Internet Technologies for Multimedia Applications

Part-II

Multimedia on the Internet

Lecturer: Dr. Daniel Pak-Kong LUN
Room: DE637  Tel: 27666255
E-Mail: enpklun@polyu.edu.hk
Contents

- Review: Multimedia on the Internet
- Multimedia Information Storage and Management
- Error Resilience in Multimedia Communications
- Scalable Multimedia Communications
- Intelligent Proxy Systems for Multimedia Streaming
A simplified model for Internet multimedia system
Current Multimedia Services on the Internet

Soul of all these services

Media Streaming

Movie broadcasting

Live broadcasting

Internet slice show

MP3 broadcasting

Video conferencing

Web TV / Education

Web multimedia advertisement
What is Media Streaming?

Performing multimedia application while receiving the data
Normal Internet Service

Streaming

Server

Client

Server

Client
Traditional Internet Service Model

- **Heavy Weight** – because using TCP/IP
- **Best Effort** – introduce delay due to retransmission and queuing in the routers
- **Stateless** – Web servers try to finish the job as soon as possible
Characteristics of Streaming Applications

- Multimedia services are **time-dependent and synchronized**
  - Need a new kind of scripting language
  - Need a new application level protocol for sending multimedia data

- Multimedia files are **huge in size**
  - Need efficient data storage and retrieval mechanism in order to serve multiple clients at the same time

- Multimedia data are **bandwidth-demanded**
  - Cannot use heavyweight transport protocol
  - Need compression
- Multimedia data have **high correlation**
  - Best-effort delivery is not desirable
  - Need error resilience

- Multimedia data access has **locality effect**
  - Can use proxy to reduce remote data access

- Multimedia services may perform across **different networks and over different kind of clients**
  - Cannot be stateless
  - Require scalable communications
Media Streaming Using RTP and RTCP on the Web

Media Server -> RTP Packet -> Media Player

RTP Packet

- Header, Seq.no.
- Timestamp
- Source ID
- Payload data

RTCP Control

Media Data

Media Player

DEPARTMENT OF ELECTRONIC AND INFORMATION ENGINEERING
Internet Technology for Multimedia Applications
SMIL and RTSP: New Scripting Language and Application Layer Protocol

Media Server

rtsp://realserver.company.com

SMIL over HTTP or RTSP

RTSP Remote Control

Media Player
Multimedia Compression

- Video file
  - Video Encoder: MPEG-1, MPEG-2
  - Image Encoder: JPEG, JPEG2000
  - Speech and Audio Encoder: PCM, G.723, GSM, MP3, Atrac3
  - Real-time Encoder: H.263, MPEG-4

- Media Storage

- Network
Multimedia Storage and Retrieval
Error Resilience in Multimedia Communications
Scalable Multimedia Communications

Media Server

Narrow band

Mobile Internet Client

Broad Band

Narrow Band

Internet Clients
Multimedia Streaming Proxy

Media Server in USA

Proxy Server

Cache

HK

Media Player

Media Player

Media Player
References

Description and tutorial on SMIL
- SMIL specification, http://www.w3.org/AudioVideo/
- Learn SMIL with a SMIL presentation,
  http://www.empirenet.com/~joseram/index.html

Description and information of RTSP
- ftp://ftp.isi.edu/in-notes/rfc2326.txt
1. New Scripting Language - SMIL (Synchronous Multimedia Integrated Language)
Background

- SMIL stands for **Synchronous Multimedia Integrated Language**
- Develop and standardize by **World Wide Web Consortium** (W3C)
- W3C is an industry consortium
- Seeks to promote standards for the evolution of the Web and interoperability between WWW products by producing specifications and reference software
- Jointly hosted by the **MIT Laboratory** for Computer Science (U.S.) and **INRIA** (Europe)
- Initially established in collaboration with **CERN**, where the Web originated, and with support from **DARPA & EC**
Current Status

- First public draft of SMIL was released in November, 1997
- SMIL 1.0 specification was announced in June 1998
- SMIL 2.0 specification has become a working draft in July 2000
- Many browsers and media players support the display and authoring of SMIL file - QuickTime 4.1, RealPlayer 7.0, Internet Explorer 5.5 (partly), RealSlideShow 2.0, etc.
What can be done with SMIL?

