A Programmable Audio/Video Streaming Framework for Broadband Infrastructures

Parveen Kumar
School of computing, National University of Singapore

L. H. Ngoh
SingAREN, Kent Ridge Digital Labs, Singapore

A. L. Ananda
Centre for Internet Research, National University of Singapore

Email: parveenk@comp.nus.edu.sg, lhn@krdl.org.sg, ananda@cir.nus.edu.sg

1 Introduction

The development of networks and ongoing improvements in desktop computing performance have made it possible to transfer and process a huge amount of audio/video (A/V) data in real time. This caused the increased popularity of A/V streaming applications over the Internet and some of the popular products are real media server & real player [1], video-on-demand [2, 3], etc.

Providing A/V media over the Internet involves a lot of set up cost and continuous maintenance. All these issues are handled by the end systems and thus it becomes difficult for A/V providers to manage the media servers.

Present slow Internet makes it undesirable to provide A/V services in the network infrastructure itself. As next-generation broadband network infrastructures like SingAREN (Singapore Advanced Research & Educational Network) [4, 5] and vBNS (Very-high-performance Backbone Network Service) [6] are emerging, it has become feasible to re-examine the way A/V services are realized on the network. These services can be invoked by A/V providers so that “A/V hosting” can be done by the network. In this framework the issue of scalability, maintenance, etc will be handled by the network infrastructure providers and A/V providers will
2 Framework

2.1 Overview

The core idea is to add some intelligence to the network infrastructure which facilitates scalable deployment of services [7]. Both A/V providers and consumers would be able to use these services with very little effort. This framework, if seen as a black box, can be considered as a network with some abstract services for multimedia streaming. These services involve providing audio & video for on-demand streaming, broadcast services, etc. These services can be programmed and configured for providing, receiving, storing, broadcasting A/V data etc.

If we see this framework as a white box then the network infrastructure will consist of elements distributed in the network at appropriate places. These elements will consist of A/V servers, an intelligent agent and a web server (Figure 1). The intelligent agent will control the web announcements and manage a database for the A/V servers and the data. The A/V servers will provide some generic objects with a high level interface through which they can be controlled. The web is used for A/V registration and availability announcement.

2.2 Application Scenarios

2.2.1 Stored Digital Media Hosting

The A/V providers can use the streaming services available in the network. They can transfer their digital media files to the servers (Figure 2) and specify their requirements through

Figure 1: Programmable Audio/Video Streaming Framework for Broadband Infrastructures
a registration form. The interface for all these actions will be the web and the agent will control the connection management, data storage & retrieval, etc. Once the media files are uploaded at the servers it is the agent’s responsibility to announce its availability, maintain it for a specified time span, arrange for an appropriate server to stream it on demand. The location information of A/V data is announced through high level scripts (e.g. CM scripts [8, 9]) which contains server address, file names, location at server and synchronization information. The A/V receivers willing to play A/V streams can use these scripts to request the server to start streaming digital A/V data on demand.

2.2.2 Live Media

Agent also provides services for storing live media. Live A/V providers can request for storing it through the web interface (Figure 3). The agent updates its database and arranges for space at the A/V server’s machine. For example, the server invokes RTP [10] recording process and live session (through vic and/or vat [11, 12]) can be recorded at the server’s side. In this way live streams can be stored for future broadcast.

2.2.3 MBone Broadcast of Recorded Media

Once A/V digital data is stored at the A/V server (say, in RTP format), the agent announces the server’s name, file location, etc. through the web (Figure 4). Whenever a request comes for playback of this RTP media, the agent allocates available ports and plays RTP data at the server’s side and multicasts it. Thus, receivers all over the Internet can just join the multicast session. The joining details are also announced through high level scripts (containing multicast address, ports, etc.) dynamically on the web.

2.3 Advantages of the Framework

This framework separates the A/V services from the A/V providers and put them in the network infrastructure. This is programmable hosting of audio and video by the network. This is a preferred A/V streaming framework for future broadband infrastructures such as SingAREN. The framework supports many types of A/V applications to be integrated easily with it.

Some of the advantages of such a system are as follows:
1. The service setup cost is incurred only once in network infrastructure and later it is used by all the A/V content providers and the users.

2. The A/V server would be able to serve more than one application as service programmability is provided in the framework.

3. High service setup cost for content providers can be avoided. Hardware and software management and maintenance will be performed by the network service providers.

4. There would not be a single point of failure as there would be many servers and data would be replicated when required. Thus better fault tolerance can be achieved.

5. The framework is able to handle both live and stored A/V streams.

6. One or more servers can be added or removed according to the actual user demand.

7. The framework will be able to provide audio and video according to a predefined “life-cycle”. A/V streams, which exceed their life-span, will be purged from the servers.

3 Architecture and Implementation

This section describes how we implemented all the components of the framework and how they interact in the system.

3.1 System Architecture

The framework consists of A/V servers, an intelligent agent and a web server. The agent maintains a database for all the information about A/V digital data. The whole architecture is shown in Figure 5. The agent is implemented as an RPC server written in Tcl-DP [13, 14]. It provides procedures to extract information from database and to update it on registration, request for playback, termination of playback, etc. The database is PostgreSQL [15] running on Linux machine which has a clean and abstract Tcl interface and is also scalable. The schema implemented in database contains information about the digital data stored at the A/V servers and as well as the multicast address and ports allocated for MBone broadcast.

3.2 Web Interface

Since the framework attempts to minimize the effort of A/V providers in streaming by embedding all the services in the network itself, it is important to give a clean and abstract web interface which hides all the A/V servers and other details from users. For this purpose, all the controlling is done through web interface generated dynamically by the agent. It contains WM MIME type scripts, which control the user-end tool WM.

