter giving us two recordings for the same event. It's part of the DCR Standards.

Multi-Channel Audio Volume Control in Playback

To do a transcript, the transcriber needs to adjust the volume levels in individual channels—even muting all channels except the one they're listening to. **Note:** The current Flash sound model for the FlvPlayback component mixes down to two channels, which means that level of control may not be possible with a Flash player on a four-channel audio file. We may need to go to an ActiveX control for playback (at least for transcriptionists), or some other workaround; worst case scenario: desktop client software for transcribers—leaving a web-based player without that level of control for everyone else.

Expected Features – Unimplemented

These are features I expect to be present in OpenCourt v1.0, but are currently unimplemented. They aren't required by the State DCR standards, and most of them are pretty technical in nature.

Native Flash/iPod compatible MPEG-4 files at Creation

I'm hopeful that with the right header parameters, no repackaging step will be needed. You may still want to do a “hints” run after the recording stops to speed playback buffering.

Start/Stop Recording Control via Keypad/Keyboard on Recording Station

After spending quite a bit of time making sure the recording system could be controlled remotely, a judge asked me if they could turn it on and off themselves from the room. Since the recording station sitting in the corner of the room is “headless” (has no monitor, or keyboard, or mouse, for that matter), I was going to use the DirectInput routines to take control of the keyboard, and a notebook numeric keypad to act as a control panel (retitle 0 and 1 as “Start” and “Stop”, perhaps.)

Custom Recording Program

I can't find Open Source software that does four-channel audio from one source and video from another, so I need to either heavily modify an existing package or create my own. I'm currently leaning toward creating my own, using existing Open Source libraries.

E-mail Alerts on System Problems/Failures

Ideally, the tech support staff should know about a problem before the users do. Have a recording station e-mail a message if it can't reach the archive server. Have the archive server send an e-mail if it can't reach the recording station.

Foot Pedal Support in Web Player

This is for transcribers; the USB foot pedals with the Pedable software I found online look like keyboards, so you can handle their input as keyboard events in the Flash player.

Recording Support for AXIS cameras with ASIO or DirectShow Audio

With VLC, I can't record from an AXIS camera at all. With another program, Mencoder, I can record from an AXIS camera, but I can't take audio from DirectShow (standard Microsoft Sound API) or Steinberg's ASIO (the special Sound API used by multi-channel gear). I need to be able to support AXIS cameras for something other than Slow-Scan preview.

OpenCourtLauncher as a Tray Application

We're currently running as a Windows Console App, which is crude. Having a little OpenCourt icon in the System Tray is the Way to Go.

Standalone Frontend Displays Video and Audio Status before Recording

I want to add video preview and audio indicators so that you can tell that the equipment is working before you start reporting. Audio lights for signal presence: dark green, low signal, light green, high signal (a peak), red, no signal at all.

Linux & Windows Support on the Front-End

In polling the other Court Technology Officers in Florida, there's interest to run the Front-End on either Windows or Linux. That may involve running Apache on the Windows versions.

Wish List

These are features I'd like to implement, if I had the time & money.

Start/Stop Control via Voice/Motion Activation

If the systems just record everything they pick up in a courtroom, we don't have to worry about somebody turning the recording on and off--the event summaries and notes can be added later. I imagine a "solid X minutes of silence" criteria to stop recording, and a "slammed door filter" to keep it from recording just because of one sharp noise. In this scenario, a courtroom's mixer control panel doubles as a recording control panel—if the mixer is in "system on" mode, and there's anyone in the room talking, we're recording.

Web Interface: Filtered views by Judge/Event

If you're just interested in seeing Judge Smith events, or First Appearances, you should be able to filter the calendar to show them.

Hearing Room Privacy Settings

Once court reporters can pull up a slow-scan view of a judge's hearing room, I expect that some judges will object. We may want to lock down certain venues so only the judge or JA can control them, or look at previews.

Slow-Scan Preview of Mic Levels

I'd like the Slow-Scan images to have recording lights below them that could verify whether sound is being picked up by the microphones on each channel, a la the Standalone Fronted Expected Feature. That way if there's an audio problem, the court monitor will be aware of it before they even start recording.

Backward Skip of Real-Time Viewing

Most streaming video clients won't let you skip backwards through a streaming source. Most file-playing clients don't support reading a file as it's being written. We'll need to create one or the other to get around this limitation.

Slow-Vid button on monitor client to handle low-bandwidth connections

384kbps isn't much, but it is if you're connected via a 56k modem; how about the option of giving us a frame every 5 seconds or so?

