Meng Project (May,1999)

Reliable Video on Demand Services over Heterogeneous Network by Multicast.

Ming Lu
Computer science, cornell university

Abstract:

This project is a demo of Video on Demand service over Heterogeneous network.

A system includes registration and video server, video client is completed. The key to the project is using layered encoding/decoding scheme and multi-channel multicast to cope with the different bandwidth of the client network. Two kind of encoding scheme are supported while one of them is invented to achieve real-time performance. Two kinds of video server, real-time video server and MPEG video server are supported.

Background:

As the result of deregulation, traditional phone networks, cable networks as well as data networks are merging. Thus to provide service to all potential customs who might have phone access, cable access or fiber access, etc, it's necessary for the application to cope with the different bandwidth, different error rate, etc. of the client network.

On the other hand, video on demand service, will be the next level of service beyond today's "pay per view" which customs can only order the several videos available and customs have no direct control of when the video will be shown. Video on demand service will provide customs the freedom to view whatever they want at the time of their convenience. Video on demand will benefit not only entertainment, but also education, business, etc.

In this project, multicast will be used to achieve better bandwidth and srm2.0 from UCB is used to provide the lower level multicast.

High level view of the project:

1. **What to achieve:**

   Suppose we have a video frame as Fig3, the customers can receive the Fig1, Fig2, Fig3 depending on what their bandwidth is.

   Then the encoded images will be multicast out into different channels, those customers with high bandwidth can subscribe to all channels, while other customers can subscribe lowest several channels depending on their bandwidth.

Final screen Shot:

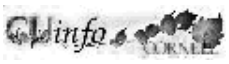


Fig1 This is a screen shot for low bandwidth, which can accommodate only One Channels

Fig2 This is a screen shot for median bandwidth, which can accommodate two channels.

Fig3 This is a screen shot for high bandwidth, which can accommodate three channels.

2) ***How it's done***

The original video is encoded frame by frame into several images which contains image components of different frequencies. For example, the Fig3 above can be encoded into Fig.4,Fig,5,Fig,6,Fig7, which contains from low to high spatial frequencies of the original image.

Fig4 Low-low encoded image Fig.5 Low-high encoded Image

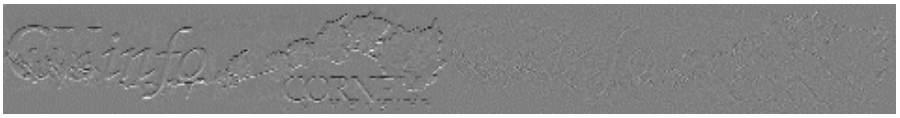
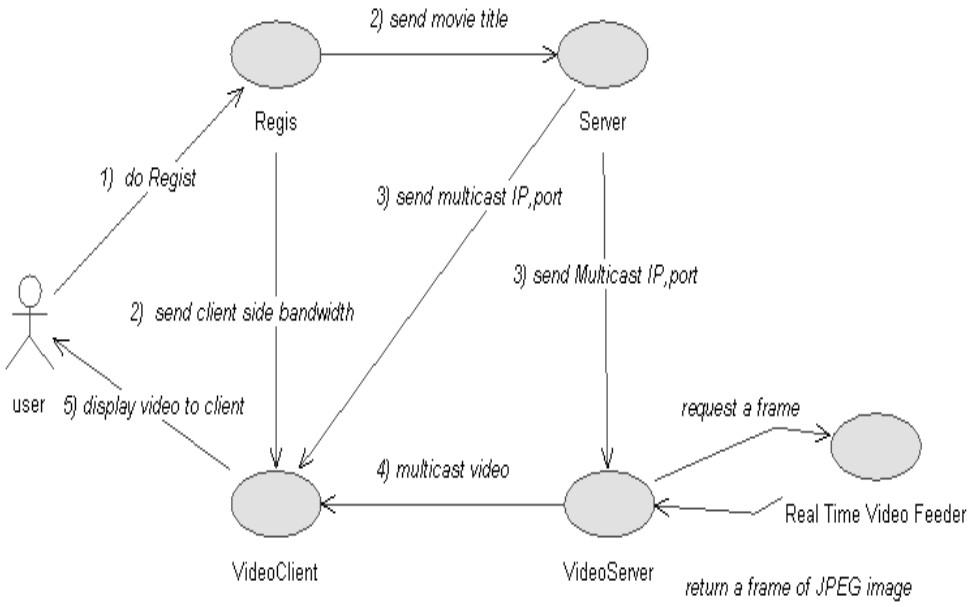


