Software compatibility and human interface for DV over IP

Tsuyoshi Hisamatsu
(ringo@sfc.wide.ad.jp)
Keio University
Targets of this research

- Providing high quality video communication tool for MacOSX
- Providing user friendly interface
Available video communication tools

- DVTS
- Netmeeting
- CUSeeMe
- etc.
Why DVTS?

- Transports TV quality video
- Uses consumer AV devices
- Open source
  - Written in C language
What is DVTS? 1/2

- **Digital Video Transport System**
  - Transmitting DV streams from a connected Firewire devices
    - RTP (Real-time Transport Protocol), RTCP (RTP Control Protocol)
  - Supported platforms
    - BSD, Linux, MacOSX, Windows
What is DVTS? 2/2

- Uses consumer AV devices
  - DV cameras
  - Digital-Analog media converters
  - etc.
The former DVTS 1/2

- Based on CUI (Character User Interface)
  - Using CUI to install
    - Ex. > Running “make”
    - Users must apply a patch to their mother OS and recompile its kernel
  - Using CUI to Configure
The former DVTS 2/2

- Lacks the functions to playback video streams on the screen
Machine & OS characteristics

- Each OS has different interface to interact with an application
  - Developpers should implement an Application based on culture of Mother OS
    - UNIX(Linux, *BSD) : CUI with keyboard
    - MacOS: GUI with mouse
GUI Requirements for DVTS on MacOSX

Installation
- With simple “Drag & Drop” install wizard

Running application
- With simple mouse and keyboard operation
Design of DVTS on MacOSX

**Using DVTS without difficulty**
- Implementation based on GUI
  - To control DVTS only with the mouse operation except when inserting IP address

**Video playback window**
- To monitor video on the screen
Design outline of DVTS on MacOSX (sender) 1/4

- User connects FireWire devices to a sender machine
Design outline of DVTS on MacOSX (sender) 2/4

- Configure application by adding an option to the sender
Design outline of DVTS on MacOSX (sender)3/4

- The sender reads DV from the connected FireWire device.
Design outline of DVTS on MacOSX (sender)4/4

- The sender adds IP, UDP and RTP header to the data
Design outline (receiver)

- User sets options on the receiver system
- Select output device, the connected FireWire device or the viewer
Tools for implementing new DVTS 1/2

Project Builder
Tools for implementing new DVTS 2/2

- Interface Builder
### Implementation (interface) 1/4

<table>
<thead>
<tr>
<th><strong>Sender</strong></th>
<th><strong>Receiver</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>✷ Start sending packet</td>
<td>✷ Start receiving packet</td>
</tr>
<tr>
<td>✷ Quit Application Loop</td>
<td>✷ Quit Application Loop</td>
</tr>
<tr>
<td>✷ Input IP Address of receiver</td>
<td>✷ Start drawing received movie (dvshow)</td>
</tr>
<tr>
<td>✷ Send using IPv4 or IPv6</td>
<td>✷ Join multicast</td>
</tr>
<tr>
<td>✷ Use default or custom Port Number</td>
<td>✷ Receive using IPv4 or IPv6</td>
</tr>
<tr>
<td></td>
<td>✷ Use default or custom Port Number</td>
</tr>
</tbody>
</table>

**dvsend**

- **send**
- **stop sending**
- **Destination Address**
  - IPv4
  - IPv6
- **Multicast**
- **default Port**
  - **Custom Port**
  - 8000
- **packet loss**
- **receivers packet loss**

**dvrecv**

- **dvrecv**
- **Quit**
- **dvshow**

- **use RTCP**
- **join multicast**
- **use default Port Number**
- **use custom Port Number**
- **use default Port for Audio**
- **use custom Port for Audio**
- **packet loss**
Implementation (controller) 2/4

- Carbon
  - C programming interface for MacOSX
  - Call-back mechanism
  - Usable the technology of MacOSX
    - Memory protection
    - Preemptive multitasking
    - Dynamic resource assignment
    - etc.

Call-back mechanism

Buttons, CheckBoxes

Event

(Mouse Click, etc.)

GetEventParameter()

Carbon

Event Manager

CommandHandler()

Determination of the configuration

Click the start button

main loop
Implementation
(controller on sender)3/3

Start Application & Send the DV stream

The former DVTS
(not using Carbon
Thread Manager)

Enter the command

Read & Send loop

Press “Send” Command

The New DVTS
(using Carbon
Thread Manager)

Configuration

YieldToAnyThread()

Window thread

Read thread

Send thread

cooperatorive thread
Implementation (controller on receiver) 4/4

Start Application

The former DVTS (not using Carbon Thread Manager)

Enter the command

The New DVTS (using Carbon Thread Manager)

Send the DV stream

fork()

Read loop

Recv loop

Press “Recv” or “View” Command

Configuration

YieldToAnyThread()

cooerperative thread

Window thread

Recv thread

Write thread

Viewer thread
Implementation (video playback window)

- QuickTime framework
  - Movie Toolbox
- Integration receiver and viewer
  - The output of the DV stream will be redirected from FireWire devices to the viewer window
### Comparing the former DVTS to this system

<table>
<thead>
<tr>
<th>Steps</th>
<th>The former DVTS</th>
<th>This system</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Downloads the source code from the site</td>
<td>Downloads the diskimage from the site</td>
</tr>
<tr>
<td>2</td>
<td>Uncompresses the archive and extract package manually</td>
<td>Click diskimage file stored in the directory</td>
</tr>
<tr>
<td>3</td>
<td>Enter “./configure” in the extracted directory</td>
<td>Drag the system from “disk image” and drop it to the hard disc</td>
</tr>
<tr>
<td>4</td>
<td>Enter “make” to construct a binary</td>
<td>Click the application to run the sender</td>
</tr>
<tr>
<td>5</td>
<td>“make install” to install the application to the binary directory</td>
<td>Click the application to start receiving</td>
</tr>
<tr>
<td>6</td>
<td>“dvrecv -h” to see the command line options</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>The user manually adds the option to the command, for example “dvrecv -4LR”</td>
<td></td>
</tr>
</tbody>
</table>
Conclusion

- Provides standard MacOSX interface
  - Concealing Unix command line input method
  - Easy-to-use high quality video transport system
- Monitoring mechanism of the video image
Impression of porting DVTS

- Porting from UNIX ware (written in C language) to MacOSX is relatively simple
  - However
    - Unsatisfied documents on Carbon
    - Distinctive threads when using GUI
Release

- Still implementing the viewer
- Still debugging others

- Scheduled for release on February
  http://www.sfc.wide.ad.jp/DVTS