DynaVideo – A Dynamic Video Distribution Service

Luís Eduardo C. Leite
DIMAp – CCET - UFRN
Campus Universitário –
Natal – RN - Brazil
+55 84 215 3814
leduardo@natalnet.br

Renata Sofia Alves
DIMAp – CCET - UFRN
Campus Universitário –
Natal – RN - Brazil
+55 84 215 3814
renata@natalnet.br

Guido L. Souza Filho
DIMAp – CCET - UFRN
Campus Universitário –
Natal – RN - Brazil
+55 84 215 3814
guido@dimap.ufrn.br

Thaís V. Batista
DIMAp – CCET - UFRN
Campus Universitário –
Natal – RN - Brazil
+55 84 215 3814
thais@dimap.ufrn.br

ABSTRACT
Most solutions proposed to implement audio and video distribution services have been designed considering specific infrastructures or have been tailored to specific application requirements, such as stream types or clients which will be supported by the service. The performance of distributed services is becoming increasingly variable due to changing load patterns and user mobility. This paper presents the Dynamic Video Distribution Service – DynaVideo. The service was designed to distribute video in a way that is independent of the video format and to interact with different types of clients. The service may be used to distribute video over any digital network, however it is focused on the Internet. The main feature of the DynaVideo is the possibility of configuring dynamically the service to a specific demand.

1. INTRODUCTION

The costs reduction of high speed networks, the development of powerful and cheaper microprocessors and the consolidation of audio and video standards to applications such as Digital TV, Video on Demand, High Definition Television and Interactive TV, are factors that motivate the development of video distribution services over digital networks, such as Internet.

This paper presents the Dynamic Video Distribution Service – DynaVideo. The service was designed to distribute video in a way that is independent of the video format and to interact with different types of clients. The service may be used to distribute video over any digital network, however it is focused on the Internet. The main feature of the DynaVideo is the possibility of configuring dynamically the service to a specific demand. Applications that deal with video distribution, such as broadcast of digital television, normally have to deal with abrupt variations on its demand. The number, type and location of the clients of the service vary in short time. This occurs every time that an interesting program begins to be exhibited.

Nowadays many systems can distribute video over digital networks like the Internet. Real System of the Real Networks [7] transmits video streams in many formats, but the focus is on the support of its proprietary format RM. The video may be transmitted with IP Multicast [13], TCP [16], UDP [15] or HTTP [14]. The Microsoft Windows Media Service [9], encode the video stream with its proprietary format ASF. The following video formats: BMP, WAV, WMA, WMV, ASF, AVI [5], e MPEG-1 [3] are also supported and can be transmitted with UDP, TCP, HTTP / TCP or IP Multicast. The IBM VideoCharger [8] transports MPEG-1, MPEG-2 [4], AVI, WAV, LBR e QuickTime [6] video streams, using RTP [11], TCP, HTTP or IP Multicast. Once configured, distribution services based on these platforms remain unchanged and cannot automatically adjust its configuration to demand variations.

This paper is structured as follows. Section 2 presents the DynaVideo Architecture. Section 3 presents some experiments results. Finally, Section 4 contains the conclusions.

2. DYNAVIDEO ARCHITECTURE

The DynaVideo service can be dynamically configured. This is its main feature. This flexibility allows that the service automatically adjust itself to demand variations and guided its conception. The idea is that the service continually tries to find an optimized configuration to attend to a given demand. In

DynaVideo the demand is defined by the number, type and location of the service clients. The target applications of the service are Digital Television broadcast and Video on Demand. In such applications, the demand can change from a few users to millions of them in a short time. Figure 1 shows the DynaVideo architecture using the UML component diagram [2].

