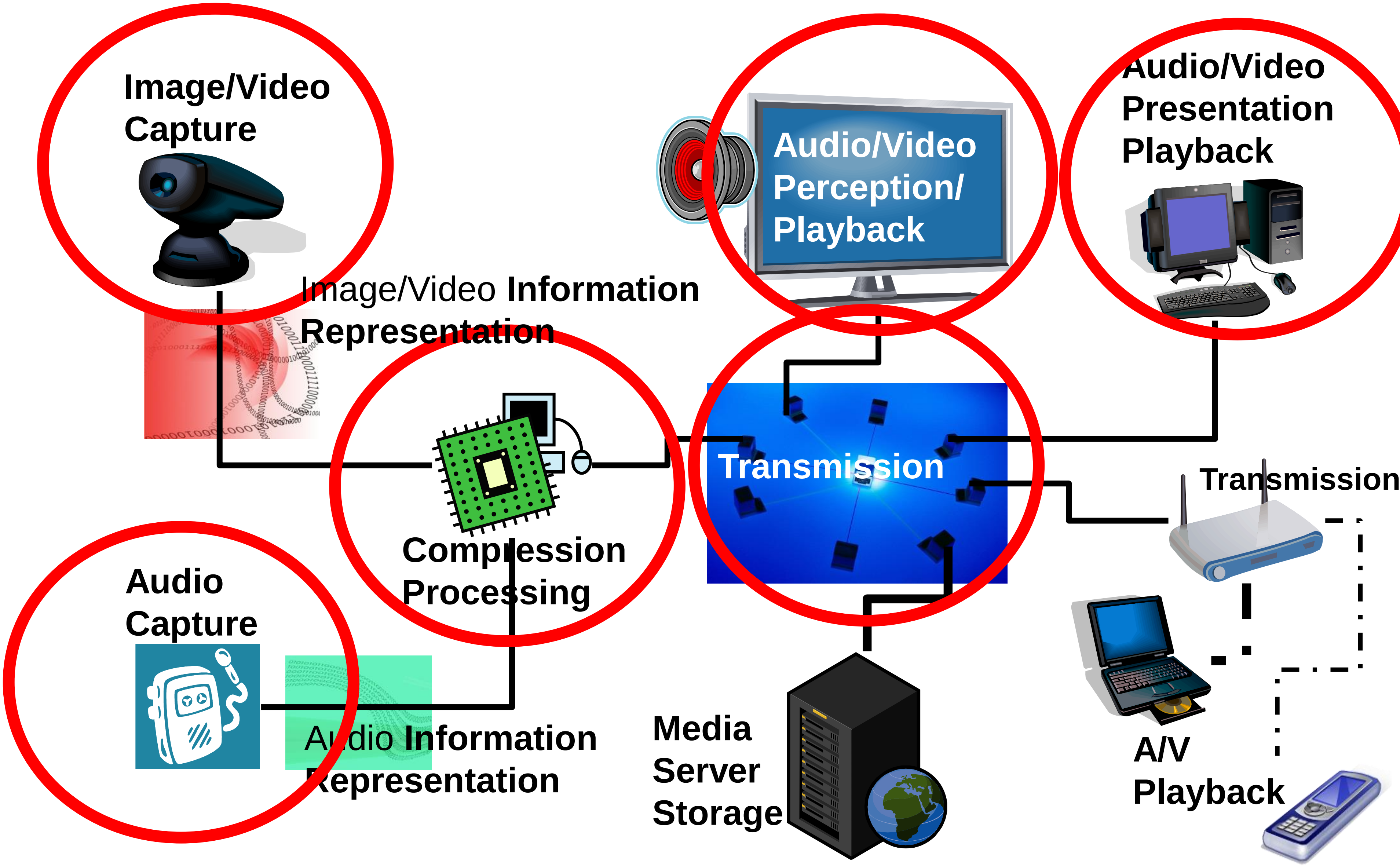
MP 2: Audio/ Video Streaming

CS414: Multimedia System

Instructor: Klara Nahrstedt

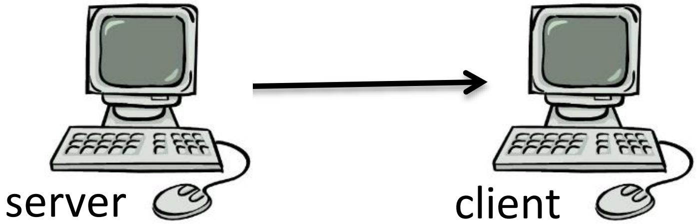
March 21, 2014

Covered Aspects of Multimedia



Learning Goals

- Learning media streaming using network

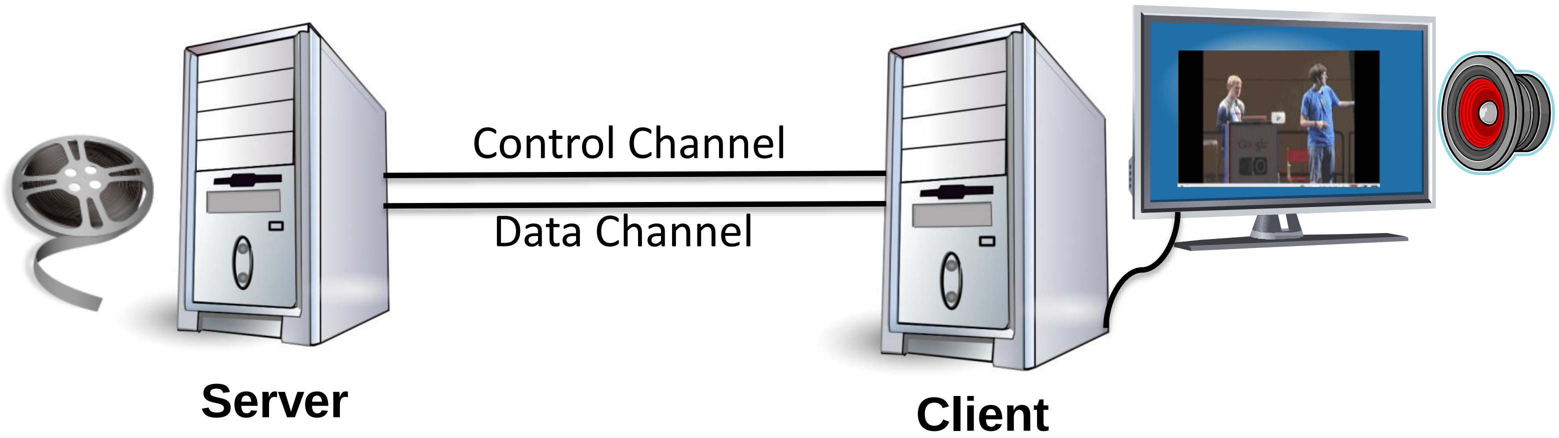


- Learning **audio video synchronization** mechanism

- Understanding following multimedia system concepts
  - **Resource Admission**
  - **QoS Negotiation**
  - **QoS Enforcement**
  - **Session Control**
  - **Session Adaptation**

- Understanding network dynamics such as **end-to-end delay, network jitter, and synchronization skew**

