A System Architecture for Managing Mobile Streaming Media Services

Sumit Roy, Michele Covell, John Ankcorn, and Susie Wee
Streaming Media Systems Group
Hewlett-Packard Laboratories
Palo Alto, CA 94304

Takeshi Yoshimura
Multimedia Laboratories
NTT DoCoMo
Yokosuka, Japan

Abstract

A mobile streaming media content delivery network (MSM-CDN) overlay system provides a scalable method for delivering media streams to a large number of clients. With the availability of such a streaming infrastucture, it becomes possible to implement enhanced media services. For example, the wide range and variability of network conditions, as well as processing and display capabilities of these devices will effectively require media streams to be adapted in the network. Each streaming session needs to be tailored to these changing environments in a practical and scalable manner. Media transcoding services can be performed by the servers of the MSM-CDN overlay, providing this flexibility. Due to the computational and bandwidth requirements of real-time video transcoding, these services require management of the placement of these tasks on the most appropriate servers, to make best use of the distributed resources available within the network.

In this paper, we address the media service assignment problem using the notion of service-location management (SLM). *An effective load balancing system requires appropriate resource monitoring. We propose alternate SLM resource monitoring schemes. Using media transcoding as a representative service, we compare the performance of these schemes on an MSM-CDN testbed. We present our conclusions on which of these alternate implementations is both most reliable and most extensible to serve a large numbers of mobile client requests.*

1. Introduction

Typically, people learn of various content sites (e.g., a video-based movie page) based on their web-browsing experiences from their desktop or laptop machines, since these devices are better able to support the input (typing various URLs or search queries) and output (reliable, high-bandwidth connections) requirements of random browsing on the net. Believing in the promise of high-bandwidth wireless access, these web users may try to connect to the same sites using their PDAs or video-enabled cell phones. This wider access results in the need for the content provider to support a wide range of different bit-rates (according to the bandwidth of the connection), video-frame rates (according to the CPU power available at the client, which itself varies dynamically according to power-management strategies), and video-frame sizes (according to the display size available at the client). Also, as seen by 3GPP [1] providers in Japan, supporting mobile access from light-weight clients requires servers to maintain and update state variables for large numbers of sessions. For example, "flash crowds" of thousands of mobile users are often seen in Tokyo during the evening transition from the downtown office area to the restaurant district.

The problem is, therefore, two-fold: one is providing video and audio content in a format that is *dynamically* tailored to the client's capabilities and the other is *dynamically* distributing the support for that streaming process to avoid unnecessary congestion and the resulting degradation in quality. Both parts of the solution should be done dynamically, since the factors on which they depend are themselves often changing quickly.

In this paper, we contend that, unless media services are integrated and managed in a distributed fashion within a streaming *content-delivery network (CDN)* infrastructure, the potential of wireless devices for *mobile streaming media (MSM)* will not be realized. In Section 2, we discuss background work on providing reliable, scalable media streaming across the existing network infrastructure in support of wireless and mobile streaming clients. Section 3 then outlines an approach to managed placement of transcoding services by dynamic monitoring of the distributed resources available within the CDN. Trade-offs between resource monitoring approaches are discussed in Section 4. Section 5 describes our current implementation of and results from a *service location manager (SLM)* within our MSM-CDN testbed. Section 6 lists some related work in distributed media processing. We summarize and provide directions for future work in Section 7.

2. Adaptive Streaming Content Delivery to Mobile Clients

The basic components of a mobile streaming media system include streaming servers for stored media content, live streaming servers, and streaming media clients. To deliver video clips to a large number of users in a scalable fashion, one can use an MSM-CDN overlay on the existing network. It contains streaming edge (or surrogate) servers and management servers. The streaming edge servers have functionalities of content distribution and caching [16], streaming, resource monitoring, resource management, and signaling. They can also perform media-service functions such as live-media adaptation. The management servers distribute content and assign media sessions based on client location and current system and network load, in other words they assign client requested sessions to the *best available* edge servers.

