Next Generation Session Announcements:

Internet Media Guides (IMGs)

Observations

- SAP/SDP tied to IP-Multicast-based session model
- Only one distribution scheme: announcement
- Only one type of service: convey multimedia session information
- (Global) IP-Multicast has not prevailed as a distribution platform
- SAP rather experimental
- Was often used for debugging Mbone connectivity

Summary

- SAP/SDP too limited
- Not appropriate as a general solution for distributing session information
- Traditionally linked to IP-only (and Multicast-only)
Background: Ubiquitous Information Access

- **Live Broadcast**
- **Studio**
- **Canned Program**
- **Cellular Networks**
- **Broadcasting Networks**
- **Internet + IP Networks**
- **Workstation**
- **Laptop / tablet PC**
- **PDA**
- **Cellphones**
- **TV set / radio**
- **…**
- **…**

“Classic” Broadcasting & Internet Multimedia

- Broadcasting has been a different world
  - (including customer expectations, philosophy)
  - **Encodings**
    - Audio/Video largely compatible (but different quality expectations)
    - **Image/text formats/HTML** vs. **Videotex, MHP, specific markups, tables**
  - **Data transmission**
    - IP + UDP/TCP + RTP/… vs. MPEG multiplex (or even analog)
  - **Addressing**
    - IP addresses + ports vs. frequency/channel, PID, satellite position, pol., …
  - **Interaction & control**
    - RTSP, HTTP, SIP, … vs. MHP

- But there is a migration towards IP in various areas
  - Content providers, transmission technologies, consumer equipment
Platform/Network-Independent Content Provision

- The same content shall be available via different networks
  - Preferably without repeated authoring

- “Content” used in a broad sense
  - Original media: Audio / video broadcasts, web pages, files, news feeds, …
  - Supplementary information: background, statistics, subtitles, ads, …

- Content needs to be globally (or regionally) identifiable
- Content needs to be found
  - Descriptive metadata
  - Availability (scheduling) metadata
- Alternate access methods must be possible
  - Network + network-specific address

Internet Media Guides (IMG)

Definition of an IMG (from MMUSIC Charter)

Content:
- A collection of multimedia session descriptions
- Expressed using SDP, SDPng or other metadata formats
- It is used to describe a collection of multimedia sessions (e.g. television programme schedules).

Distribution:
- The IMG must be delivered to a potentially large audience (push or pull), who use it to join a subset of the sessions described, and who may need to be notified of changes to the IMG.
IMG ≈ EPG

- Generalized for arbitrary...
  - Types of media
  - Types of sessions and interactions: services!
  - Classes of devices
- Plurality of access methods
  - Physical delivery
  - (Reliable) Broadcast / multicast (push)
  - Interactive retrieval (pull)
  - Provision of full IMGs and of deltas
  - Notification about changes
- Network-independent
  - For the delivery of IMGs
  - For the (request and) transmission of actual media in sessions

The same IMGs should be usable everywhere.
IMG Delivery Models / Operations

IMG announcer → IMG ANNOTUCE
Broadcast / Multicast

IMG resolver
Full IMG
IMG QUERY (Pull)

IMG querier

IMG resolver
Full IMG
IMG RESOLVE

IMG querier

IMG notifier
Full IMG
IMG SUBSCRIBE

IMG subscriber

IMG notifier
Full IMG
IMG NOTIFY (w/ content)

IMG subscriber

IMG sender
Full IMG
IMG NOTIFY (w/o content, w/ pointer)

IMG receiver

IMG sender
Full IMG
IMG QUERY

IMG receiver

IMG Architecture

Metadata Formats
#1 #2 ...

