Machine Problem 3: Multi-Source Streaming System

CS414 Spring 2014: Multimedia Systems

Instructor: Klara Nahrstedt

Posted: April 15, 2014 Due: 5:00pm May 3, 2014

1. Introduction

In this final MP, you will use all the components built in previous MPs to build a multi-source streaming system. You will be able to claim that you have some experience with the "large-scale" system after you finish this MP. Trust me! It will be more complicated than you think.

Your final work will be part of the CS414 class project competition. Thus, we do not want to limit your topic. The only requirements of this MP are:

1. It has to be a video streaming system that includes at least three machines: two servers and one client.
2. It has to contain at least two video+audio sources, coming from different servers. These streams have to be played out simultaneously on the client side.
3. The client machine has to be able to assign different priorities to contents originated from different servers. The priorities can be changed on-the-fly by the user during a session.
4. All session management functionalities that you learned from MP2 (listed in Section 2.2).

As long as your system fulfills these requirements, you can target any application you want. The two (or more) video+audio sources can be pre-recorded files or real-time captured via cameras+microphones or both. An example application would be a multi-camera surveillance system, where client receives multiple streams from multiple servers, and each server handles a camera+microphone set. The client can switch the priority of the streams on-the-fly. For example, if the user sees something suspicious going on camera 1, she can assign a higher priority to get a higher resolution of that stream. **In the following, we will be using this surveillance system example to introduce details of the system design.** Your own application may need different system architecture/algorithms/protocols. As long as the requirements are fulfill, be creative to surprise us (and judges from Google)!

2. Problem Description

2.1 System Overview

Machine Problem 3: Multi-Source Streaming System

CS414 Spring 2014: Multimedia Systems

Instructor: Klara Nahrstedt

Posted: April 15, 2014 Due: 5:00pm May 3, 2014

For this MP, you have to create at least two servers and a client component. They should run on separate machines. The functionalities of server and client components are given below.

Server

Each server always listens to a port for any incoming client request. A camera is connected to each server to capture live audio and video streams at a fixed rate: 30fps for the video capture and 8000Hz for the audio capture. (Note: you have to capture at these fixed rates even though a client may request for lower rates of audio and video).

Client

A client is a separate machine with Graphical User Interface (GUI) that can send a connection request to the servers for video and audio streams of different rates. The client receives audio/video streams from each server, which can be in two different modes: *active mode* and *passive mode*. In active mode, the client requests for both audio and video streams. The video rate may vary between 15fps to 25fps depending on the resource availability and the audio rate is fixed to 8000Hz. The video resolution is high in active mode (640X480). However, in passive mode, the client requests for only the video stream (no audio) with 10 fps, and the resolution of the video stream is low (320X240). A client can request any combination of active and passive modes from the two servers. For example a client can request active mode communication from server 1 and passive mode from server 2 or vice versa. It may also be possible that the client requests streams in active modes from the both servers. In that case, you need to implement the MUTE functionality so that only one audio is played. MUTE does not stop the streaming rather drops the audio frames at the client side. Likewise, a client can also request streams in passive mode from the both servers.

Machine Problem 3: Multi-Source Streaming System

CS414 Spring 2014: Multimedia Systems

Instructor: Klara Nahrstedt

Posted: April 15, 2014 Due: 5:00pm May 3, 2014

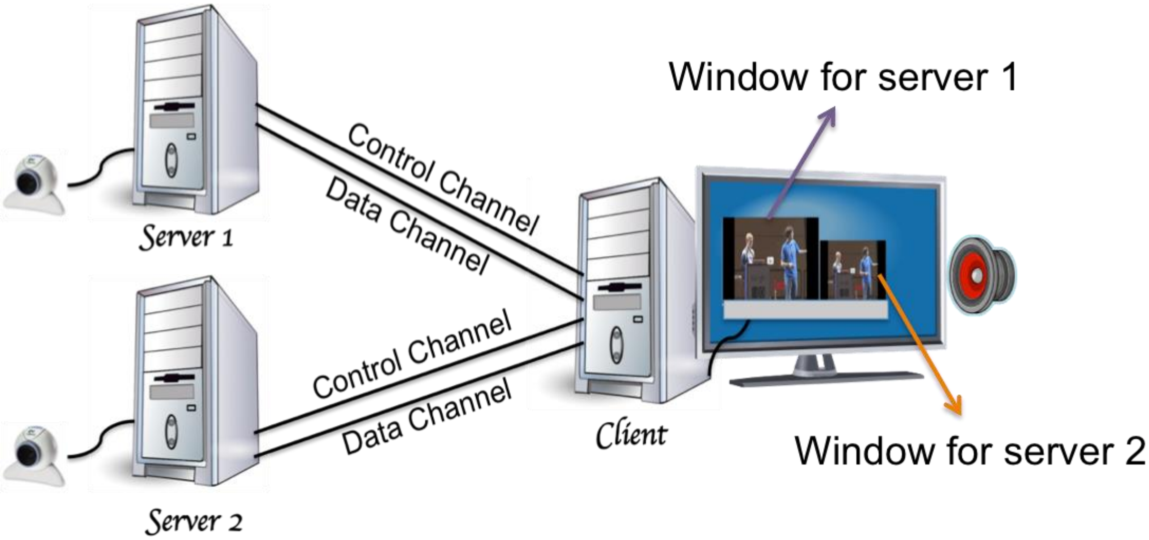

Figure 1. Communication Architecture of a Multi-view Video Surveillance System

Figure 1 shows the communication details of this MP. The GUI should display two video windows coming from the two servers. The GUI should also allow the users to change modes for any server at any time (**without stopping the current stream**).

