Design and Implementation of the Content Manager in a Multimedia Streaming Framework

Sungyoung Lee, Young-Rae Hong, Hyung-Il Kim, Byeong-Soo Jeong

School of Electronics and Information
Kyung Hee University
South Korea

Abstract

Content management in a streaming system is important since it allows users to access media contents more easily by efficiently handling meta information about media contents for the streaming server, web server, and clients. This paper describes a design and implementation of the content manager in Integrated Streaming Service Architecture (ISSA). The proposed content manager is not only to provide these functions efficiently, but also to provide streaming information to the media manager and the transport manager for efficient streaming. Furthermore, it supplies database interface by which ISSA can interwork with BeeHive database.

Keywords: streaming, multimedia, real-time

1. Introduction

Streaming is a technology for multimedia communications that makes it possible to deliver and display multimedia (such as audio and video) data in real-time. By using streaming technology, users can access multimedia contents immediately upon demand without waiting for the whole file to be downloaded. It is well suited to the transmission of video data that has real-time characteristics and requires high bandwidth communication. Several commercial products for multimedia streaming, such as Microsofts Window Media Technology and RealNetworks RealSystem G2 are widely used in the Internet environment.

Even though numerous streaming systems have been designed and implemented, most systems do not consider interoperability with other streaming systems and do not support diverse media formats and network interfaces. Thus, they lack flexibility, extendibility, and network transparency. For this reason, ISSA (Integrated Streaming Service Architecture) has been proposed to overcome the limitations of existing streaming systems while providing diverse audio and video CODEC and the ability to run adaptively on different operating systems and networks [1,3].

Since ISSA uses standard real-time communication protocols, such as RTSP (Real-Time Streaming Protocol) [6] and RTP (Real-Time Transport Protocol) [7], it can provide openness. It also provides the function of media data storage by interoperating with a real-time multimedia database system (BeeHive). ISSA is an integrated streaming service architecture that provides a group of library functions by which streaming applications such as VOD system and real-time broadcasting system can be easily developed. ISSA consists of several modules, including Session Manager that

* This work was partly supported by International Collaborative Research Program (TRP-9803-6) sponsored by Ministry of Information and Communication of Korea

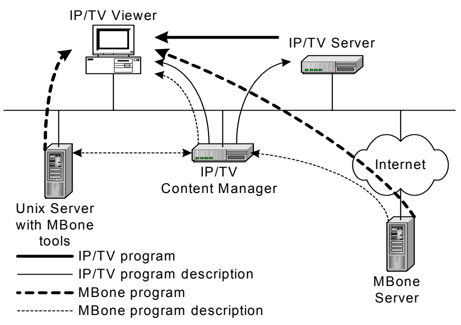
controls the session of streaming applications; Transport Manager that supports RTP, TCP, and UDP for media data transmission; Media Manager that manages diverse media types; Gateway that provides interoperability with other streaming frameworks, such as CORBA A/V streaming service [8,9]; and Content Manager that provides database connections (with BeeHive and Oracle) and transaction processing.

Generally, streaming applications deal with large amounts of media content that have diverse formats and categorization. Thus, in order that users may easily access media contents when they want, the streaming framework should manage information about media contents and efficiently provide them to the streaming server, web server, and clients respectively. Media contents can be classified into three categories, on-demand media contents streamed by user demand, live media contents streamed directly from video camera or microphone, and scheduled media contents streamed during the predefined time interval. Media contents are often stored in file objects or database objects and need different interfaces to handle them. For efficient streaming, the system needs additional information like compression type, transmission type (multicast/unicast), and QoS parameters; and the Content Manager should efficiently provide such information to streaming the server. In this paper, we describe the design and implementation of the Content Manager in ISSA.

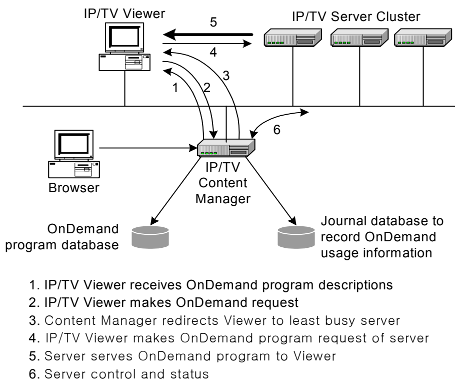
This paper proceeds as follows. Section 2 provides an overview of existing content management implementation. In Section 3, we describe the system architecture and functions of our integrated streaming framework (ISSA). Section 4 describes the design and implementation of ISSA Content Manager, and Section 5 explains how to interoperate with BeeHive, a real-time multimedia database through a database connector controlled by the Content Manager. Finally, we state our conclusions in Section 6.