Complete Description, Delta Description, Pointer

IMG Data Types
IMG Envelope

IMG Operations
IMG ANNOUNCE
IMG SUBSCRIBE
IMG QUERY

IMG NOTIFY
IMG RESOLVE

IMG Transport
Point-to-Multipoint
Point-to-Point
IMG Envelope: Security Requirements

- Authentication + Integrity validation of contained metadata
  - Must work for complete and delta information
  - Must work across IMG transceivers
    - Aggregation, splitting, filtering of pieces of metadata

- Privacy
  - Must be able to protect (parts of) contained metadata
    - User protection + access control
  - Enable (limited) IMG transceiver functionality

- Interdependency with metadata formats
  - What to expect from metadata?
  - Granularity of embedded metadata objects
  - DRM? metadata formats

IMG Envelope

- Container for metadata
  - Complete, delta, pointers
  - Independent of metadata
  - Likely to become some kind of wrapper mechanism
  - Metadata itself defined by other bodies

- Generic management information
  - Identification + version + validity information
  - Content-Type: to identify metadata format
  - Support for security?
    - authentication + integrity information
    - Privacy of content

MIME vs. XML
Envelope Features (1)

- Container for metadata (independent of these)
  - Complete, delta, pointers
  - Metadata itself defined by other bodies

- Version number
  - Determine the most recent (i.e., valid) copy
  - Referenced as basis for delta encoding

- Validity time
  - Period: from, to

- Metadata URI
  - Identifies the metadata element contained in the envelope
  - Helps to deal with fragments

- Content-Type
  - Defines the type of metadata contents

Envelope Features (2)

- Support for digital signatures (on parts of the envelope)
- Support for encryption
  - Only partly specified so far
  - May use S/MIME

- Metadata contents:
  - Inline
    - Envelope { … }
    - Metadata
  - External (via pointer)
    - Envelope { … }
    - Metadata
Envelope Encoding: XML vs. MIME

- Present focus: XML (also used by 3GPP MBMS)
- Example (with SDP as metadata)

```xml
<?xml version="1.0" encoding="UTF-8"?>
<metadataEnvelope
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:noNamespaceSchemaLocation="envelope.xsd"
    metadataURI="http://www.example.com/img001/session001.sdp"
    version="1"
    validFrom="2003-12-17T09:30:47-05:00"
    validUntil="2003-12-17T09:30:47-05:00"
    contentType="application/sdp">
    <metadataFragment>
        v=0
        o=jo 2890844526 2890842807 IN IP4 10.33.57.27
        s=SDP Seminar
        c=IN IP4 224.2.17.12/127
        t=2873397496 2873404696
        a=recvonly
        m=audio 49170 RTP/AVP 0
        m=video 51372 RTP/AVP 31
    </metadataFragment>
</metadataEnvelope>
```

IMG Metadata

- Past focus on traditional contents
  - Conveying plain TV-schedules
  - Streaming in 3GPP Release 6

- Broadening the scope
  - Cover services in a more general fashion
  - Provide region/location information
  - Support personalized inquiries
  - Address issues of cost
    - Make offers automatically comparable

- Technical level: enable service discovery (and location)
- Business level: support adequate service selection
IMG URN

- IMGs need to be identified globally
  - In particular, across different networks and providers
- Motivates the use of IMG URNs
- Format
  - `urn:img: ProviderId : DateId : IMGResourceId [: FragmentId]`
  - `ProviderId`: domain name
  - `DateId`: Point in time when the domain name was owned by the entity
  - `IMGResourceId`: provider-selected string
  - `FragmentId`: some identifier for a piece of an IMG
- Examples
  - `urn:img:example.org:20051021:my-img`
  - `urn:img:example.org:20051021:my-img:subset`
- Mapping to URIs (e.g., HTTP, SIP) to be defined

IMG Transports

- Need to provide mechanisms for IMG Operations
- ANNOUNCE
  - Reliable multicast transport protocol: FLUTE + MUPPET
- SUBSCRIBE / NOTIFY
  - Session Initiation Protocol (SIP): Extensions for Subscription/Notification
- QUERY / RESOLVE
  - HTTP
- Identify IMGs properly across protocols: IMG URN (yet tbd.)
  - Mappings to individual protocols for actual processing
**IMG ANNOUNCE: Reliable Multicast**

- **Layered Coding Transport (LCT)**
  - Single sender multicast transport
  - Defines single or multi-object delivery across an LCT session
    - Provides identifiers for objects (TOI)
    - Provides session identification (TSI)
  - LCT session comprises a group of channels
    - Each identified by the respective (multicast) transport address

- **Forward Error Correction (FEC)**
  - General container for various FEC schemes
  - Allows to identify payload + provides in-band signaling of FEC parameters