In active mode, the audio is played on the sound card. Remember that, the audio and video should be properly synchronized in active mode. Displaying multiple videos on the same screen can be confusing. Therefore, remember to design your GUI nicely.

2.2 Functional Components

As a part of this MP, you will use the following functionalities from MP2 (the descriptions are given in MP2 specification).

- Client Resource Admission
- Resource Negotiation between Server and Client
- Server Resource Admission
- Server and Client Resource Reservation
- QoS Enforcement (Rate control at the server side)
- Data Plane Communication: video streaming, audio streaming

- Audio Video Synchronization
- Session Control: START, STOP, PAUSE, RESUME, SWITCH, FAST_FORWARD/REWIND (for pre-recorded files only)
- Session Adaptation: Rate should be re-negotiated if bandwidth changes
- Session Monitoring: Synchronization skew, jitter, bandwidth, frame rate

3. Required Features

Video Streaming (15 points): The client should be able to display the video from the servers in different quality modes (like the active and passive modes in our example).

Audio Streaming (10 points): The client should be able to play the audio from the servers. Based on your application, audio content might not always be presented on the client side (like the passive mode in our example).

A/V Synchronization (15 points): When both audio and video content are presented, the audio and video received from the server should be properly synchronized. You will be judged depending on your synchronization perfection.

Resource Admission and Negotiation (10 points): You should implement a simple resource admission and negotiation protocol.

QoS Enforcement (Rate control) (10 points): You should create a proper rate control mechanism so that the network always gets the packet according to the negotiated rate. You can use leaky bucket, token bucket or gstreamer rate control mechanism.

Session Control (20 points): The client GUI should implement the STOP, START, PAUSE, RESUME, (and FF/REW for file sources) functionalities. You should also be able to SWITCH modes of the viewing across the servers at any time. In addition. **You should not stop any time while switching stream priorities.**

Session Monitoring (10 points): Display the monitoring output on the GUI in the text form. **You must monitor failure rate, jitter, frame rate, synchronization skew and bandwidth at the client for individual streams**.

Report Writing (10 points): Write a clean report describing your approach, algorithm and assumptions. You should submit two reports: development manual and user manual. The development manual should describe your design approach, assumptions and architecture. On the other hand, the user manual should contain instructions on how to compile and run your code.

4. Bonus

Creativity and Practicality (10 Points): Be creative about the application you build. (Obviously, building an exactly same system described in this spec gets zero point.) Also keep in mind that a creative system has to be practical, too. Practicality will be judged on whether the targeted scenario of your application is realistic and whether your application can truly serve the purpose/solve the problem/fulfill the needs that you claim.

5. Submission

Pack all source codes and documents into a zip file or tar ball and submit them through Compass. Do not submit your solutions through email unless there are technical problems with the Compass system. The submission deadline is May 3rd at 5:00 pm. You can get up to two days bonus for each MP, but please remember you can use only 3 bonus days throughout the semester. Further late submissions are not accepted and will get 0 point.

5.1 Source Code

You should only submit your source code, Makefile, and any open source libraries (or jar files) you use in your solution into compass. Do not include any pre-compiled obj files (.o), binary execution, or pre-recorded media files in the directory.

5.2 Documentation

Your documentation should include two parts: user manual and development manual. The user manual should include all instructions on how to compile and install your source code, and how to run your program and test all features. If you have designed any GUI, it is better to attach some screen shots to explain. The development manual should have the implementation details of all features. The implementation details include program flow, key data structures; media file formats, important algorithms, and so on.

6. Evaluation

Machine Problem 3: Multi-Source Streaming System

CS414 Spring 2014: Multimedia Systems

Instructor: Klara Nahrstedt

Posted: April 15, 2014 Due: 5:00pm May 3, 2014

The evaluation will be done by face-to-face interview with each group. You will run your program. We may ask you to show the source code implementing different functionalities. Your solution is evaluated based on how many features you have implemented and demonstrated. In order to get full score (100 points) of this MP, you need to implement all 100-point required features. 10-point optional features will be used to offset your lower marks at MPs, homework's or exams.
Evaluation will happen on **5pm, Monday May 5th**. A sign-up sheet will be provided to schedule your demonstration slot. **You MUST use at least THREE machines for the demonstration.**

[Please Attend MP3 Lecture on Friday April 18th, 2014. TA will discuss various design alternatives and gstreamer functionalities that you can use for implementing this MP. You are encouraged to bring your design to the lecture. TA can discuss with you about whether your application fulfills all the requirements.]