System Components

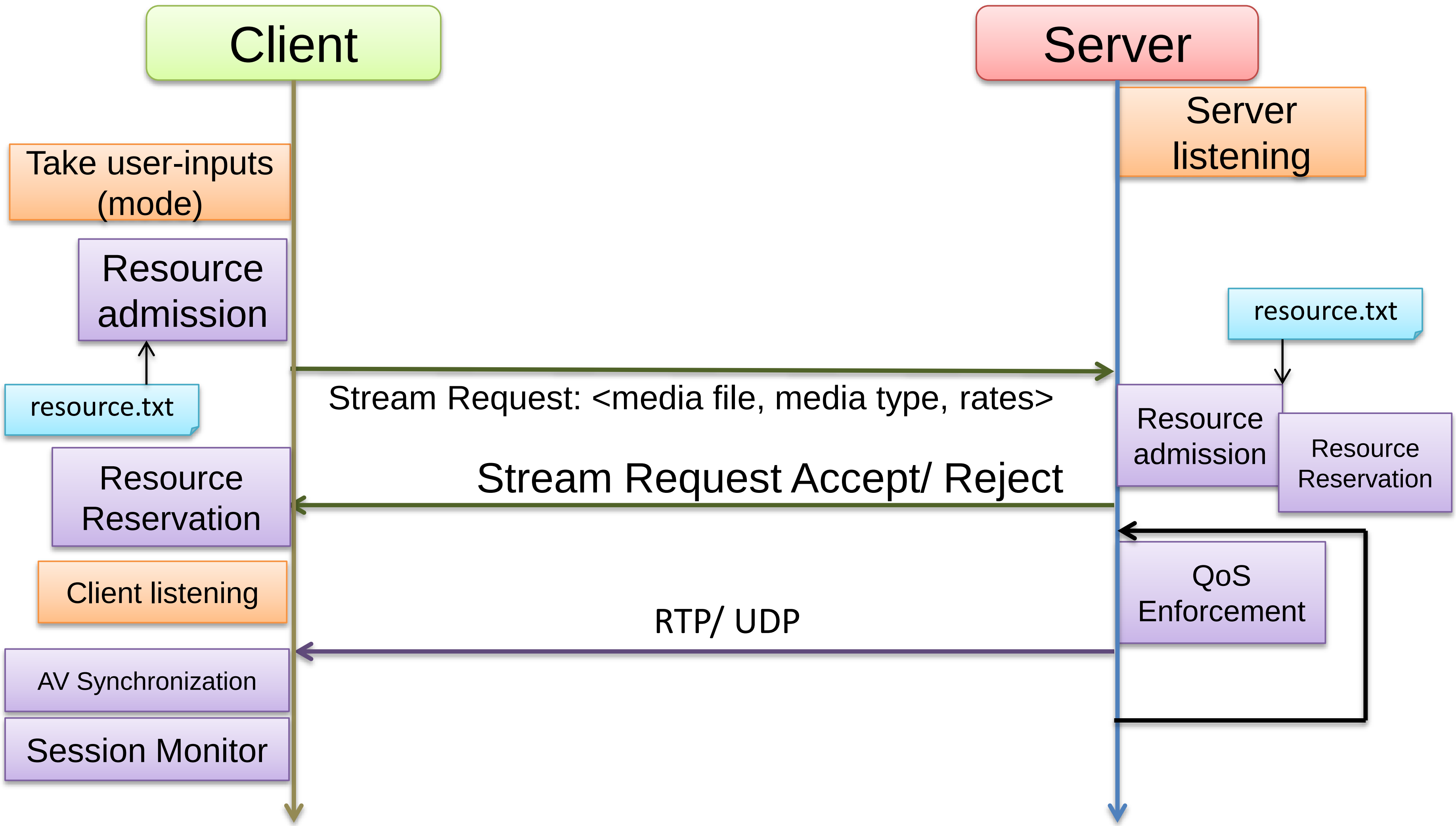


- Server
  - Stored Audio and Video
  - Video: 30 fps, Audio: 8000Hz
- Client
  - Requests for Video and/or Audio
  - Works in Two modes: **Active Mode** and **Passive Mode**