- Allow real-time synchronization of multimedia streams
- Position media elements wherever you want on the screen
- Provide interactivity between user and content provider
- Display media follows user-preferences, language, bit-rate, etc...
An Example

EIE552 Internet Technologies for Multimedia Applications

Example 4

Inter-media Synchronization using SMIL (Introducing Hong Kong)

Hong Kong
- Our Home

A multimedia clip to introduce Hong Kong
• Similar to HTML, SMIL files are sent to the client before the actual execution takes place
• Different from HTML, containing timing information
• Can be considered as a **schedule** of multimedia playback

**Media Server**
rtsp://realsever.
company.com

**SMIL over HTTP or RTSP**

**RTSP Remote Control**

**Media Player**
How to write a SMIL Document?

• What you need to create a SMIL document is a simple text editor!

• SMIL is XML-based and is very similar with HTML and this makes the language easy to read and understand

• There are however differences between SMIL and HTML
  – SMIL is case-sensitive. All tags must be written in lower case.
  – SMIL is XML-based. Tags have to be ended.
A. Creating a SMIL File

• In its simplest form, a SMIL file lists multiple media clips played in sequence

```xml
<smil>
  <body>
    <audio src="rtsp://abc.company.com/one.rm"/>
    <audio src="rtsp://abc.company.com/two.rm"/>
    <audio src="rtsp://abc.company.com/three.rm"/>
  </body>
</smil>
```
B. To Play a SMIL File

- To play a SMIL file, one needs to have a **SMIL enabled player**, such as, RealPlayer 7.0, QuickTime 4.1, etc.
- To play a SMIL file on the Web using RealSystem G2, one can directly put down the path of the SMIL file on the Web page and play as a normal RealMedia file

```html
```
C. Interesting features

C.1 Grouping Clips

- Clips can be grouped in one of the two ways:
  - `<seq> ... </seq>` - clips play one-by-one in sequence (default)
  - `<par> ... </par>` - all clips play together in parallel (be sure not exceed the bandwidth)
With PAR

Without PAR, i.e. SEQ
• Different clip can place in different region

```xml
<smil>
  <head>
    <layout>
      <root-layout width="460" height="300"/>
      <region id="region1" top="20" left="70"
               width="320" height="240"/>
      <region id="region2" top="260" left="70"
               width="320" height="20"/>
    </layout>
  </head>
  <body>
    <par>
      <img src="Convention.jpg" region="region1"/>
      <text src="Location.txt" region="region2"/>
    </par>
  </body>
</smil>
```
C.3 Specifying Timing

- **SMIL timing** elements let one specify when a clip or group starts playing and how long it plays.
- It is optional. If not set, follow the normal timelines and their positions within `<par>` and `<seq>` groups.
- Time is designated with shorthand markers ‘h’, ‘min’ and ‘s’:

  - `2.5h` \(\Rightarrow\) 2 hours, 30 minutes
  - `2.75min` \(\Rightarrow\) 2 minutes, 45 seconds
  - `15.55s` \(\Rightarrow\) 15 seconds, 550 milliseconds
  - `670.2ms` \(\Rightarrow\) 670.2 milliseconds
  - `2s = 2.0s` \(\Rightarrow\) 2 seconds
  - `2:34 = 2:34.0` \(\Rightarrow\) 2 minutes 34 seconds
Example

```<audio src="warm_poem.ra" begin="5s"
clip-begin="15s" clip-end="30s"/>
```

- All timing attributes apply only to streaming data format, hence .rm and .ra work but .mp3 does not work.
- `clip-begin` and `clip-end` only apply to clips that have internal timelines, such as audio, video but not image, static graphic.
C.4 Switching Between Alternate Choices

• The `<switch>` tag allow one to specify multiple options that the player can choose between

  `<switch>`
  
  `<choice1 test-attribute="value1"/>`
  
  `<choice2 test-attribute="value2"/>`
  
  `</switch>`

• The **test-attribute value** is embedded in the machine where the player is installed

• Based on the value of the test-attribute, either choice1 or choice2 is selected

• For **RealPlayer**, there are two things that one can choose: **language** and **bandwidth**
<switch>
  <par system-bitrate="75000">
    <video src="video1.rm"/>
    <audio src="song1.rm"/>
  </par>

  <par system-bitrate="47000">
    <video src="video2.rm"/>
    <audio src="song2.rm"/>
  </par>

  <par system-bitrate="20000">
    <video src="video3.rm"/>
    <audio src="song3.rm"/>
  </par>
</switch>
<switch>
  <video src="french/video1.rm" system-bitrate="47000"
         system-language="fr" />
  <audio src="french/song1.rm" system-bitrate="47000"
         system-language="fr" />
  <video src="english/video2.rm" system-bitrate="20000"
         />
  <audio src="english/song2.rm" system-bitrate="20000"
         />
</switch>

- **test-attributes** can put together to form a logical **AND** operation
- For this example, if not French and ISDN, English version will play if modem connection is available
C.5 Linking to Other Media

- As HTML, a SMIL file can define links to other media using `<a>` tag or `<anchor>` tag

```xml
<smil>
  <body>
    <a href="file://Harbcru.rm">
      <video src="HKIsland.rm" title="Hong Kong Island Tour"/>
    </a>
  </body>
</smil>
```

Nevertheless, RealPlayer 8.0 does not support `<a>` tag!