A MSM-CDN system should help support a wide variety of clients in terms of display and decode capabilities. A "traditional" way to do this is to store multiple copies of the source material on the content server and to then select which copy to send according to some initial negotiation with the client (Figure 1). However, the reliability and bandwidth of a connection from various parts of the network to the client will change during a streaming session as the client moves physical location and as streaming sessions from other clients begin and end within the shared wireless environment. This suggests that this negotiation needs to span a wider range of options than is easily provided by multiple stored encodings and that the negotiation process should be dynamically updated as the network conditions change. Since real-time video transcoding is both practical and affordable on today's network-server machines, this wide range of needs in video rates, sizes, and bandwidths can be met by embedding *transcoding* services within the network.

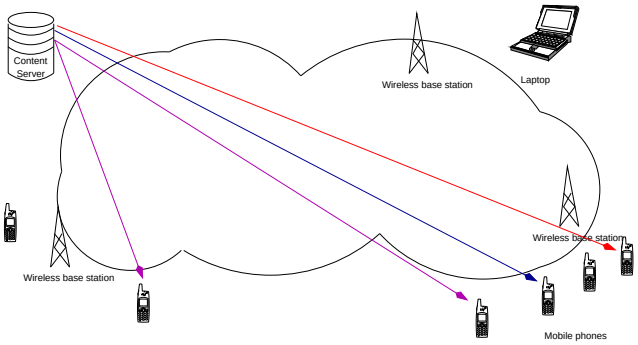

Figure 1. Static encoding of source material

Providing this real-time, low-latency video transcoding is one of the key functions of the edge servers [2, 7]. The transcoding process can adapt the compressed video stream

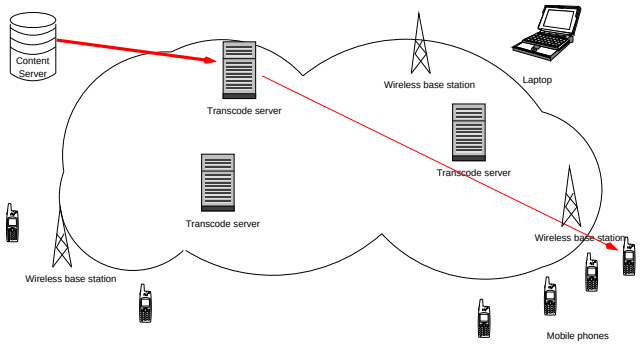

Figure 2. Static transcoding server assignment

to the client display. It can also use RTCP-based feedback to dynamically adjust the bit rate within the stream to the changing bandwidth conditions experienced by the client device. These real-time video transcoding service can now be provided on standard desktop or server machines, due to the use of compressed-domain processing [14, 15, 10].

These new compressed-domain transcoding techniques greatly reduce the computational cost of each individual transcoding session, thereby making mobile streaming both practical and affordable. However, as with content management, the size and duration of the video transcoding streams and the computational demands associated with modifying those streams require careful management. In the presence of thousands or millions of mobile clients, computationally powerful servers must be be dispersed throughout the infrastructure so that transcoding could be provided as a distributed edge service.

One way to provide the transcoding services called for by the previous discussion would be for each content server to provide static redirection of the client browsers to a fixed transcoding server. This type of static redirection is well explored in terms of content delivery: redirections to local "mirror" sites are done routinely in today's web environment. A similar process could be used to redirect video clients to a fixed transcoding-enabled server (Figure 2). The disadvantage of this static redirection is that it does not take into account any of the dynamics of the network and server loads. The bandwidth and computational load available at various servers will change according to changing requirements of the client and of newly added or dropped clients. Thus, the placement of the transcoding processes on the different servers should itself be dynamic and, preferably, adjusted as the client processor changes physical location. Finally, for ease of use by the mobile web-browsing public, all of these dynamic decisions should be hidden and automatic.