2. Related Work

Currently, many streaming systems have been developed and some have been commercially successful. However, most systems have focused on the streaming function itself and there has not been much attention to content management functions. IP/TV [10], B · media [12], and Sonera Live [11] are a few examples that consider content management as an important component in a streaming system. Here, we will describe IP/TV briefly to compare it with our content manager design.



(a) IP/TV Components – Scheduled Programs


IP/TV Viewer
IP/TV Server Cluster
5
4
3
1
2
6
Browser
IP/TV
Content
Manager
OnDemand
program database
Journal database to
record OnDemand
usage information
1. IP/TV Viewer receives OnDemand program descriptions
2. IP/TV Viewer makes OnDemand request
3. Content Manager redirects Viewer to least busy server
4. IP/TV Viewer makes OnDemand program request of server
5. Server serves OnDemand program to Viewer
6. Server control and status

(b) IP/TV Components – OnDemand Programs

Figure 2. Architecture of Cisco's IP/TV

IP/TV is a streaming system that can transmit diverse media formats, e.g., MPEG1, MPEG2, MPEG4, and H.261 on IP-based LAN and WAN (Internet). IP/TV consists of three parts; the viewer that plays media stream, the content manager that provides media meta information to the viewer, and the server that transmits media contents. The content manager of IP/TV handles two kinds of media, on-demand media and scheduled media that can be provided from live media and stored media data.

Streaming architecture for scheduled media and on-demand media is described in Figure 1. Lists of scheduled media are controlled by the content manager. Scheduled media does not produce an excess of network traffic since it uses the multicast protocol. By contrast, on-demand media requires more network bandwidth since it establishes an individual channel to every client.

3. Integrated Streaming Framework Architecture

In this section, we describe the architecture of our streaming system that has a content management function. Our streaming framework aims to support diverse audio and video CODEC and adaptively run on different operating systems and network environments. As shown in Figure 2, the overall system architecture of our streaming framework consists of two parts, Streaming Applications and ISSA (Integrated Streaming Service Architecture). The latter is comprised of a Content Manager, a Control Manager, a Session Manager, a Transport Manager, a Resource Manager, a Media Manager, Database Connector, and Gateway Modules. MOA (Media Object Architecture) resides between them. ISSA performs the essential functions for multimedia streaming services and MOA provides an API (Application Programming Interface) as object-oriented wrapper classes that can be easily used by streaming application developers. VOD (Video On Demand) server and client programs developed by using MOA can be one example of the Streaming Applications layer.

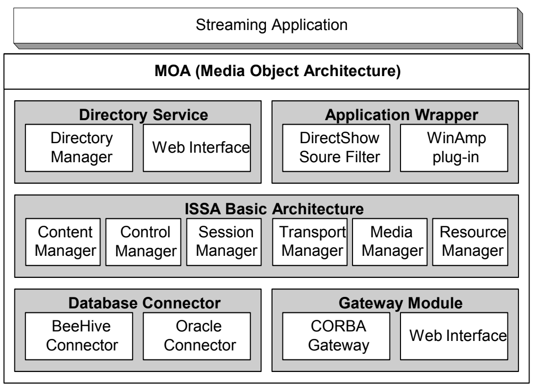


Figure 2. Architecture of Integrated Streaming Framework

The Session Manager performs session control functions between streaming application programs, such as session creation/destruction. It is also responsible for streaming media control, such as play-back and stop. These functions are provided by implementing RTSP (Real-Time Streaming Protocol) developed by IETF (RFC 2326) in 1998. The Control Manager supports the Session Manager by informing several control information from users. The Media Manager provides several functions related with media data processing, media file I/O through file systems or DBMS, A/V CODEC to encode/decode diverse media data, and control of audio/video devices. The Content Manager is a component for managing media data such as audio and video. It facilitates interaction with DBMS in order to retrieve and store media data. A Database Connector is required to interact with different DBMS. The Resource Manager provides QoS related functions, including QoS specifications, mapping, monitoring, memory buffer management, and thread scheduling. The Gateway Module is a component of ISSA that is intended to allow interworking with other streaming frameworks such as the CORBA A/V Streaming Service and to support streaming service in the WWW (World Wide Web) environment. The Transport Manager performs packetization and depacketization of multimedia data represented by diverse media formats and is responsible for transmitting such packetized media data in real time through the diverse network environments. These functions are accomplished by implementing RTP (Real-time Transport Protocol) that was developed by IETF (Internet Engineering Task Force) in 1996 (RFC 1889) as an application level protocol for multimedia transport.