- **Asynchronous Layered Coding (ALC)**
  - Simple combination of LCT and FEC

**IMG ANNOUNCE: FLUTE Basics**

- **File Delivery over Unidirectional Transport**
- **Uses ALC (= LCT + FEC)**
  - Fixed parameter sets for the protocol instantiation

- **Specifies semantics of objects**
  - Files
  - File Delivery Table (FDT)

- **FDT**
  - XML-based format to carry file attributes (name, location, size, etc.)
    - Carried as Transport Object ID = 0
  - Transmitted in a carrousel style together with files
IMG ANNOUNCE: FLUTE FDT

- XML-based structured information

Example
<FDT-Payload Expires="<date>" complete="true">
  <File
    Content-Location=
    TOI=
    Content-Length=
    Transfer-Length=
    Content-Type=
    Content-Encoding=
    Content-MD5=
    ... plus some FEC stuff ... >
  <File ...>
  ...
</FDT-Payload>

IMG ANNOUNCE: MUPPET

- Specific usage of FLUTE for carrying IMG envelopes
- Defines various lower layer parameters
- Defines usage of multiple layers
IMG QUERY / RESOLVE

- "Naturally" maps to HTTP GET + 200 OK
- HTTP URI: http://<hostname>/<resource>?param1&param2&...
  - Parameters identify IMG version
    - type: full or delta IMG, pointer
    - version requested
    - diffVersion: base for delta IMG
- Querier response format selection
  - Accept: application/img-envelope+xml
    - Provide IMG in envelope format
  - Accept: text/plain, text/html
    - Provide a human-readable description of an IMG as optional fallback
  - Allow for directly returning the plain metadata without envelope?
- 200 OK carries response in body
- HTTP headers used accordingly

IMG SUBSCRIBE / NOTIFY

- Based upon the Session Initiation Protocol (SIP)
  - Particularly its SUBSCRIBE / NOTIFY mechanism
  - Details to be discussed
- SUBSCRIBE / NOTIFY
  - Register interest in (part of) an IMG
  - Receive an immediate response and updates upon changes
  - Soft-state based: subscription times out and needs refreshing
- IMG usage of SIP SUBSCRIBE / NOTIFY
  - Define SIP event package: img
  - Presently suggests a MIME-based IMG envelope
    - Natural choice for SIP
  - Content-Type:, Content-Location:
  - Content-ID: major.minor, Expires: valid-until
Regionalization & Personalization with IMGs

TV-EPG Distribution

© 2006 Jörg Ott
IMGs: “Final” Remarks

- In use in 3GPP MBMS
- Stalled in the IETF

TV industry going various other ways
- Specific EPGs in DVB
- TV Anytime forum
- Web/RSS-based program pages of TV magazines and broadcasters
- Open source platforms use yet other formats
- XMLTV

Media Streaming in the Internet

- Introduction to Media Streaming
- Real-time Streaming Protocol (RTSP)
- HTTP-based Streaming
Real-time Media Streaming

Retrieving content from a source where

- the content is continuous in nature (e.g. audio, video),
- the content is (potentially) presented to the user before it has been downloaded entirely, and
- there is no human-to-human interaction involved (i.e. latencies are acceptable to a certain degree).

Contrast: interactive, interpersonal communications

Two Types of Streaming

- Broadcast streaming (non-interactive)
  - Sender transmits media stream according to its own schedule
  - Receivers “tune into a media stream” of interested
  - Receivers have no means to influence the transmission
  - Suitable for multicast / broadcast networks

- Interactive streaming
  - Sender provides media stream to receivers “on demand”
  - Receivers may start / stop transmission
  - Receivers may invoke further operations
    - Fast forward, search, play offset, …
  - Suitable for P2P sessions or coordinated small groups
Architectural Components

- **Content Description**
  - Describe type of content, format, access methods, ...
  - SDP, SDPng, ...

- **Content Description Delivery / Access Protocol**
  - Delivers Content Description
  - HTTP, SMTP, NNTP, SAP, ...

- **Content Access (= Media Streaming) Protocol**
  - Initiates, controls, and terminates media streams
  - RTSP, proprietary protocols, ...

- **Content Delivery (= Media Transport) Protocol**
  - Carries the actual content
  - RTP/RTCP, proprietary protocols, ...