Fig.6 High-Low encoded Image Fig.7 High-High Encoded Image

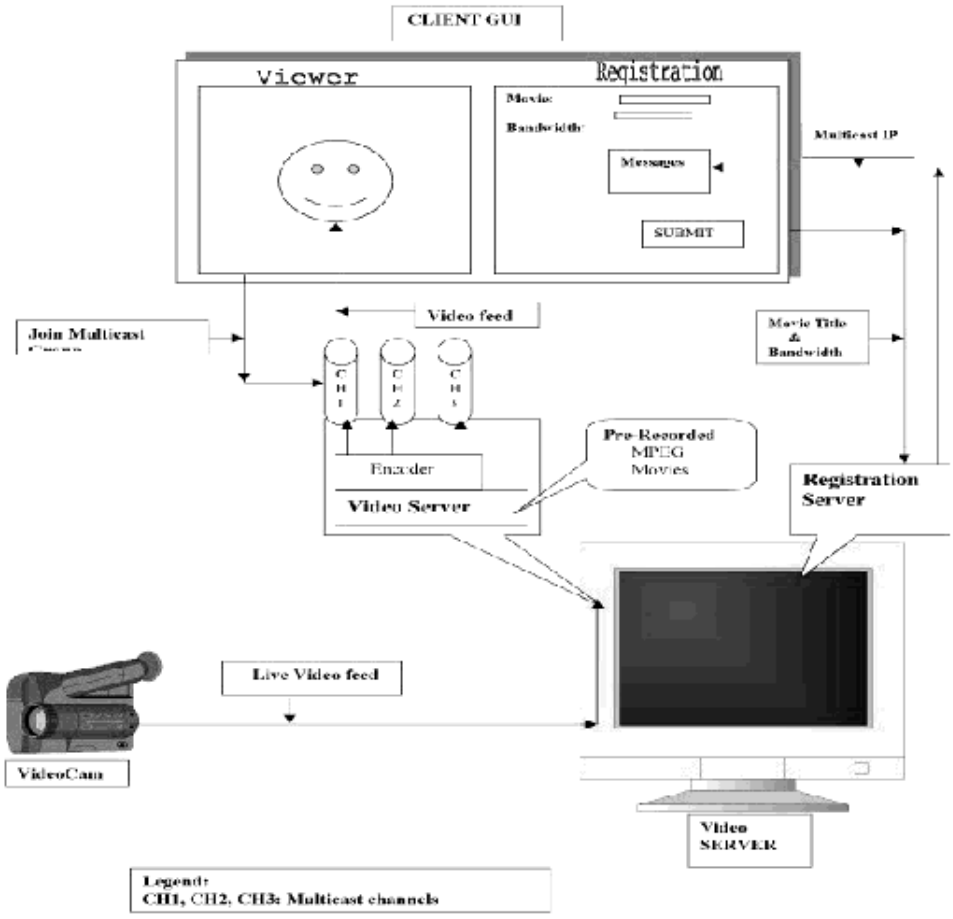
2. **Simple system view**



Note: The system follows 1) to 5) step by step.

1. users do the registration through the Regist client
2. Regist client send request to regist server and send local bandwidth information to VideoClient .
3. Regist Server send multicast IP and port to both Video Client and Video Server. It also sends what kinds of video to the VideoServer
4. Video Server multicast video out.
5. Video Client get the video through multicast channels, and reconstruct the image, display it to the user.

3. detailed system view



Detailed system component description:

I. Registration

The Registration system includes one Registration client, one registration server.