- In this example, Harbcru.rm will play instead of HKIsland.rm when user clicks on the image window of RealPlayer
C.6 Interactivity by image

```html
<image
    src="HongkongIsland.gif?url=file://HKIsland.rm&target=_player"
    region="ads"/>

<par endsync="last">
    <video src="HKIsland.rm" region="video"/>
    <img src="Goto17_4.gif?url=command:seek(17.4)
    &target=_player" region="image"/>
</par>
```

Other commands: play(), pause() and stop()
2. RTSP - Real Time Streaming Protocol
A. Background

- RTSP stands for **Real Time Streaming Protocol**
- An application-level protocol enables **control of on-demand delivery of real-time data**, e.g. audio, video
- Include both live data feeds and stored clips
- First draft of protocol specification, RTSP 1.0 submitted to IETF (Internet Engineering Task Force) on Oct 1996
- Submitted by RealNetworks and Netscape
- Became **“Proposed Standard”** on June 1998 - means still require further improvement
- Support by more than 40 companies, e.g. Apple, Cisco, HP, Sun, Silicon Graphic ...
B. RTSP Features and Functionality

- Provide synchronization between media streams
- Virtual presentation = synchronized playback from several servers
- Load balancing using redirection at connect, during stream
- Supports any session description
- Device control, including camera pan, zoom, tilt …
RTSP and HTTP: Similarities

- RTSP is intentionally similar to HTTP
- Extension to HTTP in most case can also be applied to RTSP
- RTSP has the same format as HTTP: text based, MIME-header, etc.
- Request / response mechanism is similar
  - request line + headers + body
- Use a similar set of status codes and reason phases,
  - 404 Not Found, 200 OK, 301 Moved Permanently ...
  - Some new ones: 453 Not Enough Bandwidth, 461 Unsupported transport ...
- Security mechanism, URL format, Content negotiation
RTSP and HTTP: Differences

- RTSP introduces a number of **new methods** and has a **different protocol identifier**
- RTSP server needs to maintain **state** as opposed to the stateless nature of HTTP
- Both an RTSP server and client can issue requests
- With RTSP, media data can be carried **out-of-band** by a different protocol (e.g. RTP, RDT)
- RTSP uses ISO 10646 (UTF-8) rather than ISO 8859-1, consistent with current HTML internationalization effort
- RTSP Request-URI always contains the **absolute URI** rather than separately in the request and header as in HTTP
Connections between Server and Client

- As an application layer protocol, **RTSP** does not restrict the underlying **transport** protocol.
- The media data packets can be transported using multicast UDP, unicast UDP or inline TCP (data interleaved with the RTSP control stream).

![Diagram of RTSP connections]

- **RTSP Player**: Port 2n, Port 2n+1
- **RTSP Server**: Server Port (default 554), Port 2m, Port 2m+1
- **TCP-Control Connection**
- **RTP Data**
- **RTCP Reports**
- RealNetwork has its own transport layer protocol - RDT

**Diagram:**
- **RTSP Player**
- **TCP-Control Connection**
- **RTSP Server**
  - Server Port (default 554)
- **RDT/UDP Data**
- **RDT/UDP Resend Requests**
- Media data may be made into packets using RTP or RDT over TCP
- A single full-duplex TCP connection is used for both control and for media data delivery
- Data stream is interleaved with the RTSP control stream
Internet Technology for Multimedia Applications

RTSP Operation

- **SETUP**: create transport layer link
- **PLAY**: request server to deliver media data
- **RTP data and RTCP control reports**
- **PAUSE**: stop until further notice
- **TEARDOWN**: release resources
Protocol Semantics

- Unlike HTTP, RTSP has a certain amount of state associated with it.
C. RTSP Messages

- Share a similar syntax to HTTP messages
  
  `{method name} {URL} {protocol version}CRLF
  {parameter}`

```
c->s
  DESCRIBE rtsp://foo.com/bar.rm RTSP/1.0
  CSeq: 892
  Accept: application/sdp, application/mheg
```

Ask the server to send a description of the media content rtsp://foo.com/bar.rm using either Session Description Protocol (SDP) or Multimedia and Hypermedia Expert Group (MHEG) formats
A message may also contain a body