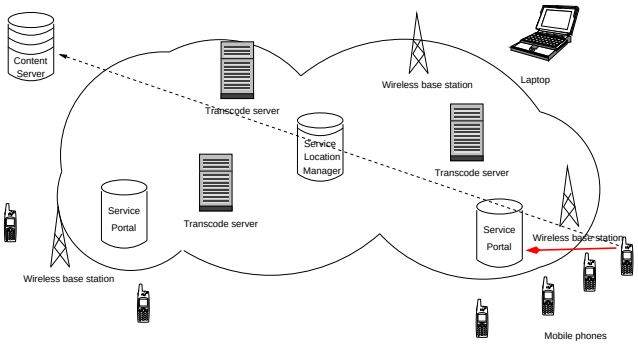

Figure 3. Initial contact from PDA to portal

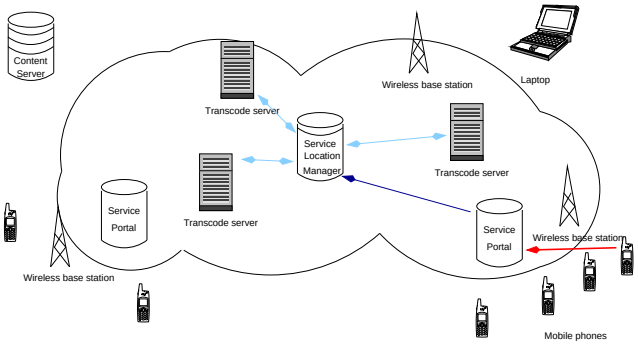

Figure 4. SLM contact with transcode

3. Service Location Management (SLM)

The idea behind dynamic service location management is to provide the flexibility required in a mobile streaming environment without requiring the mobile user to change the initial contact site. The general system instead provides some number of well-published portal sites. These portals are the first point of contact for the mobile user and accept redirection to an original content site (Figure 3). All subsequent redirection is done in a client-transparent manner, using dynamic SMIL rewriting [16].