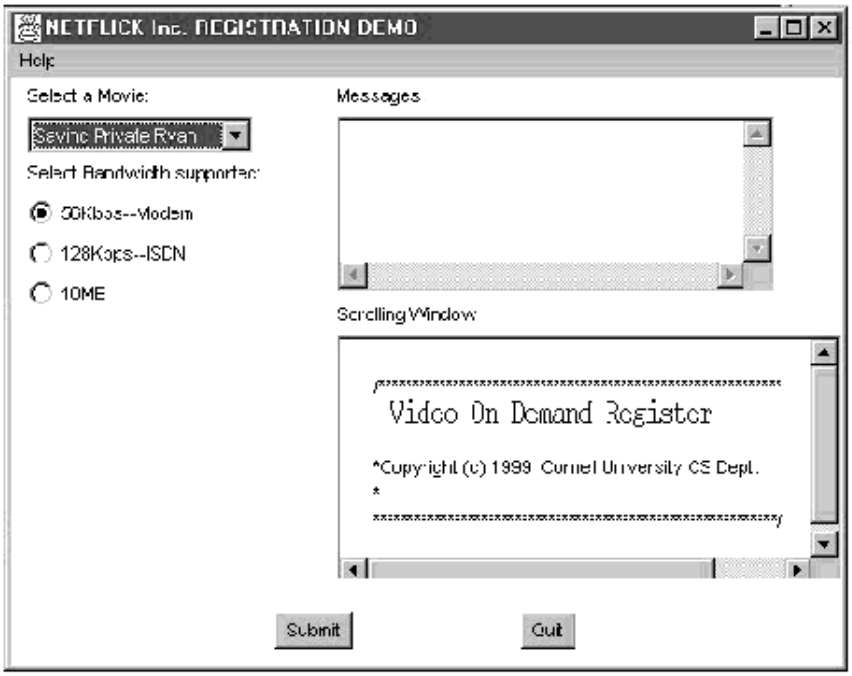
Registration Client
The registration client is used to accept request such as what movie, etc,
And let user to provide bandwidth of their network. Fig.8 is a screen shot of the

registration client.

Registration server :

This server is responsible to accept user input and decide if the request can be guaranteed. If it is, it'll send multicast information such as multicast IP, multicast port, etc to both the Video Server and Video Client.

A VIEW OF THE REGISTRATION CLIENT INTERFACE



II-Video Multicast

Video Multicast has a Video Sever and a Video Client.

Video Server :

The video Server is responsible to get the video, encode the video frame by frame using Pyramid Encoding techniques. (I create it myself, and detailed information will be listed later.) and multicast the video out into several channels. This is video Server is designed

in the way, that different video source, such as real time video from a live camera, MPEG video clip, etc can be easily configured. The encoder for the video server is also configurable. At this time, only wavelet encoding and pyramid encoding is supported.

Video Client:

The video client is responsible to subscribe multicast channels depending on bandwidth of local network. Once the video frames from several multicast channels is received ,it'll reconstruct the vide frame and send to Video Client GUI for user watching.

Multicast for Video Server and Video Client:

Scalable multicast is used for the multicast. Multicast is inherently NON reliable. A layer is used to provide reliable multicast by retransmission. The up layer is able to configure the reliability for this multicast. For the video, it is multicasted with limited reliability which is configurable. The command information between the Video Client and Video Server is delivered by reliable multicast.

Video Caching:

The video client is able to cache the video it received, for replay later.

III. Video Encoding:

The Video on demand over heterogeneous network requires the encoder at Video Server side can encode the video stream into several video streams. As human beings are very sensitive to low spatial frequency details of the image, it's reasonable for those user at the low bandwidth only receive the low spatial frequency part of the video stream while others may receive more. Layered encoding /decoding is needed. In this project, wavelet is the first option, such it can preserve high quality of video quality with good compression. However it turns out to be too slow. So invented a new layered encoding scheme which can encode the video streams into several channels with different spatial frequencies.

The basic idea of this new layered encoding is similar to the wavelet encoding.

It first encoding along x-dir, low frequency components goes to low end of low end of intermediate image, and high frequency components goes to another end.

Then do it along y-dir. Thus, four image can be computed to represent the LOW-LOW, LOW-HIGH,HIGH-LOW,HIGH-HIGH. Packet the LOW-LOW image into **one** channel, and pack other three into another channels. If more than two channels is necessary, repeat

the same operation over the LOW-LOW image, etc.

The Decoding scheme is a reverse process of the encoding. No compression is supported at this stage. DCT compression will be supported later.