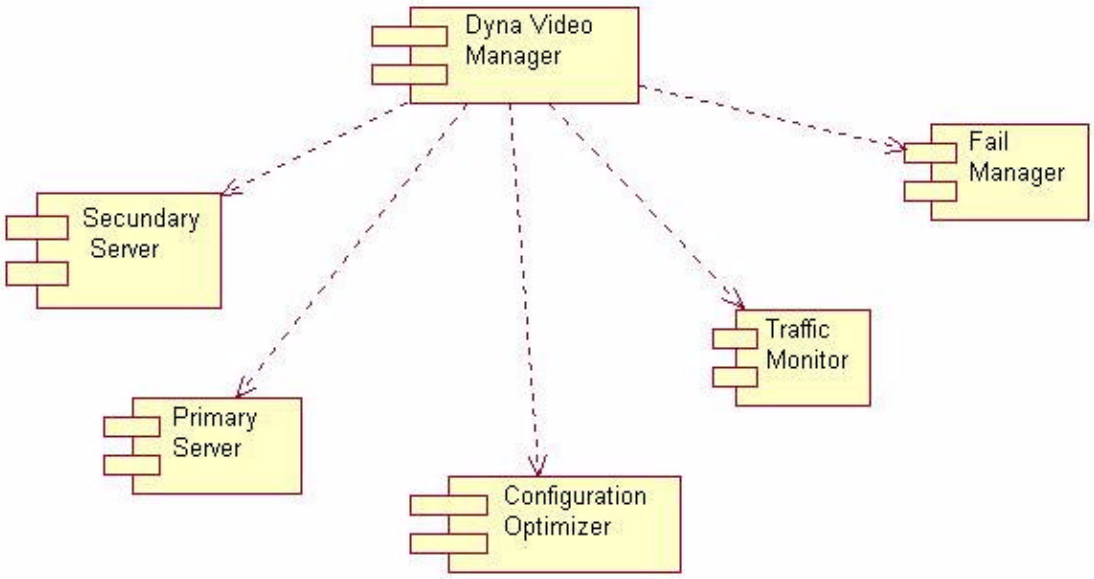

Figure 1- DynaVideo Architecture

The *DynaVideo Manager (DM)* controls the service execution. When clients request connections, this module looks for a server with capacity to support them. If it finds one, *DM* associates the client with this server. Otherwise, a *Secondary Server* is initialized to support the client. Note that initially the service policy is to try to serve the client as fast as possible, and not to find an optimized configuration to the video distribution.

The *Primary Servers (PS)* have direct access to video sources, which can be real time encoders or video file servers. PS captures a video stream from the source and transmits it to the service clients.

The *Secondary Servers (SS)* are different from PS because they do not have direct access to video sources. They receive the video stream from a PS and forward it to the service clients, acting as a reflector. The main feature of the DynaVideo *SS* is the ability of moving through the network. This way, when is necessary in order to optimize the service configuration, *DM* can determine that a certain *SS* move to a given node of the network.

The arrival of a client determines a change in the service configuration. This event activates the *Traffic Monitor (TM)* [1] to find the routes from the active servers to the client. This way, the role of *TM* is to create and to update a data structure, the *route graph*, which records the routes from the active servers to the service clients.

When the *route graph* is updated, *DM* activates the *Configuration Optimizer (CO)* module to compute an optimal configuration to the service. *CO* is executed in background, searching for a better configuration to the video distribution service, considering the current demand represented by the route graph. Once computed a configuration better than the current one, *CO* requests *DM* that:

- move clients from a server to another one;
- add or delete *Secondary Servers*;
- move *Secondary Servers* from a local to another.

The goal is to tune the service configuration to optimize the use of transmission, processing and storage resources. Figure 2 illustrates an example of a reconfiguration, showing the evolution from a

configuration with unnecessary streams traversing links and routers over the network to another where this do not occur. One *SS* was added and three clients were transferred from *PS* to *SS*. The other modification was the creation of a multicast group with three clients. This optimization takes into account that R2 do not support multicast, so the only way to eliminate the unnecessary transmissions in the route that traverses R2 is to use a *SS*.