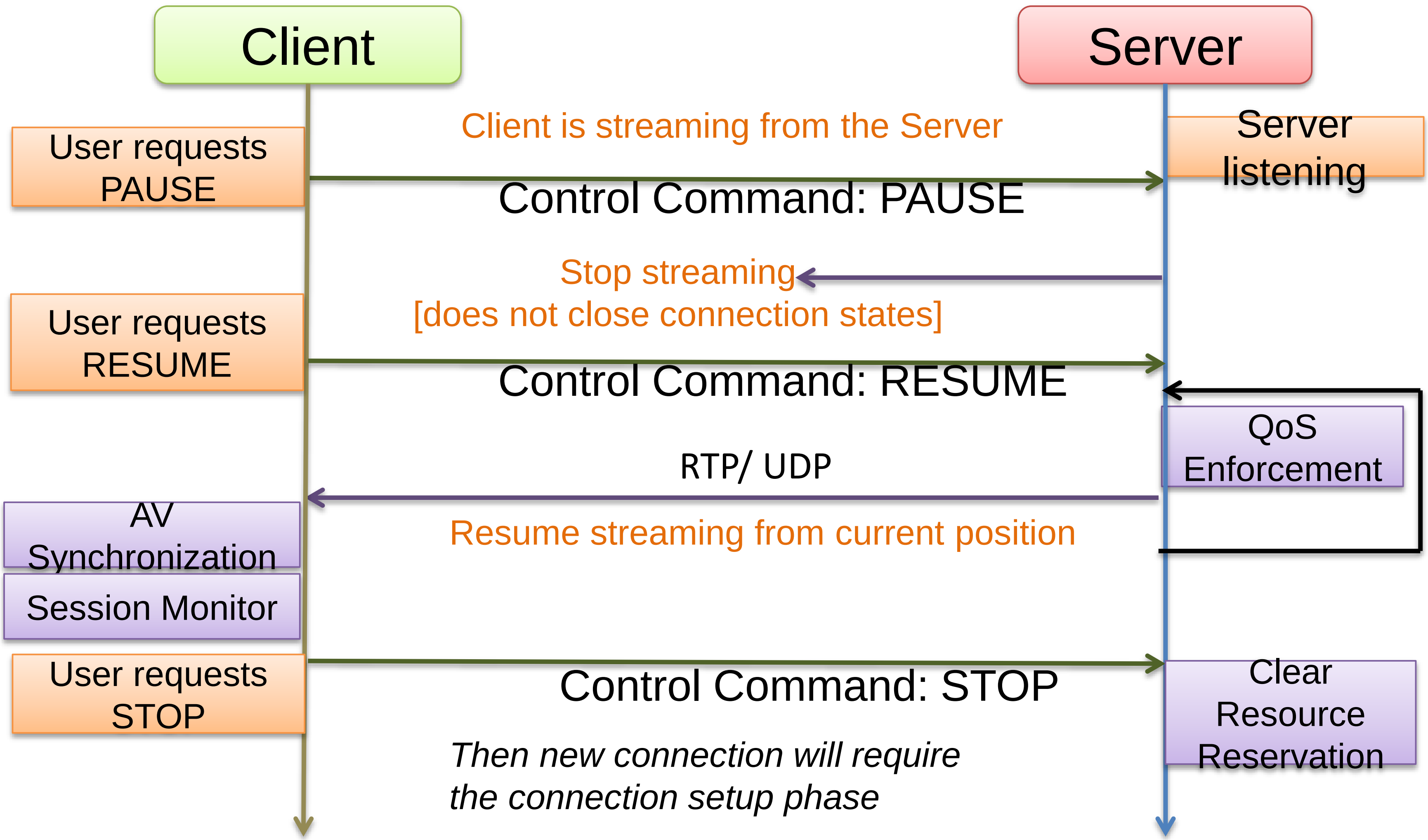
- **Active Mode:**
  Media type: Audio, Video
  Video Rate: 15 to 25 fps
  Audio Rate: 8000Hz