4. Design of Content Manager

Content Manager provides the functions that generate meta information about media contents (such as media title, type, location, format, and security) and transmit such information to clients in the form of meta file and URL. Content Manager handles media contents differently according to content type, e.g., on-demand content, live content, and scheduled content. On-demand content is streamed by user demand and stored in a system as file objects or database objects. Live media contents are streamed immediately after acquiring data from video camera or microphone. Thus, VCR functions, such as fast forward and rewind are not possible for live media streaming. Scheduled media content is streamed during the predefined time interval. Live media content and scheduled media content can be streamed by multicast communication protocol. On the other hand, on-demand media content is streamed by unicast protocol.

Another function of Content Manager is to manage the database connector that controls multimedia database access and performs database transactions. The database connector provides main functions that extract media data and meta information from multimedia databases and supports the BeeeHive connector for interoperating with a BeeHive database. As shown in Figure 3, Content Manager performs its functions by communicating with the media source that is responsible for sending/receiving media data and the database connector that controls database interface. That is, Content Manager initializes the corresponding media source by receiving parameter values, i.e., file name and database id from the media source, and by communicating with the database connector performs information retrieval, update and several transactions for streaming service.

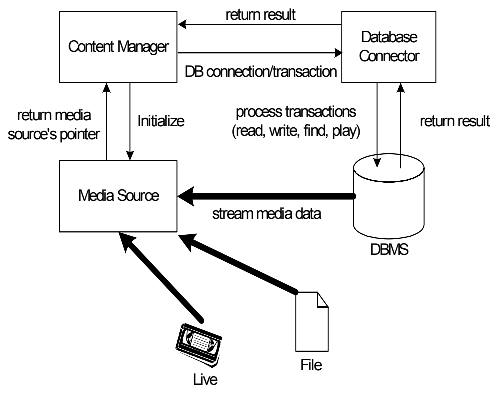


Figure 3. The Structure of Content Manager

Figure 4 describes the relationship between Content Manager, media source, and database connector by a class diagram of UML (Unified Modeling Language). As can be seen in Figure 4, *cmContentManager* generates *cmBHConnector* and *cmBHConnector* requests transaction service to a database server through API of database interface.

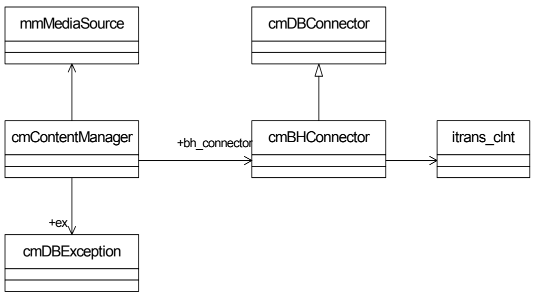


Figure 4. Class Diagram for Content Manager

5. Implementation of Content Manager

The Content Manager largely consists of two parts: *media content management,* which provides the mechanism to extract meta information of media contents, efficiently show them to users and perform initialization process for media streaming; and *database interface,* which provides interworking with a BeeHive database and manages database transactions that read media contents from the database.

Table 1. Audio/Video Meta Information

media_type == Audio	media_type == Video

mt=audio
title=(title)
author=(author)
en=(encoding method)
br=(bit rate)
bps=(bit per sample)
ch=(number of channels)
ba=(frame byte align)
fs=(frame size)
sr=(sampling rate)
	mt=video
	title=(title)
	author=(author)
	en=(encoding method)
	bps=(bit rate)
	fps=(frame per second)
	h=(frame height)
	w=(frame width)

In order to extract meta information about media content, the Content Manager provides two APIs as follows, which can be called with the parameters such as URL or media source reference (*mmMediaSource*).

```
issaString
   GetMediaDesc(issaString &path);
issaString
   GetMediaDesc(mmMediaSource &src);
```

After execution of these API functions, meta information (as shown in Table 1) is returned as *issaString*. When users request specific media after reviewing meta information, the Content Manager performs an initialization process according to the type of requested media. The initialization process is performed by the following four functions and returns *mmMediaSource* reference value.

```
mmMediaSource *
  MediaInit(issaString &path, int type);
mmMediaSource *
FileMediaInit(issaString &abs_path);
mmMediaSource *
  DBMediaInit(issaString &id);
mmMediaSource *
  LiveMediaInit(cmLiveContent &live);
```

In order to utilize BeeHive transactions in an ISSA environment, we add a BeeHive transaction interface to the Content Manager of ISSA. The Content Manager of ISSA provides a database interface with BeeHive and processes media source requests from ISSA clients. We use RPC mechanism for interaction with BeeHive and devise RTTP (Real-Time Transaction Protocol) for transaction processing between ISSA client and server. RTTP is a protocol that handles transaction requests for BeeHive in ISSA environments. But it is not directly related with BeeHive.