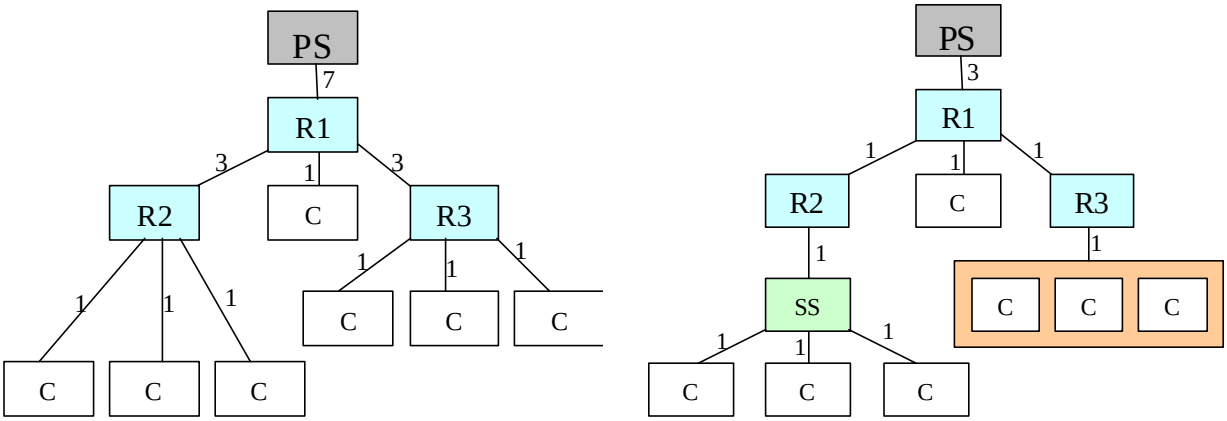

Figure 2- DynaVideo reconfiguration example

Finally, the *Fault Manager (FM)* goal is to identify failures in the service components and to arrange its replacement through a service reconfiguration. For example, if a server fails, *FM* detects this event and asks *DM* for moving its clients to another servers.

3. RESULTS

In order to test the performance of the DynaVideo service we did some experiments considering the following scenarios.

The *Primary Server (PS)* was configured in a PC Pentium MMX 300 with 128 Mb RAM memory, a 100 Mbps Ethernet card and Linux operating system.

A 4 Mbps real time MPEG-2 stream was generated and transmitted from the *streamer* to *PS*. This stream was generated by an Apollo card installed in a PC Pentium II 400 with 64 MB RAM memory and with Microsoft Windows NT 4.0 operating system.

Figure 3 shows the utilization of the link that connects PS to a IBM 8260 Switch. This figure shows two different areas. In one, PS is off and the utilization is nearly null. In the other, PS is transmitting UDP datagrams to a multicast address and the utilization is 9.5% (9.5Mbps). Since a video stream is generated in a constant rate of 4 Mbps, the reception and transmission of this stream consume 8 Mbps, in other words, 8% of the band. Considering that no other application was using the link at this moment, we have concluded that 1.5 % of the band was consumed by the overhead of transport, link and network protocols.

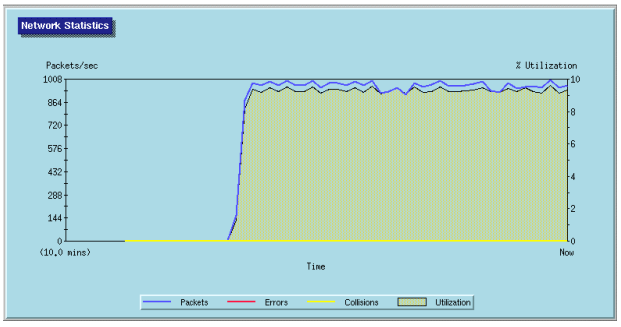

Figure 3 – Utilization of the PS network link

In another experiment, a *PS* was configured in a PC Pentium III 600 with 128 Mb RAM memory, 100 Mbps Ethernet card and Linux operating system. This server has received a 4 Mbps MPEG-1 video stream directly from the *streamer* and transmitted it to twenty (20) MTV clients using the UDP protocol. This scenario is illustrated by the first part of Figure 4. In this case, the utilization of the link that interconnects *PS* to the network reaches 100%.

To prove the feasibility of the *Secondary Server* concept, we move a *SS* to a machine at the LCC network. In this scenario (second part of Figure 4), we can serve 29 clients: PS transmitting to 19 clients and to *SS*, and *SS* transmitting to 10 clients. With this experiment, we confirm the scalability of the DynaVideo approach.

The DynaVideo service was tested in a local network with the following clients: *MTV* (MPEG-1 using TCP, UDP, HTTP and IP Multicast), *VideoLan* (MPEG-2 using UDP), *Windows Media Player* (MPEG-1 and MPEG-2 using HTTP), *RealAudio* (MPEG-1 using HTTP), and *JMF* (MPEG-1 using RTP). In all clients, the video was played with a good quality.