The registration involves filling a form giving details of media like description, heading, its type as stored or live, etc. Once an A/V provider registers for a media, it is immediately announced on the web through an HTML page containing necessary WM MIME type links to upload the digital files at the A/V servers. The page also contains links to playback the A/V data by receivers after it is uploaded to A/V servers. The A/V providers who intend to use services for providing live data can just start tools like vic and vat which capture live video and audio and it is recorded at A/V servers.

3.3 Web Media Tool

We have written a tool (WM) that performs actions based on the announcements on the web in the form of WM MIME type scripts. The tool WM is a user-end tool, written in Tcl-Tk and Tcl-DP, which is used by both A/V providers and receivers. WM tools takes WM MIME type scripts, parses them and invokes tools and procedures at local or remote machines.

For providing stored A/V digital media we need to upload the local media files at the A/V
server’s machine and then the agent announces them on the web. The WM interface for this case is shown in Figure 6.

Here one or more audio and video files are selected on the local machine which need to be uploaded to the A/V server and then made available for video-on-demand. Once the files are selected, all the file names are packed in a string and an “expect” script is executed which takes this packed list as an argument and uploads these files to the A/V server. The “expect” script enables WM tool to run interactive programs in the background [16]. Once the transfer of files is complete, it is announced and the agent’s database is updated. The whole process is shown in Figure 7.

For sending live A/V data WM Tool fetches free ports from the agent, launches vic and/or vat on these ports and this starts streaming live data immediately. Meanwhile RTP record process is initiated at the A/V server and the whole data is recorded and announced for future access.

### 3.4 Database Design

The agent maintains and interacts continuously with the database to update the announcement as well to control the A/V servers. This database is implemented as PostgreSQL [15] running on Linux machine. Its Tcl interface is being used to interact and update it.

The schema is designed to store information about each digital media. Every time any A/V digital media is uploaded or recorded, it creates a unique media identification pair which consists of IP address of the sender and a unique media_id for that IP address. Media information like type, description and availability is stored and can be accessed from unique media identification pair. Another schema maintains the address of each server and storage path for the digital media. Information like ports, multicast IP address etc. is stored for each
multicast data (say RTP data).

3.5 The Agent

The agent plays the central role in the whole framework. It performs many actions like announcing the media availability on the web, giving a web interface to A/V providers, keeping information for stored digital data on VoD servers, keeping information about MBone RTP data, generating CM scripts to playback VoD streams from CMT’s VoD server and allocating free ports for MBone applications.

The agent is implemented as an RPC server, written in Tcl-DP, which interacts with the PostgreSQL database and dynamically generates HTML files and other data for announcement purposes. The interaction of the RPC servers and other elements is shown in Figure 8. The multimedia providers interact with the agent to update the database information like pre-recorded digital file details, update HTML announcement, get free multicast address and ports, etc. The A/V servers update database whenever they start or stop playing or record-
ning ftp servers for dynamic uploading of files by A/V providers, etc.

The A/V servers contains CMT VoD server for playing back pre-recorded media which enables access of data through CMPlayer [9]. These servers also maintain the digital files for each media. The servers contain an RPC daemon, which is used to start and stop RTP record process, and start RTP play process. These processes use ‘rtpdump’ and ‘rtpplay’ [17] programs respectively.

4 Related Work, Conclusions and Future Work

In this section we review the related work, conclude the paper and talk about future work.

4.1 Related Work

Most current research has provided a strong base, which can be used to develop new framework for the need of emerging broadband infrastructures. U.C Berkeley Continuous Media Toolkit contains many general purpose well-developed objects for streaming applications. Project MASH [18, 19] integrates many Mbone tools [11, 12, 20]. Project DiVA [21] which makes use of CMT is a well-distributed multimedia application to play and control remote audio/video clips using UDP and RTP protocols. These works have given us much insight on modeling, design and realization of A/V objects.

4.2 Conclusions

In this paper we presented a novel approach of a programmable framework. This framework is different from current popular streaming applications on the Internet as it adds a lot of intelligence (programmable A/V services) in the network at application level. This intelligence provides easy creation and deployment of programmable services to A/V providers. Such a framework is desired in future broadband infrastructures like SingAREN (Singapore Advanced Research & Educational Network) [4, 5] and vBNS (Very-high-performance Backbone Network Service).

We implemented a user front-end tool WM (Web Media) which is used to invoke these services made available by generic application servers in the network. These application servers include an intelligent agent with a database, a web server for announcements, and A/V servers for storing, recording and playing-back digital A/V media for video-on-demand systems and real time applications. We briefly discussed how we implemented all these elements and came up with an initial implementation of the framework which can be extended to support many other applications.

4.3 Future Work

Though the framework is intended to have a lot of features, currently only A/V hosting features are supported. Issues like scalability, caching policy, quality of service, dynamic additions of servers, multiple agents, pricing policy, have not yet been resolved. Since the first requirement of a basic framework is fulfilled, now the advanced features can be added to the framework.

References


[5] Lek-Heng Noh, Chen-Khong Tham, Francis Yeoh, A L Ananda, Tee-Hiang Cheng, Francis Lee, & Manjeet Singh,


[14] Tcl-DP - An extension to Tcl/Tk which adds TCP and IP connection management, remote procedure call (RPC), and distributed object support to Tcl/Tk. http://www2.cs.cornell.edu/zeno-Projects/tcldp/


[18] MASH - Scalable multimedia architecture for distributed multimedia collaboration in heterogeneous environments, a project at University of California, Berkeley. http://www-mash.cs.berkeley.edu/mash/


[21] Project DiVA (Distributed Video/Audio Architecture from the Federico II University) http://grid.grid.unina.it/projects/diva/