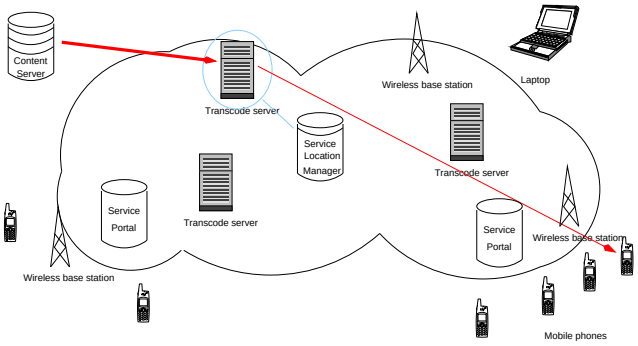

Figure 5. SLM redirection of first client

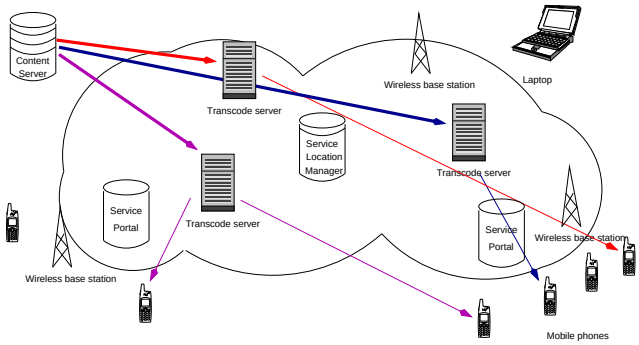

Figure 6. Placement of subsequent tasks

Once contacted by a client, the portal site contacts the service location manager (SLM). The SLM then determines what type of transcoding is needed to serve the requested material to the given client and examines the status of the transcoding-enabled servers that are (partially or completely) under its control (Figure 4). That status is summarized in terms of available cycles and available memory on each of the server machines. Additional status indicators include the expected bandwidth and reliability of connections from each of the transcode servers to the content provider (or the nearest mirror site) and to the streaming client. Based on the collected status information, the SLM dynamically generates a SMIL file, redirecting the client to the appropriate transcoding server by embedding its URL, along with the negotiated transcoding parameters, in that newly generated SMIL response (Figure 5). The 3GPP or ISMA [5] compliant streaming client then parses the rewritten SMIL file to set up the appropriate streaming session. Thus the whole processing is transparent to the end user. Subsequent content requests from other clients that require transcoding are also distributed according to the newly current network and computational resources (Figure 6).

4. Resource Monitoring for Dynamic Service Location

In the above description, the SLM examines the status of each of the servers that is under its control to determine how best to dispatch the transcoding task required by the current client request. There are various ways that this examination can be completed. This section details some of those options and points out their strengths and weaknesses.

4.1. Basic “poll-based” monitoring

One approach to monitoring the status of transcoding-enabled servers under the control of the SLM is for the process to be “poll-based.” In this approach, whenever the

SLM gets a new client request for transcoding, it actively contacts each of the servers that may have adequate resources (in terms of number and clock speeds of its CPUs, its installed memory, and its best-case network bandwidth). In response to this "resource poll", each transcoding server provides a description of its currently available resources. This includes the number of free compute cycles and the amount of free memory at a given point in time. Ideally, it would also include some estimate of the free network bandwidth to the content server and to the client. The SLM collects this information and dispatches the requested transcoding task to whichever server provides the best combination of free network-bandwidth, computational, and memory resources.

This "poll-based" approach has the advantage of providing up-to-date snapshots of the free server resources. It also provides a clear indication of when a transcoding machine is out of service, either due to a network or machine failure. On the other hand, poll-based resource monitoring has serious limitations in terms of extensibility. As the number of client requests and the number of monitored servers grows, the number of polling requests grows as their product. Since the number of monitored servers will tend to grow in direct proportion to the number of client requests for services, the number of polling requests effectively grows as the square of the number of clients.

4.2. Basic "table-based" monitoring

An alternative to polling is for resource information to be "pushed" from the transcoding server machines to the monitoring SLM. In this approach, updates are provided on a periodic basis by a *service-location supervisor (SLS)*, a lightweight background daemon running on each transcoding-server machine, such as provided by system and network management software. On each client request, the SLM accesses the free-resource database created from collecting (and dating) the SLS-provided information. This reduces the connection requirements incurred by resource monitoring from a quadratic dependence to a linear dependence on the number of server machines.

Furthermore, monitoring and "re-launch" capabilities could be included in the SLM itself: a simple SLM daemon would monitor the timestamps of the latest SLS database refreshes and attempt to contact SLS machines that are out-of-touch for more than some preset time interval. Presumably, a fair portion of these contact attempts will fail, due to an ongoing network or server failure. However, since these attempts to relaunch SLS contact would be done asynchronously, they will not affect the response time of the SLM to client requests.

Table-based monitoring has the disadvantage of relying on resource information that is more out of date than direct poll-based results. This weakness is addressed by the next proposed approach to resource monitoring.

4.3. Enhanced "table-based" monitoring

The table-based monitoring approach can be modified to reduce the drawback of out-of-date information. This is done by having the SLM maintain a short-term record of the machines to which it has dispatched recent client tasks. The SLM then adjusts its prediction of what resources will be available for new jobs accordingly. For example, when a transcoding task was dispatched to a server less than 1 minute before the resource statistics where last transmitted from that server, the resource record of that server would be lowered by a resource budget requested by that previously dispatched transcoding job.