IV.Srmlib 2.0

This project uses srmlib 2.0 developed by UCB to provide multicast. Here is a brief description of srmlib2.0.

1. srmlib uses event-driven architecture.
2. Uppler layer call define callback function to customize event-handling.

   These functions are :

   void srm_recv(srm_source_t source, unsigned int cid, unsigned int seqno,

   const unsigned char *data, int len,

   const srm_adu_info *info);

   int srm_should_recover(srm_source_t source, unsigned int cid,

   unsigned int ss, unsigned int es)

void srm_source_update(srm_source_t source,

const unsigned char *info, int len);

void srm_recv_cid(srm_source_t source, unsigned int cid, unsigned int parent,

const unsigned char *name, int name_len);

The srm_recv function defines how the upper layer will react to the event that

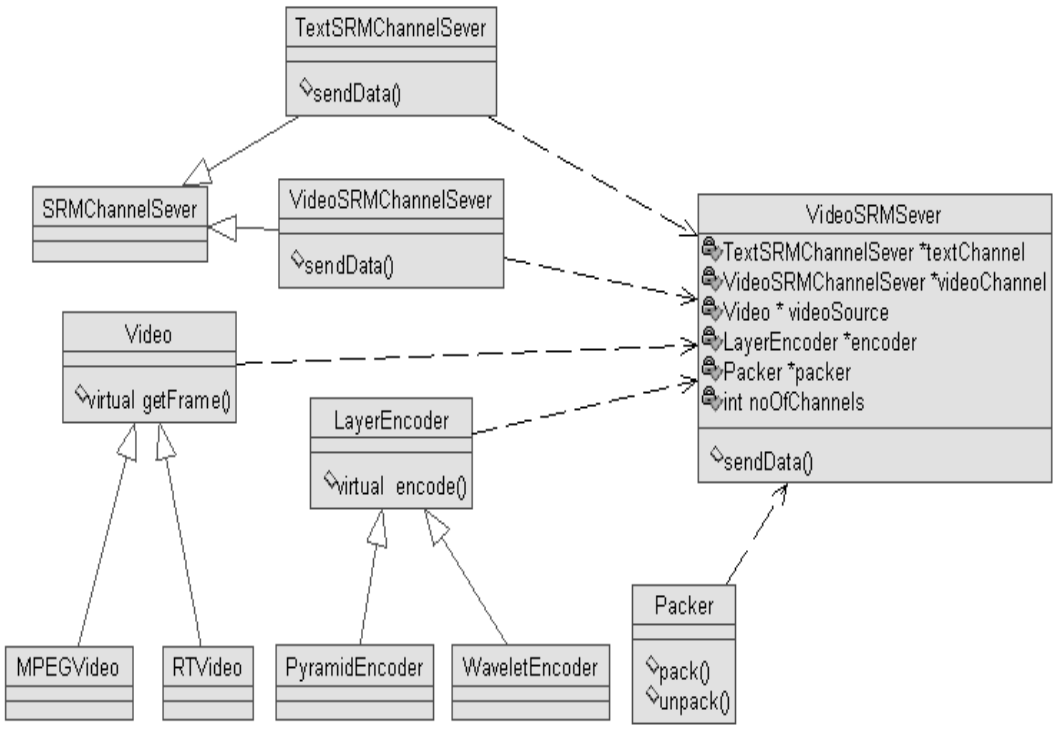
A package has been received.