- **Passive Mode:**
  Media type: Video
  Video Rate: 10 fps

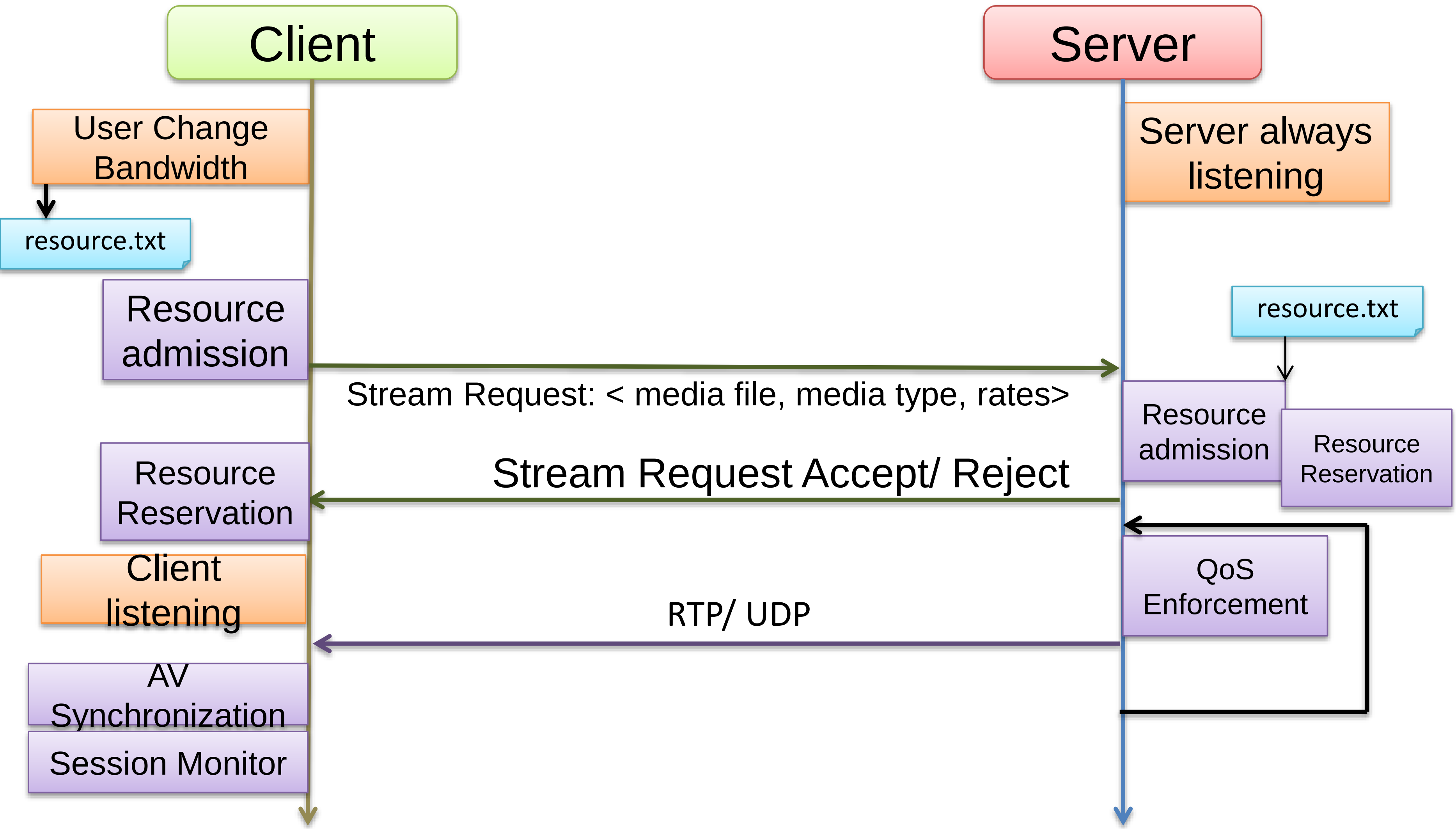
Functional Diagram: Connection Establishment



Functional Diagram: Control Playback



Functional Diagram: Session Adaptation



Resource Admission

- Client
  - Available Application Bandwidth AB_N
  - Application Frame Size, M_N = ?
  - Application Frame Rate, R_N = ?
  - Audio Bandwidth = 8000 * 16
  - Bandwidth
    - Active Mode: $B_N = (M_N * R_N)$ + Audio Bandwidth
    - Passive Mode: $B_N = (M_N * R_N)$

resource.txt

Optimistic Allocation

MJPEG Video

8000Hz Audio Signal

fps: 10

Assuming resolution is known by the client, e.g., stated in the file name like "MIB3-1080p.mjpeg"

In active mode, the video rate should be considered the Maximum between [15fps-25fps] provided that $AB_N >= B_N$

Notice: You have to be able to show your computation during the demo

Resource Admission

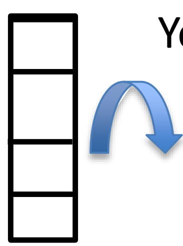
- Server
  - Available Application Bandwidth AB_N resource.txt
  - Required Bandwidth for Client B_C
  - Admission is successful if $B_C <= AB_N$
  - To support multiple client
    - *You also need to consider bandwidth assigned to other*

Client1	**10**
Client2	10

Resource Table

Rate Control: QoS Enforcement

- You can build leaky bucket
- You can implement token bucket
- You can use gstreamer rate control
- You can implement your own method