If some of the transcoder servers are under the purview of more than one SLM (that is, if more than one of a distributed set of SLM machines is allowed to redirect transcoding requests to that transcoding server), then each SLM should also propagate information about dispatched jobs to the SLS daemon on that server as soon as the dispatch occurs. That way, the SLS daemon can retransmit all dispatch notifications on to the other SLM processors, thereby minimizing the number of times that server computational or network resources are over-booked due to crossing dispatches from the different SLMs.

5. Testbed Results

The service location management architecture presented in this paper was designed to integrate media services with a mobile streaming media delivery system. A mobile streaming media (MSM) testbed was designed, developed, and implemented to demonstrate these capabilities. The MSM testbed consists of a number of stored-content and live-content streaming servers and streaming media clients. The servers are located at Hewlett-Packard Laboratories in Palo Alto, as well as in the NTT DoCoMo Laboratories in Yokosuka, Japan. Streaming edge servers and management servers together form an adaptive MSM-CDN. The streaming edge servers provide support for content distribution and caching, streaming, resource monitoring, resource management, and signaling. In addition, they perform media service functions such as live-stream splitting (or application-layer multicast of media streaming sessions) and real-time media transcoding of MPEG-4 video streams.

The streaming servers, clients, and edge servers are all compliant with 3GPP standards, and therefore use the Session Description Protocol (SDP) [4], Real Time Streaming Protocol (RTSP) [13], and Realtime Transport Protocol (RTP) [12] and support the MPEG-4 [8] video and AMR audio media standards. The streaming edge servers and

management servers use the Simple Object Access Protocol (SOAP) [3] for signaling.

The service location manager (SLM) assigns client-requested streaming/transcoding sessions to "best available" streaming edge servers based on network and system resource usage. As proposed in Section 4, the SLM collects statistics on a set of streaming edge servers, analyzes those statistics to choose the best available edge server, and conveys the chosen edge server in response to client requests. The SLM uses SOAP/XML signaling to gather resource usage statistics from edge servers and to dynamically convey the chosen edge server to the requesting client.

Each of the three proposed approaches to SLM resource monitoring was implemented and tested in our MSM-CDN testbed. The poll-based monitoring occasionally resulted in complete streaming failure. This would happen when the response time-out period on the mobile client was set too low, so that the SLM did not have adequate time to collect all of the poll responses, process them, and provide the dynamically generated SMIL responses before the client gave up. These too-slow responses would typically happen when one or more of the transcoding server machines was off the network: in these cases, the SLM waited for a standard SOAP timeout period before disregarding that server as a potential transcoding platform for the client. The delays associated with poll-based monitoring also do not gracefully support scaling of the network: as the number of monitored transcoder server machines increases, the delay associated with polling increases proportionally.

The basic table-based monitoring did not suffer from this timed-out failure mode. However, it often resulted in suboptimal load balancing. This occurred when client requests came in quick succession. Even if the SLS on the transcoder server was modified to update free-resource information contained in the SLM database whenever it saw a new local transcoding task, this suboptimal load balancing still occurred. Sometimes, this suboptimal task assignment was due to the latency in the free-resource statistics response to a newly instantiated task. More often, the suboptimal task assignment was due to new client requests arriving after the SLM dispatched a transcoding task to a particular server (by transmitting the dynamic SMIL file to the client) but before that earlier client actually established that transcoding task on the selected server (by transmitting a RTSP SETUP request).

The enhanced table-based monitoring avoided both the timed-out failures seen with the poll-based monitoring and the interleaved-request mistakes seen with the basic table-based monitoring.

6. Related Work

The *Degas* system allows user defined media processing using programmable media gateways [9]. Programs, called *deglets*, can be uploaded into the gateways using a declarative programming model. The *Degas* system requires a special client to interact with the media gateways. On the other hand, the SLM system described in this paper is completely transparent to a 3GPP compliant client. The *Degas* system tries to locate gateways optimally with respect to network bandwidth utilization and can dynamically migrate processing tasks when necessary. However resource management was not implemented. The system uses a multimedia software library to optimize code at the media gateway.