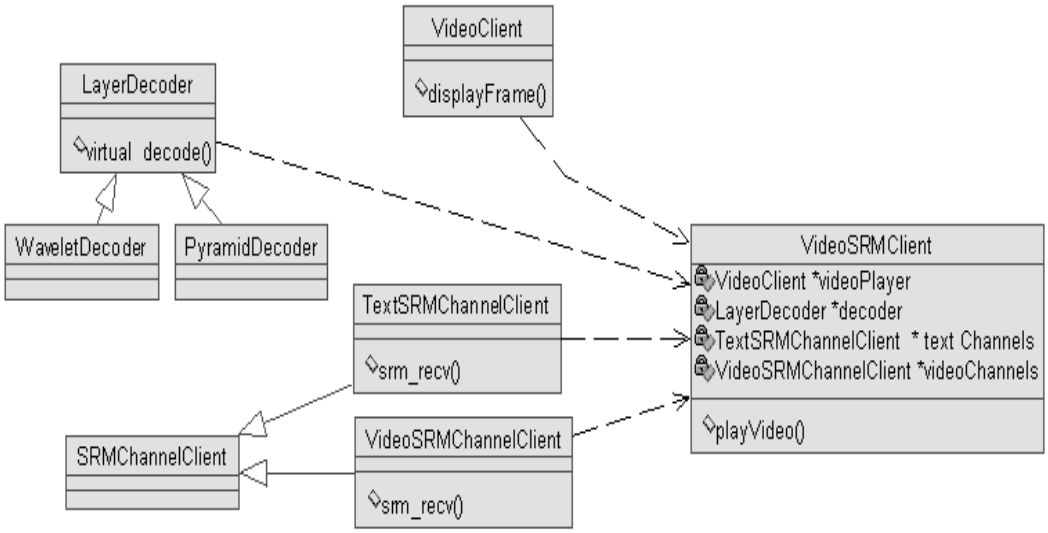
In this project, the client side will define this function to process the data received.

Coding structure:

There are four main classes, the VideoSRMSever and VideoSRMClient, which serves the server side and client side. RegistrationSever and RegistrationClient servers registration.

Here is a UML representation of the class structure.





Concerns about the design above:

1. VideoSRMChannelSever, VideoSRMChannelClient, TextSRMChannelSever, TextSRMChannelClient defines the channel behavior of video channel and text channel. Each channel uses one pair of multicast IP and port. Text channel is ued to pass data which should be reliable. If an error occurs, retransmission will be requested. On the other hand, video channels is defined as non-reliable. In the case where the communication media is extremely unreliable, retransmission will be requested to guarantee the quality of the video data.
2. VideoSRMChannelClient define a callback function srm_recv to handle the received data at the event when a package is received. It'll call the function playVideo of VideoSRMClient to copy the data to some buffer and reconstruct the image in the suitable time.
3. VideoSRMSever has pointers to Video, LayerEncoder to define the video source, and video encoder. It also has pointers to VideoSRMChannelSever and TextSRMChannelSever. Once a frame of image is received from the video source, it'll call the encoder to encode the image into several channels, and call the function sendData of VideoSRMChannelSever to push data into each channel.

4) VideoSRMClient has pointers to VideoClient, and LayerDecoder . The

VideoClient defines the video client which will be display the video or cache the video. The LayerDecoder defines how the client will reconstruct the image.

It define a function playVideo which will be called by a callback fucntion of VideoSRMChannelClient to handle the received data at the event a package is received.

4. In this project, two kinds of encoder/decoder are tried, one is wavelet encoder/decoder. It has great compression rate with good video quality. However, it's really slow, it's very difficult to achieve 30 frames/second real-time video processing. Then another one is used, I invented it myself, I called it as PyramidEncoder/PyramidDecoder. It's not complicated, however, it has pretty good quality of video and fast enough for real-time image processing. (Detailed of the information is listed in sections above).
5. In this project, two kinds of video source is implemented, one is Real-time video which is captured from live -camera. Once is mpeg video which is decoded by mpeg decoder in the background process as needed. The real-time video is obtained from a HTTP video Sever which provides jpeg compressed image frame by frame. So Real-time video in this project is actually a client of this HTTP video server.
6. In this project, videoClient is a java application. It receives constructed image from VideoSRMClient through sockets.
7. Registration Sever will accept the Registration Client's request and decides if the request can guaranteed. If it is, it'll send multicast IP and port number to both the VideoSRMClient and VideoSRMSever. It also send the movie title to the VideoSRMSever. On the other hand, the Registration client will also ask the client what the bandwidth of local network is. This bandwidth will be passed to VideoSRMClient to decide how many channels it'll subscribe from the multicast channels. The VideoSRMSever doesn't need this information, it'll use the system defined MAX_noOfChannels for decoding and multicasting.

Acknowledgment

Great thanks goes to Bela Ban, computer science, cornell university
Great thanks to Prof.Birman, my favorite faculty in computer science, cornell university.

References:

SRM : scalable reliable multicast, http://www.cs.berkeley.edu/mash