ALD support (via Sennheiser IR emitters)

Can we add these inexpensively to hearing rooms with the USB Mixers we have? We don't have ALD capability in the hearing rooms, except when we bring in portable units.

Time/Pitch Scaling on Playback for transcribers

A neat feature—this lets you slow down the recording without changing the pitch...people just seem to talk (and move) more slowly. There's some open source software that can do this for audio; just drop the frame rate down for video; we're talking ActiveX control at least to add this feature.

BlackBerry Support for Video

Once I showed people playback via iPod, they asked about Blackberries; the answer is currently "No".

Intercom (monitoring PC to room recording station PC audio)

Let the remote court monitor interact with the courtroom ("Your honor, could you please announce the case # and name for the record?") Headset mic & speaker on one end; it works in Half-Life & World of Warcraft...

Pan/Tilt/Zoom or capable cameras via monitor interface

Even some USB webcams support PTZ now.

Mixer Remote Control (Mute/unmute, playback on/off)

Sometimes people forget to press the buttons on the mixer control panel. It would be nice to be able to do it via the OpenCourt web interface. Difficult to achieve with our current mixers; Peavey Nions would be capable, however.

Support Multiple Video Sources, with "Software Dual/Quad"

Instead video recorded from one webcam, record from two (or more). I've tested with three webcams on a single PC, and it worked...

User Permissions to Specific Events (with Expirations), Judge Presiding Permissions

Current permissions don't let you grant user X access to see event Y at venue Z—user X can either see venue Z, or he can't. Setting an expiration would mean user X could see it until some date, at which point that access would go away.

On the other hand, if a user's a judge, they should be able to see any court event they presided at. (And so should their JA.)

Integration with STAC, ICMS for Auto Time-Stamping Event Notes

If a judge in court pulls up a case in ICMS (our case management system) at 11:02am, or a State Attorney in court pulls up a case in STAC (their case management system) at 11:02am, we could create an 11:02am event note with that case # on the recording. That way, you could later find the exact section of the recording that refers to that case. This would allow case time-indexing of a recording **with no courtroom monitors**, i.e. no one listening to the recording and manually adding case #s as they're mentioned in court.

Cobranet Support

The Peavey Nion mixers support this, among other brands—Cobranet is a way of sending audio data over a LAN. You could connect a recording station to a given venue over Cobranet, allowing you to have remote stations connect to a venue without running analog wires, and allowing you to set up fewer stations, since all venues aren't in use at once.

Dictaphone Emulation

An OpenCourt setup is pretty much 9/10th of a Dictaphone setup. Just omit the video and put a control app on the judge's desktop PC?

Point-to-Point H.323 Support (with Separate Audio Channels)

A netbook can run OpenCourt; a netbook can run the PolyCom PVX software. But a netbook can't run both at the same time, so you can't record a videoconference with OpenCourt.

If we could integrate the code of the OpenMCU project with OpenCourt, we could videoconference and even record both sides (via the Software Dual Quad feature mentioned above). Better yet, we could connect to an MCU and record the entire thing (much cheaper than a \$15,000 Polycom RSS 2000). Keeping the audio channels separate would keep us in line of our DCR standards.

Multipoint H.323 Support (with Separate Audio Channels)

If we can do point-to-point, how about Multipoint? Faster PC required

Skype or ooVoo Integration

Skype is point-to-point, but ooVoo is multi-point (and the pay versions already have recording capabilities...)

Wake-on-LAN forwarding from server to server

Even if your WAN doesn't forward Wake-on-LAN packets, the archive servers could, so a wake command from one web frontend to tell an archive server on a different LAN to wake up the recording station it asked for.

Upload of Phillips 9600 Portable Digital Recorder Files

We're using these for some events (phone conferences). It would be nice to upload them to OpenCourt so they could be indexed with everything else. Would need to transcode them to something Flash could play. (They're just audio, though...)

Upload of Streaming Audio from SunCom Conferencing Site

Here again, another source of audio we'd like to assimilate into OpenCourt.

Editing, Redacting, and DVD ISO File Creation via Web Client

Currently, any video editing is handled via third-party video editing software; if we can build that into a flash client, it's a bit nicer. "Click here to download disk image..."

Current Development Work & Code Status

Here's where things stand, and what I have in place.

OpenCourt Launcher

The launcher is currently win32 console app as mentioned above; it works, but it's crude (launching the VLC Media Player & repackager via .BAT files).

OpenCourt Standalone Launcher

This is a Windows Forms C++ app, with a very similar look and feel to the Record dialog box in OpenCourt. It's not up-to-date. It's designed to launch an external player, but doesn't have any repackaging provisions.