Figure 5 shows a MPEG-1 video being exhibited by an *MTV* client. Figure 6 shows an MPEG-2 video being exhibited by a *VideoLan* client.

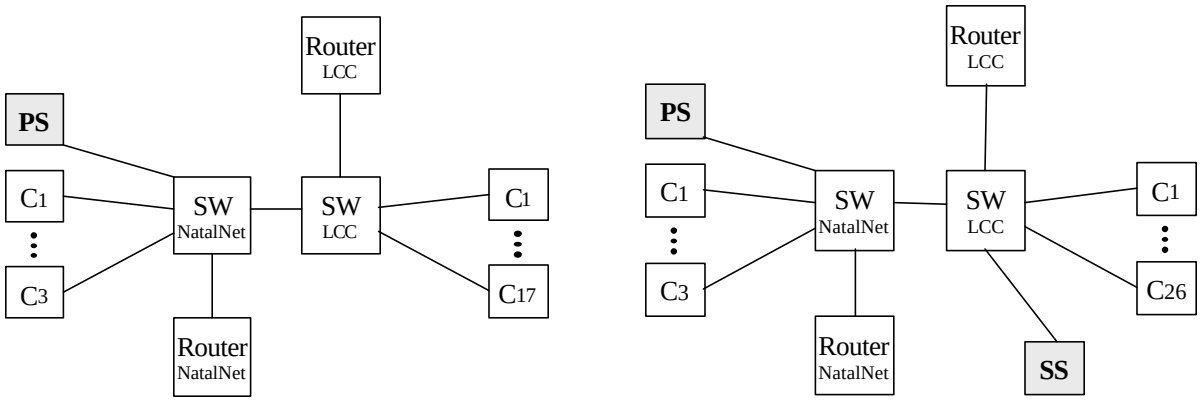

Figure 4 – DynaVideo service experimentation

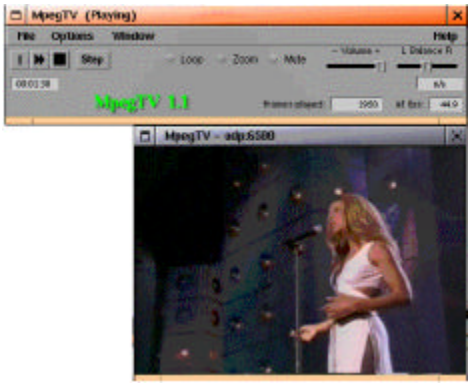

Figure 5 – MPEG-1 Video exhibited by MTV client

Figure 6 – MPEG-2 Video exhibited by VLC client

4. CONCLUSIONS

This paper has presented the architecture and implementation of a Dynamic Video Distribution Service (DynaVideo) designed for generic data communication environments, supporting different video formats and different client types.

In this paper, we have described the following DynaVideo components:

- *Dynavideo Manager* that controls the service execution;

- *Primary Server* that has direct access to video sources and whose function is to capture a video stream from the source and to transmit it to the service clients; and
- *Secondary Server* that acts like a reflector, receiving the video stream from a *Primary Server* and forwarding it to the service clients.

The main feature of the *Secondary Server* is that it can move through the network. This allows the dynamic reconfiguration of the service. The importance of using mobile agents for dynamic reconfiguration of systems is discussed in [20]. This work proposes an agent based infrastructure to distribute video, but do not deal with different video formats and third part clients as DynaVideo. Other works [22, 23] have adopted the replication idea, but do not provide support to service reconfiguration during real time video transmissions.

DynaVideo allows the transmission of MPEG streams to the MTV clients (MPEG-1 using TCP, UDP, HTTP and IP Multicast), VideoLan clients (MPEG-2 using UDP), Windows Media Player clients (MPEG-1 and MPEG-2 using HTTP), RealAudio clients (MPEG-1 using HTTP) and JMF clients (MPEG-1 using RTP). In the experiments done, the video received by these clients presented a good quality.

In the implementation, we have adopted a configuration-based approach to the *DynaVideo Manager*. This module does the integration among the other modules of DynaVideo and acts as a mediator, sending and forwarding commands to other modules. The configuration-based approach facilitates the change of one module without disturbing the others. This issue promotes the reuse and allows integration of new services in DynaVideo whenever necessary.