{method name} {URL} {protocol version}CRLF
{MIME header field}CRLF
...
{MIME header field}CRLF
CRLF
{optional body, length defined in “content-length”}

\textbf{ANNOUNCE} rtsp://foo.com/bar.rm RTSP/1.0
CSeq: 892
Date: 9 Sep 1998 13:00:00 GMT
Session: 45991232
Content-Type: application/sdp
Content-Length: 332

\textbf{c->s}

Describe the media content to server using 332 bytes of SDP
- Each RTSP request is followed by a response message
  
  `{protocol version} {status code} {reason}CRLF
  {parameters}`

- \text{c->s: PAUSE rtsp://example.com/baz.rm RTSP/1.0}
  \text{CSeq: 834}
  \text{session: 12345678}

- \text{s->c: RTSP/1.0 200 OK}
  \text{CSeq: 834}
  \text{Date: 23 Jan 1997 15:35:06 GMT}
D. RTSP Methods

- **OPTIONS** get available methods
- **SETUP** establish transport
- **ANNOUNCE** change description of media object
- **DESCRIBE** get (low-level) description of media object
- **PLAY** start playback, reposition
- **RECORD** start recording
- **REDIRECT** redirect client to new server
- **PAUSE** halt delivery, but keep state
- **SET_PARAMETER** device or encoding control
- **TEARDOWN** remove state
Example: PLAY

- **Tell server to start sending data** via the mechanism specified in SETUP
- **Must not** issue PLAY request if SETUP not successful
- Default: play from the beginning - normal play time (npt)
- Can specify the start time
- Multiple PLAY requests can be queued in the pipeline
Internet Technology for Multimedia Applications

Relative to the beginning of the presentation

```plaintext
c->s: PLAY rtsp://example.com/baz.rm RTSP/1.0
    CSeq: 835
    session: 12345678
    Range: npt=10-15

C->s: PLAY rtsp://example.com/baz.rm RTSP/1.0
    CSeq: 836
    session: 12345678
    Range: npt=20-25

C->s: PLAY rtsp://example.com/baz.rm RTSP/1.0
    CSeq: 837
    session: 12345678
    Range: npt=30-
```
- Range header may contain an **absolute time parameter**
- If message is received after the specified time, start immediately
- For on-demand system, **server replies the actual range**
- **May differ** from the request range

```plaintext
CSeq: 833
session: 12345678
Range: smpte=0:10:20-;time=19970123T153600Z

s->c: RTSP/1.0 200 OK
CSeq: 833
Date: 23 Jan 1997 15:35:06 GMT
Range: smpte=0:10:22-;time=19970123T153600Z
```

**Start time relative to the beginning of the Clip**

**Playback at 15:36:00 on 23 Jan 1997**
E. Example - Single Stream Playback

\[
\begin{align*}
c->s: \text{ OPTIONS} & \quad \text{rtsp://foo.com:554 RTSP/1.0} \\
& \quad \text{CSeq: 1}
\end{align*}
\]

\[
\begin{align*}
s->c: \quad \text{RTSP/1.0 200 OK} \\
& \quad \text{CSeq: 1}
\end{align*}
\]

Identify the protocol version. Server will further list the available methods supported and other options.

\[
\begin{align*}
c->s: \text{ DESCRIBE} & \quad \text{rtsp://foo.com:554/single.rm RTSP/1.0} \\
& \quad \text{CSeq: 2}
\end{align*}
\]

\[
\begin{align*}
s->c: \quad \text{RTSP/1.0 200 OK} \\
& \quad \text{CSeq: 2} \\
& \quad \text{Content-length: 197}
\end{align*}
\]

Describe the media content with 197 bytes of SDP (default).
c->s:  SETUP rtsp://foo.com:554/single.rm RTSP/1.0
     CSeq: 3
     Transport: x-real-rdt/udp;client_port=6970,
                rtp/avp;unicast;client_port=6970-6971
s->c:  RTSP/1.0 200 OK
     CSeq: 3
     Session: 968367008
     Transport: x-real-rdt/udp;client_port=6970;
                 server_port=7970

Setup a RealNetwork’s RDT channel for later media playback
c->s: **PLAY** rtsp://foo.com:554/single.rm RTSP/1.0
   CSeq: 4
   Session: 968367008
s->c: RTSP/1.0 200 OK
   CSeq: 4

Play the stream immediately with the transport mechanism specified in SETUP response message

---

c->s: **TEARDOWN** rtsp://foo.com:554/single.rm RTSP/1.0
   CSeq: 5
   Session: 968367008
s->c: RTSP/1.0 200 OK
   CSeq: 5

Client requests the server stop sending data referring to this session