Web Front-End

A small amount of PHP code, with some data being stored in a PostgreSQL database, running Apache on a Linux box. No reason this couldn't be a virtual machine, and I'm indeed looking in that direction.

Flash Player

I developed the OpenCourt-Player.swf use Adobe Flash CS3 Professional; I have CS4 but haven't ported it yet. I'm not aware of any CS4 enhancements that would affect development.

Recording Software - Platform

I'm running Microsoft Windows XP on the Recording Stations due to the difficulty of finding device drivers in Linux for the audio hardware I'm using.

Since I'm running Windows, I'm developing software for the Recording Stations using Visual Studio. I'm working in C/C++ because many of the libraries I'm using are written in those languages, and I'm already familiar with C & C++. I'm using a lot of Open Source libraries, but have not hit any significant roadblocks getting them to compile under Visual C/C++ (although in reality as long as I can link, I'm still OK...) Also, DirectShow apps must be written in C++, there's no managed API for Directshow. (according to [http://msdn.microsoft.com/en-us/library/ms786508\(VS.85\).aspx](http://msdn.microsoft.com/en-us/library/ms786508(VS.85).aspx)).

Recording Software – Origins & Derivations

I thought about extending the VLC Media player for this project, but it does many things I don't need to do, and some things I want to do in inconvenient ways.

Another possibility for expansion might be Mencoder; it has RTSP support and creates Flash-compatible files; a lot less code there too. I experimented with it when I was looking at AXIS network cameras heavily; it had no DirectSound (or, unsurprisingly, ASIO support) so it didn't implement enough to get started.

My current thinking is that we want to build at the library level and make our own client.

Multi-Channel Audio (i.e., more than two channels)

The Windows DirectSound API doesn't support more than two channels of audio. Every single vendor of audio hardware I've looked at supports the Steinberg ASIO API. Sadly, the SDK for this API has not been released under the GPL, which means I can't freely distribute compiled versions of anything I build with the SDK. The Audacity project has the same limitation. We'll just distribute the source code that will compile once you have downloaded the SDK yourself...also what Audacity does...

I've written a simple C++ program that can connect to an M-Audio Fast Track Ultra and write the raw sound data for four channels at once to separate files. I've then successfully repackaged those via external tools into an audio file that I've played back. The file's a

heavily modified “hostsample” from the ASIO SDK; it’s located in my G:\Visual Studio 2008\Projects\ASIOSDK2\host\sample\. I was trying to merge in x264, with the idea of getting everything under one roof. Should have worked on FAAC and MP4v2, though, to hit the appropriate milestones (see below).

Video

As I’ve been concentrating on Audio, I haven’t done much on video; I have the WinCap demo source code and have been analyzing that. I also looked at a sample of the X264 app to see how that library is used.

Libraries

Here are the libraries I am investigating and expect to use...

ASIO SDK 2.2 (Steinberg)	API to multi-channel audio (not Open Source, but no cost)
X264	Open Source implementation of H.264 encoder/decoder
FAAC/FAAD	Open Source MPEG4-AAC encoder/decoder
Mp4v2	Open Source MPEG-4 Library

Development Milestones

Here is how I currently see development proceeding:

- 0. Write a program to grab a frame from a WebCam and return it as a JPEG image via HTTP.**
1. Integrate #1 with the existing OpenCourt Launcher to provide Slow-Scan preview for WebCams...
2. Record 4-channel audio from an ASIO source to a Wave format file.
3. Record 4-channel audio from an ASIO source to an HE-AAC MPEG-4 file.
4. Record 4-channel audio from an ASIO or DirectSound source to an HE-AAC MPEG-4 file.
5. Record video from a webcam to an AVI file.
6. Record video from a webcam to an H.264 MPEG-4 file.
7. Record video from an Axis Network Camera to an H.264 MPEG-4 file.
8. Record 4-channel audio from ASIO or DirectSound and video from a DirectShow or Axis Network Camera to an H.264/HE-AAC MPEG-4 file.
9. Record 4-channel audio from ASIO or DirectSound and video from a DirectShow or Axis Network Camera to an H.264/HE-AAC MPEG-4 file that is compatible with Flash & QuickTime without repackaging.
10. Stream the H.264/HE-AAC MPEG-4 file to a web client for live viewing (preferably from the Web Front-End Server) OR Create a Flash Player or ActiveX Control that can view a file as it’s being written
11. Add to the live viewer the ability to create Event Notes while live-viewing.
12. Display created Event Notes during playback, and allow seek to a specific Event Note.

And on from there until the entire feature list is taken care of...

Questions? Comments? Please e-mail me at opencourt@circuit8.org...