To support *Secondary Server* mobility, we have developed two implementations. One of the implementation uses the Aglet library, an IBM product. The other one uses aLua, an event-driven mechanism that offers support to move processes. The implementations validated the feasibility of the approach, which is a fine alternative to services dealing with unstable demands, like TV distribution.

The implementation of the Traffic Monitor module was done in another work and it is described in [1]. Currently, the Configuration Optimizer module and the Fail Manager module are under development.

5. REFERENCES

[1] Madruga, Marcos; Batista, Thais; Lemos, Guido. *"SMTA: Um Sistema para Monitoramento de Tráfego em Aplicações Multimídia*". CLEI'2000. Cidade do México – México. September, 2000.

[2] Booch, G.; Jacobson, I..; Rumbaugh, J.. *"The Unified Modeling Language for object-oriented Development*". Documentation Set Version 0.91 Addendum UML Update, September,1996.

[3] ISO/IEC International Standard 11172. *"Coding of moving pictures and associated audio for digital storage media up to about 1,5 Mbits/s*". November,1993.

[4] ISO/IEC International Standard 13818. "*Generic coding of moving pictures and associated audio information*". November, 1994.

[5] Mcgowan, John F.. "*AVI Overview*". 1998. < http://www.jmcgowan.com/avi.html>.

[6] Apple, Inc.. "*QuickTime*". <http://www.apple.com/quicktime/>

[7] RealNetworks. "*Realserver Administration Guide RealSystem G2*". <http://www.real.com/serveradminguideg2.pdf>.

[8] VideoCharger, *IBM DB2 Digital Library*. "*VideoCharger Server Key Features*". <http://www-4.ibm.com/software/data/videocharger/vcserverkey.html>. January, 2000.

[9] Microsoft, Inc., "*Windows Media Technologies*". <http://www.microsoft.com/windowsmedia/>.

[10] École Centrale Paris. "*VideoLan*". < http://www.videolan.org>.

[11] Schulzrinne, H.. Casner, S.; Frederick, R.; V. Jacobson. "*RTP: A Transport Protocol for Real-Time Applications*". RFC 1889. January, 1996.

[12] Hoffman, D.; Fernando, G.; Goyal, V.; M. Civanlar. "*RTP Payload Format for MPEG1/MPEG2 Video*". January, 1998.

[13] Deering, S.. "*Host Extensions for IP Multicasting*". STD 5, RFC 1112. August, 1989.

[14] Fielding, R.; Gettys, J.; Mogul, J.; Frystyk, H.; Masinter, L.; Leach, P.; Berners-Lee, T.. *"Hypertext Transfer Protocol -- HTTP/1.1"*. RFC 2616. June, 1999.

[15] J. Postel.. "*User Datagram Protocol (UDP)*". RFC 768. August, 1980.

[16] J. Postel.. "*Transmission Control Protocol (TCP)*". RFC 793. September, 1981.

[17] Stevens, W. R.. "*UNIX Network Programming - Networking APIs: Sockets and XTI "*. Prentice-Hall, Inc., 1998. Second Edition.

[18] Ururahy, C; Rodriguez, N. *"Alua: An event-driven communication mechanism for parallel and distributed programming*". In PDCS'99, Fort Lauderdale, Florida, 1999.

[19] Ierusalimschy, R; Figueiredo, L; Celes, W. "*Lua - an extensible extension language*". In Software: Practice and Experience, 26(6):635-652, 1996.

[20] Kon, F.; Campbell, R. et al. Dynamic Reconfiguration of Scalable Internet Systems with Mobile Agents. Technical Report, Department of Computer Science at the University of Illinois at Urbana-Champaign, 1999.

[21] Lange, D. and Oshima, M. Programming and Deploying Java Mobile Agents with Aglets. Addison Wesley. ISBN 0-201-32582-9. 2000.

[22] Parveen Kumar, L. H. Ngoh, A. L. Ananda. A Programmable Audio/Video Streaming Framework for Broadband Infrastructures. Network Storage Symposium - NetStore '99. Seattle, WA. October, 1999.

[23] Brian Noble, Ben Fleis, Minkyong Kim, Jim Zajkowski. Fluid Replication. Network Storage Symposium - NetStore '99. Seattle, WA. October, 1999.