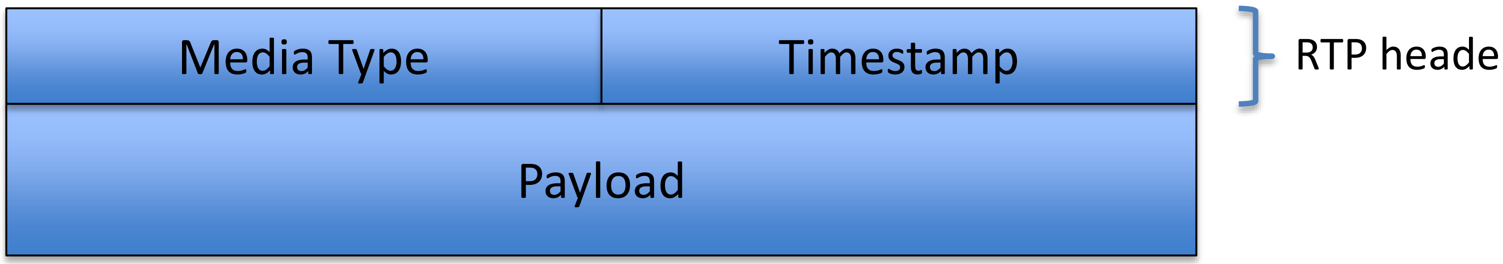


You an simply read using loop

Sleep (100) => 10 fps

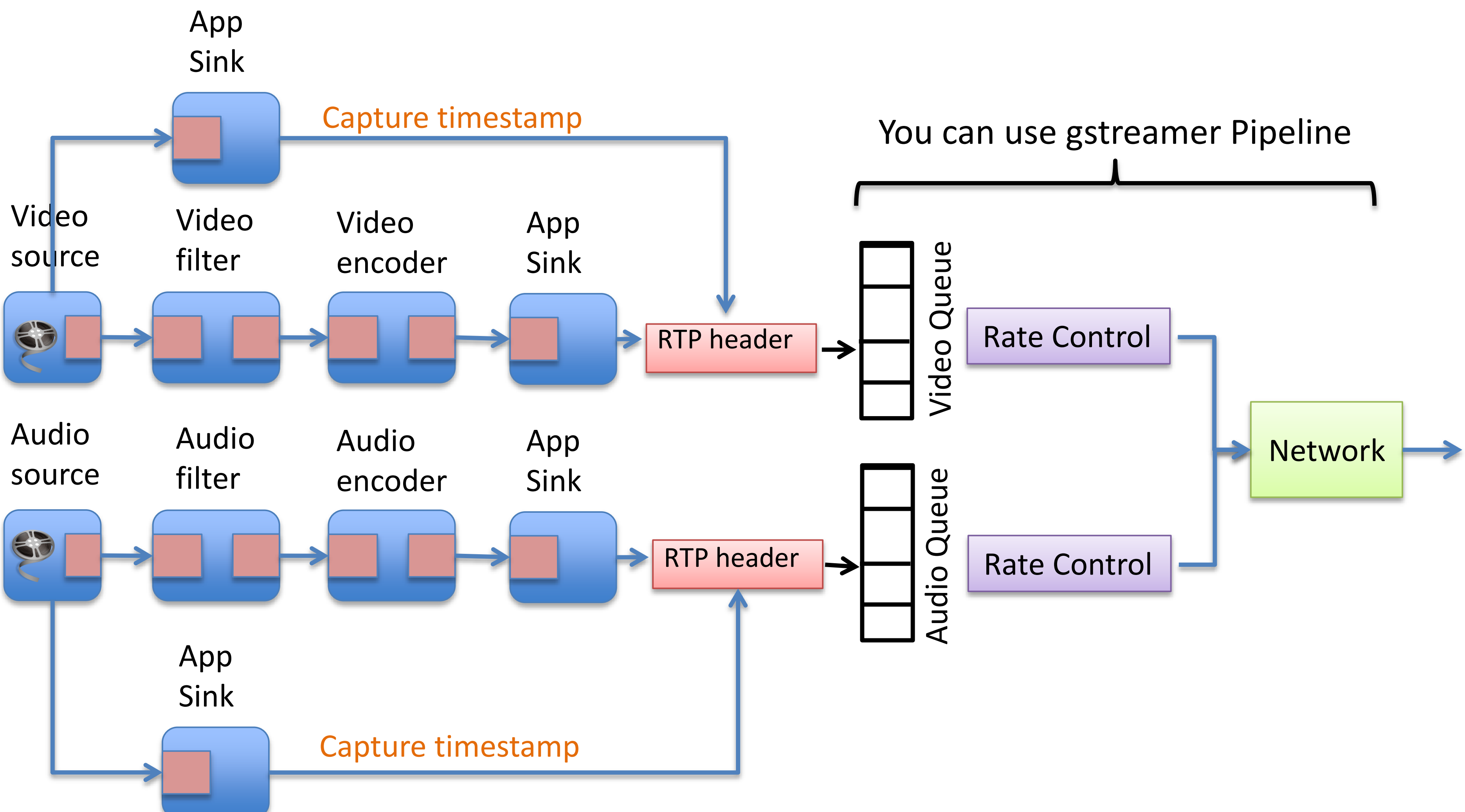
<u>**You have to do this for both audio and video**</u>