Conceptual Overview

1. Reference to Media Server
2. Content Description

Clients
Real-Time Streaming Protocol (RTSP)

- RFC 2326 (“buggy”, “underspecified”)
- draft-ietf-mmusic-rfc2326bis-13.txt

Interactive streaming control in the Internet
- Media servers provide media streams to users on demand
- Content described by presentation descriptions

“Network Remote Control” of a media server
- PLAY [and RECORD]
- Numerous options for media control
  - PAUSE, faster / slower playback, selection of ranges from a stream, ...
Protocol Characteristics

- Borrows heavily from HTTP
  - Syntax, quite a bit of semantics, parts of the architecture

- Important differences
  - Servers may issue requests, too!
    - Symmetric communication
  - Servers are stateful
  - Different methods
  - Different headers
    - But many HTTP headers re-used
  - Entities (=request/response bodies) only describe content
  - Content itself (=media) is carried out of band
    - e.g. in RTP; also support for interleaving of media with RTSP connection

- Transport: TCP [or UDP]
  - Reliability handled at the RTSP level
RTSP Components

<table>
<thead>
<tr>
<th>Media Server</th>
</tr>
</thead>
<tbody>
<tr>
<td>media-server.tkk.fi</td>
</tr>
</tbody>
</table>

Content Base
/movies/matrix
/movies/matrix/video
/movies/matrix/audio/de
/movies/matrix/audio/en

- Video
- Audio - 1
- Audio - 2

rtsp://media-server.tkk.fi/movies/matrix/audio/en

RTSP URIs

- Schemes:
  - rtsp: reliable, connection-oriented (TCP)
  - rtspu: potentially unreliable, connectionless (UDP)
  - rtsps: secure, reliable, connection-oriented (TLS)

- General scheme:
  - rtsp:// host / local identifier

- Host
  - Should be DNS name
  - Support for IPv4; IPv6 now being added

- Local Identifier
  - Opaque; may be used for aggregate / non-aggregate control
Time in RTSP

- SMPTE Timestamps
  - SMPTE = Society of Motion Picture Television Engineers
  - Measured in hours, minutes, seconds, frames, fractions (subframes)
  - 29.97 or 25 frames per second (default: 29.97)

- Normal Play Time (NPT ≠ NTP)
  - Relative to beginning of stream
  - In seconds: SS.fff 10.74
  - In human readable time: HHH:MM:SS.fff 3:47:09.314159

- Absolute Time
  - Using ISO 8601 format
  - 20021211T101435.89Z

- (RTP Media Time)
  - Media-specific clock for the RTP timestamp
  - Synchronized with absolute time via RTCP

RTSP Sessions

- Shared state between RTSP client and server
- Establish by SETUP message
- Removed by TEARDOWN
  - Or due to some timeout
- Independent of underlying TCP connections
  - TCP connections may be closed and re-opened during a single RTSP session
- Typically bound to a single presentation
  - in case of SDP, valid for one SDP session (description)
- May contain several RTP sessions
  - e.g. one per media stream
RTSP Request Message

SETUP rtsp://ms.tkk.fi/movies/matrix RTSP/1.0
CSeq: 302
Date: 10 Dec 2002 15:35:06 GMT
Session: 47112344
Transport: RTP/AVP; unicast;
  client_port=4588-4589

<CR LF>
[Optional Message Body]

RTSP Response Message

RTSP/1.0 200 OK
CSeq: 302
Date: 10 Dec 2002 15:35:07 GMT
Server: Matrix-Server 0.4.2
Session: 47112344
Transport: RTP/AVP; unicast;
  client_port=4588-4589; server_port=6256-6257

<CR LF>
[Optional Message Body]
RTSP Protocol Operation: DESCRIBE

- Obtain presentation description from server
  - e.g. SDP
- Media initialization
  - Contains information about all embedded media streams
  - Support for aggregate / non-aggregate control
  - Allows a client to determine suitability of content
  - Choose encoding if possible
- Optional: description may be obtained out-of-band

Client Server

DESCRIBE

200 OK + SDP

RTSP Protocol Operation: ANNOUNCE

- Updates the presentation description actively from the server
  - e.g. add or remove media streams
- May be issued at any time