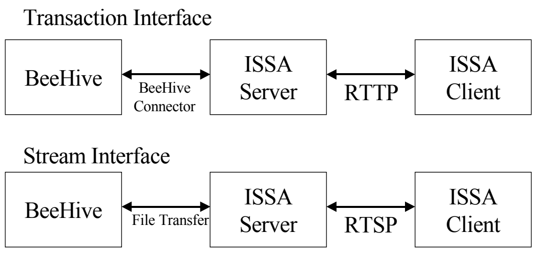


Figure 5. Transaction and Stream Interfaces between ISSA and BeeHive

Interoperable interface model between ISSA and BeeHive consists of Transaction Interface and Streaming Interface (as shown in Figure 5). Transaction Interface deals with a series of functions which the ISSA client requests for transactions of BeeHive. Streaming Interface provides a mechanism to stream media between ISSA client and server. Transaction Interface is composed of a GUI part that handles transactions requested from an end-user, a RTTP (Real-Time Transaction Protocol) part which deals with transaction processing between ISSA client and server, and an RPC part which is an interface between ISSA server and BeeHive. Streaming Interface consists of a part that relates with the media display of thw ISSA client, a part that controls RTSP and RTP session between ISSA client and server, and a part that handles media stream transmission between ISSA server and BeeHive. All functions previously mentioned are contained in the BeeHive Connector that resides in the Content Manager of ISSA server.

6. Conclusion

This paper describes a design and implementation of Content Manager, which is a part of our integrated streaming framework that integrates heterogeneous environments and works easily together with other streaming systems. Content Manager deals with diverse media formats and provides efficient management for meta information about media data. It also provides a user-friendly interface for media data access by generating a meta file or URL. Currently, the database interface of Content Manager supports BeeHive database. We plan to extend this database interface to industry standard DBMS like Oracle and Informix in order to support a wider range of existing media contents.

Reference

[1] Chan-Gyun Jeong, et. al., "Design for an Integrated Streaming Framework", *Department of Computer Science, University of Virginia,* Technical Report, CS-99-30, November, 1999.

[2] HyungIl Kim, Sungyoung Lee, "Design of Media Object for Multimedia QoS, *'98 Korean Information Science Society, Spring Conference*, pp.699-701, April 1998.

[3] ChanGyun Jeong et. al., "Design of Distributed Streaming System", *'99 Korean Multimedia Society, Fall Conference*, pp.338-343, Nov. 1999.

[4] J. Stankovic, S. Son and J. Liebeherr, "BeeHive: Global Multimedia Database Support for Dependable, Real-Time Applications", In *Proc. of Second Workshop on Active Real-Time Databases*, Lake Como, Italy, September 1997.

[5] J. Stankovic and S. H. Son, "An Architecture and Object Model for Distributed Object-Oriented Real-Time Databases," *Journal on Computer Systems Science and Engineering, Special Issue on Object-Oriented Real-Time Distributed Systems*, vol. 14, no. 4, pp 251-259, July 1999.

[6] H. Schulzrinne, A. Rao, and R. Lanphier, “Real-Time Streaming Protocol (RTSP)”, *IETF RFC 2326*, April 1998.

[7] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, “RTP: A Transport Protocol for Real-Time Applications”,

IETF RFC 1889, January 1996.

[8] Object Management Group, Control and Management of A/V Streams specification, *OMG Document telecom/97-05-07 ed.*, October 1997.

[9] S. Mungee, N. Surendran, and D. C. Schmidt, "The Design and Performance of a CORBA Audio/Video Streaming Service", In *Proc. of the 32st Hawaii International Conference on System Systems(HICSS)*, Hawaii, January, 1999.

[10] Cisco, IP/TV Content Manager: User Guide, http://www.cisco.com/univercd/cc/td/doc/product/software/iptv30.

[11] Sonera, Sonera Live Content Manager, http://www.live.sonera.com/eng/tools/production_software.html.

[12] Banta Integrated Media, B·Media Content Manager, http://centrus.com/pr_media_body_2.html.

[13] C. Aurrecoechea, A. T. Campbell, and L. Hauw, "A Survey of QoS Architectures", *ACM/Springer Verlag Multimedia Systems Journal , Special Issue on QoS Architecture*, Vol. 6 No. 3, pp. 138-151, May 1998.

[14] S. N. Bhatti and G. Knight, "Enabling QoS adaptation decisions for Internet applications", *Journal of Computer Networks*, Vol. 31, No. 7, pp. 669-692, March 1999.

[15] Microsoft Corps., Introduction to DirectShow, http://www.microsoft.com/directx/dxm/help/ds/default.html.