Synchronization

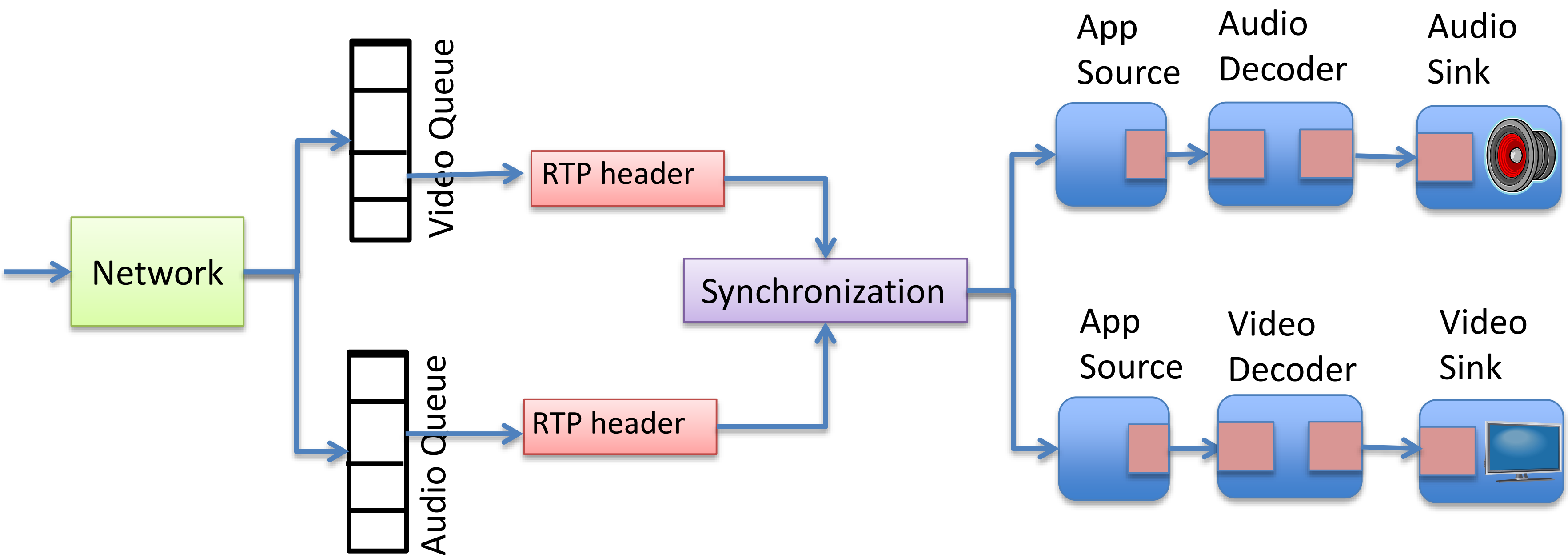


- Build your own synchronization algorithm
- Synchronization Skew < 80 ms
- Display synchronization skew on the screen

Server-side Data Flow Architecture



Client Side Data Flow Architecture



How to use AppSrc?

http://code.google.com/p/gstreamer-java/source/browse/trunk/gstreamer-java/src/org/gstreamer/example/AppSrcTest.java?r=509

Evaluations

Required Points: 100, Optional Points: 30

Features	Points	Properties
Video Streaming	10	Use MJPEG encoding. It will make rate adaptation easier.
Audio Streaming	10	you can use any audio encoding
A/V Synchronization	15	synchronization skew <= 80ms
Resource Admission	10	
Rate Control	15	
Session Control	10	STOP, START, PAUSE, RESUME, SWITCH, REW, FF
Session Adaptation	10	should allow to change “resource.txt” (via GUI)
Session Monitoring	10	synchronization skew, jitter, e2e delay, failure rate
Report Writing	10	clearly write your algorithms and development manual
Support Multi-Client	30	make sure you properly use server resources

Points will be considered based on live demo and interview performance