Client Server

DESCRIBE

200 OK + SDP

ANNOUNCE + SDP

200 OK
RTSP Protocol Operation: SETUP

- Initiate an RTSP session
- Reserve resources at the server
  - Server may redirect to other servers (e.g. if busy)
- Convey transport parameters for media sessions
  - Negotiate transport protocol
  - e.g. RTP/UDP vs. tunneling
  - Enable firewalls to open holes

Client Server

DESCRIBE

200 OK + SDP

SETUP + transport

200 OK + transport

RTSP Protocol Operation: PLAY

- Start streaming
- Allows to specify a variety of streaming operations
  - Range(s) to play
    - = seek operation
    - E.g. 10-20s; 30-45s; 60s-
  - Forward / backward
  - Speed
    - +3.0
    - -2.5
RTSP Protocol Operation: PAUSE

- Interrupt streaming
  - But keep resources allocated
- May take effect
  - Immediately or
  - At a specified point in time
- PLAY may be used to resume streaming

RTSP Protocol Operation: TEARDOWN

- Stop streaming
- Terminate RTSP session
  - Free resources
- Takes effect immediately
RTSP Methods

- OPTIONS
- DESCRIBE, ANNOUNCE
- SETUP, TEARDOWN
- PLAY, PAUSE
- REDIRECT
  - May be used by a server to refer a client to a different location

- GET_PARAMETER
  - Retrieve parameter value specified in the header (in the Session: context)
    - Returned in 200 OK response body as “Name: value” pairs
  - May be used for keep-alive purposes

- SET_PARAMETER
  - Set value of parameter(s) per response body (“Name: value” pairs)

- [RECORD]
  - Record a media stream at a server
  - Underspecified, not really supported, now removed from base spec

RTSP General Header Fields

(For reference only)

- Cache-Control:
- Connection:
- CSeq:
- Date:
- Timestamp:
- Via:
RTSP Request Header Fields

(For reference only)

- Accept, Accept-Encoding, Accept-Language
- Authorization
- Bandwidth
- Blocksize
- From
- If-Modified-Since
- Require, Proxy-Require, Supported
- Referer
- Scale, Speed, Range
- Session
- Transport
- User-Agent

Some Response Status Codes

- 100 Continue
- 200 OK / 201 Created
- 300 Multiple Choices
- 301 Moved Permanently / 302 Moved Temporarily
- 304 Not Modified
- 305 Use Proxy
- 400 Bad Request
- 401 Unauthorized / 407 Proxy Authentication Required
- 403 Forbidden
- 404 Not Found
- 405 Method Not Allowed / 406 Not Acceptable / 408 Request Timeout
- 451 Parameter Not Understood
- 454 Session Not Found
- 455 Method not valid in this State / 457 Invalid Range
- 461 Unsupported Transport
- 500 Internal Server Error / 501 Not Implemented / 551 Option not Supported
Response Header Fields

(For reference only)

- Accept-Ranges:
- Proxy-Authenticate: / WWW-Authenticate:
- Public:
- Location:
- Range: / Scale: / Speed:
- Retry-After:
- RTP-Info:
- Transport:
- Unsupported:
- Vary:
- Session:

Entities

- Entities contained in RTSP messages are typically presentation descriptions
  - e.g. an SDP message
    (Content-Type: application/sdp)
  - Should always fully specify the media stream(s)

- Header fields:
  - Content-Length:, Content-Type:, Content-Encoding:, Content-Base:, Content-Location:, Content-Language:
  - Allow:
  - Last-Modified:, Expires:
Interleaving

- RTSP should use RTP/UDP for media streaming
  - Not always feasible (e.g. firewall, see next slide)

- Interleaving of RTSP and media data
  - Escape binary data (“$”)
  - Define multiple “channels”
  - Specify packet length in binary
  - Yields a four byte header:
    - Interleaved with RTSP messages
    - Starts right after previous message
    - Length used to determine how many bytes to skip / pass

RTSP 2.0

- Presently under development (well advanced)
- draft-ietf-mmusic-rfc2326bis-13.txt

- Tons of editorial changes (readability, coherence, …!)
- Better state machine descriptions
- Updated (more coherent) semantics for various header fields
  - Significant alignment with SIP based upon experience gained there
- RECORD disappeared from base spec
  - Was underspecified anyway

- Support for NAT traversal upcoming
  - draft-ietf-mmusic-rtsp-nat-04.txt
Firewall Friendliness

Several means to support RTSP across firewalls
- Interleaving support
- Transport: header indicates port numbers, IP addresses, ...
  - Firewall logic does not need to parse SDP format
- SOCKS support

Still may be insufficient
- Firewalls may block RTSP in the first place
- “Last resort”: HTTP tunneling
  - Really bad (dubious!)
  - Boils down to a competition between firewall vendors and application developers
  - Defeats the purpose of a firewall in the first place
- Nevertheless: widely deployed (“HTTP streaming”)
  - Apple, Microsoft, …

RTSP: Implications for Session Descriptions

Session Announcements (SAP)
- Session Descriptions (SDP) specifies fixed parameter set
- May be updated by the server later on

HTTP-based retrieval of session information
- SDP specifies fixed parameter set or alternatives
- Client gets to choose one of these

RTSP-based session initiation
- SDP from server describes set of alternatives
- Clients may choose which one to use
- Both sides may update their offering / choice later

Need for negotiating session parameters
- Both side may provide suggestions, make choices, and update these
- Particularly relevant for interactive communications
- Generalized Offer/Answer model for SDP + negotiation with SDPng
“HTTP Streaming”

- Tunneling media and control in an HTTP connection

**Simplest case**
- Start replay before download is complete
- No extensions needed
- Mainly client-side operation
- But: server needs to use appropriate media file format

**Alternative: add additional headers (MS)**
- Preserve packetization of media within a TCP connection

---

**Old(?) MS HTTP Streaming Format**

```
0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7 0 1 2 3 4 5 6 7
+----------------------------------------+-----------------------------+-----------------------------+
|          Type                     |          Chunk Length       |               |
+----------------------------------------+-----------------------------+-----------------------------+
|                                            |                             |               |
+----------------------------------------+-----------------------------+-----------------------------+
|                        Sequence Number  |                             |               |
+----------------------------------------+-----------------------------+-----------------------------+
|                                            |                             |               |
+----------------------------------------+-----------------------------+-----------------------------+
|      Flags (Unknown)             |     Chunk Length (again?)   |               |
+----------------------------------------+-----------------------------+-----------------------------+
|                                            |                             |               |
+----------------------------------------+-----------------------------+-----------------------------+
|                                      :                                       :
+----------------------------------------+-----------------------------+-----------------------------+
|                                      :                                       :
|                                      +                                       +
+----------------------------------------+-----------------------------+-----------------------------+
```

© 2006 Jörg Ott
Sample Request Header (1/2)

GET test.asf HTTP/1.0
Accept: */*
User-Agent: NSPlayer/4.1.0.3856
Host: media_host
Pragma: no-cache, rate=1.000000, stream-time=0, stream-offset=0:0,
     request-context=1, max-duration=0
Pragma: xClientGUID={c77e7400-738a-11d2-9add-0020af0a3278}
Connection: Close

Sample Request Header (2/2)

GET test.asf HTTP/1.0
Accept: */*
User-Agent: NSPlayer/4.1.0.3856
Host: media_host
Pragma: no-cache, rate=1.000000, stream-time=0,
     stream-offset=0:0, request-context=2, max-duration=40"
Pragma: xPlayStrm=1
Pragma: xClientGUID={c77e7400-738a-11d2-9add-0020af0a3278}
Pragma: stream-switch-count=1
Pragma: stream-switch-entry=ffff:1:0
Connection: Close
Media Resource Control Protocol (MRCPv2)

- Another protocol to control media resources
  - Based upon a proprietary version by Cisco et al. (MRCPv1, RFC 4443)
- Enable a client to task a third entity to perform on its behalf
  - Media stream generation (basic and advanced speech synthesis)
  - Media processing (recording, DTMF/speech recognition, speaker verification)
MRCPv2 Overview (1)

- MRCPv2 defines a common framework for rather different application classes

- Commonalities
  - Media stream consumption or generation by a media resource server
  - Control of the media stream generation or processing by the client
  - Report on media stream contents, characteristics, and resource server status

- Text-based protocol
  - Start line + headers + message body
  - Borrows heavily from HTTP and RTSP
  - Yet, subtle differences (later)
  - Message bodies identified by entity headers (using MIME types, etc.)

- Symmetric operation
  - Both peers can initiate actions: Methods (client->server), Events (server->client)
  - Headers + contents to parameterize operations or deliver results

MRCPv2 Overview (2)

- Uses TCP as underlying transport (+ optional TLS)
  - Reliability required; limited real-time interaction requirements only (true?)
    - Or do we assume sufficiently well interconnected clients and media resources
  - One of more TCP connections multiplexed
    - Concept of logical channels

- Uses RTP for media streams
  - Explicit correlation to TCP control channels in SDP using new grouping

- Relies on SDP offer/answer (using SIP) for session setup
  - Connection-oriented media (TCP, TLS) as well as RTP sessions
**MRCP Overview (2)**

The diagram illustrates the interaction between a **Controlling Client** and a **Media Resource Server** via TCP connections and SIP+SDP signaling. The TCP connections are labeled as TCP connection 1 and TCP connection 2, indicating the pathways for control and media resources, respectively. The SIP+SDP protocol is shown to facilitate the setup and negotiation of media sessions.

**SIP** and **RTP** are used to establish and manage the sessions. The **SIP** connection initiates the session setup, while **RTP** handles the real-time media streams. The correlation between **SIP** and **RTP** is highlighted, showing how media sessions are correlated to **SIP** sessions.

The **Media Resource Server** is responsible for managing media resources, such as speech synthesis, which is indicated by the **m=application 9 TCP/MRCPv2** and **a=resource:speechsynth** fields in the SDP. The media resources are transported over **RTP** connections, as shown by the **m=audio 49170** and related **a=rtpmap** and **a=recvonly** fields.

**Answer:** The answer message includes a **response code**, indicating successful setup and correlation, **a=channel:32AECB234338@speechsynth**

---

**MRCP Packages**

- Different command sets defined for different packets
  - Building upon a small common subset of protocol elements
  - Otherwise largely independent of one another
  - Methods and events, response codes
  - Header fields
  - Content types (references to externally defined content formats)

- One package type per application
  - Speech Recognition
  - DTMF Recognition
  - Basic synthesis
  - Speech synthesis
  - Speaker verification
  - Recording

- Highly specialized for the specific application domain
  - You wonder why all this stuff goes into a single spec
Simple Example: Recording (1)

- **Methods**
  - RECORD — start recording
  - STOP — stop recording
  - START-INPUT-TIMERS — configuration

- **Events**
  - START-OF-INPUT — media stream recording has begun
  - RECORD-COMPLETE — recording done

- **Some useful headers**
  - Sensitivity-Level — for silence suppression
  - Media-Type — what to record
  - Record-URI — where to store recording
  - Trim-Length — limit length of recording
  - Capture-on-Speech — wait for speech
  - Various timeouts for input sensing, end of recording, …

- **Message bodies**
  - Captured recording (unless stored at a URI)

---

Simple Example: Recording (2)

```
C->S: MRCP/2.0 386 RECORD 543257
    Channel-Identifier:32AECB23433802@recorder
    Record-URI:<file://mediaserver/recordings/myfile.wav>
    Capture-On-Speech:true
    Final-Silence:300
    Max-Time:6000

S->C: MRCP/2.0 48 456234 200 IN-PROGRESS
    Channel-Identifier:32AECB23433802@recorder

S->C: MRCP/2/0 49 START-OF-INPUT 456234 IN-PROGRESS
    Channel-Identifier:32AECB23433802@recorder

S->C: MRCP/2.0 54 RECORD-COMPLETE 456234 COMPLETE
    Channel-Identifier:32AECB23433802@recorder
    Completion-Cause:000 success-silence
    Record-URI:<file://mediaserver/recordings/myfile.wav>;
    size=242552;duration=25645
```
More Media Control

- Media Gateway Control Protocol (MEGACOP)
  - Configuring (PSTN) media gateways for IP telephony
  - Controlling media resource functions in 3GPP

- Media Server Control Markup Language and Protocol
  - Controlling conference servers
  - Controlling Interactive Voice Response (IVR) systems

- MEDIACTL WG in the IETF (newly created last week)

- Lots of non-IETF work (e.g., W3C)

- Gains importance in the context of service creation for interpersonal communications (using SIP)