A content services network (CSN) was proposed in [7]. Video segmentation with keyframe extraction was used as a sample infrastructure service. Similar to our architecture, the CSN leverages an existing CDN to add computation (i.e., processing and transcoding) as an infrastructure service. Services Distribution and Management (SDM) servers are used to maintain information about the services in the network and a history of server loads and client demographics. Redirection servers are placed at the network edge to send the processing request to an application proxy server. The proposed CSN uses DNS redirection to send the request to the nearest application proxy. In our architecture, this function is performed completely at the application level by dynamic SMIL rewriting. This eliminates the need for DNS-redirection capabilities from the infrastructure.

7. Summary

In summary, we believe these media services are needed to support a rapidly expanding and highly dynamic set of display, processor, and bandwidth restrictions presented by mobile devices as they move from place to place, as they start and stop background tasks, and as they adjust their processor and display parameters to allow for various power management strategies. The SLM solution outlined in this paper can effectively address the problem of load balancing a CPU intensive media processing task across multiple servers in the network. When a client accesses a well known portal site, the service location manager dynamically routes the request to the least loaded server. Furthermore, the transcoded streams are provided in a 3GPP compliant client-transparent manner from appropriate servers in the network.

In future work, we plan to extend this architecture to trigger application level hand-off of media transcoding sessions for mobile clients as outlined in [6, 11]. The SLM architecture is well suited to determine transcoding servers that are close to the new client position. The ability to perform

mid-session hand-off allows load balancing at a much finer granularity than outlined in this paper.

References

[1] 3GPP TS 26.233/234. Transparent End-to-End Packet Switching Streaming Services (PSS). ftp://ftp/3gpp.org/Specs/2001-03/Rel-4/26_series/.

[2] E. Amir, S. McCanne, and R. Katz. An Active Service Framework and its Application to Real-time Multimedia Transcoding. In *Proceedings of SIGCOMM'98*, Vancouver, B.C., 1998.

[3] D. Box, D. Ehnebuske, G. Kakivaya, A. Layman, N. Mendelsohn, H. F. Nielsen, S. Thatte, and D. Winer. Simple Object Access Protocol (SOAP) 1.1. http://www.w3.org/TR/SOAP, May 2000. W3C Note.

[4] M. Handley and V. Jacobson. SDP: Session Description Protocol. RFC 2327, April 1998.

[5] ISMA. Internet Streaming Media Alliance Implementation Specification, August 2001.

[6] R. Karrer and T. Gross. Dynamic Handoff of Multimedia Streams. In *Proceedings of the Workshop on Network and System Support for Digital Audio and Video*, pages 125 – 133, Port Jefferson, NY, June 2001.

[7] W.-Y. Ma, B. Shen, and J. Brassil. Content Services Network: The Architecture and Protocols. In *Proceedings of the 6th International Web Content Caching and Distribution Workshop*, Boston, MA, 2001.

[8] MPEG-4 Industry Forum. http://www.m4if.org.

[9] W. T. Ooi, R. van Renesse, and B. Smith. Design and Implementation of Programmable Media Gateways. In *Proceedings of the Workshop on Network and System Support for Digital Audio and Video*, Chapel Hill, NC, June 2000.

[10] S. Roy and B. Shen. Implementation of an Algorithm for Fast Down-Scale Transcoding of Compressed Video on the Itanium. In *Proceedings of the 3rd Workshop on Media and Streaming Processors*, pages 119 – 126, Austin, TX, December 2001.

[11] S. Roy, B. Shen, V. Sundaram, and R. Kumar. Application Level Hand-off Support for Mobile Media Transcoding Sessions. In *Proceedings of the Workshop on Network and System Support for Digital Audio and Video*, Miami, Florida, USA., May 12-14 2002.

[12] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobsen. RTP: A Transport Protocol for Real-Time Applications. http://www.ietf.org/rfc/rfc1889.txt, January 1996.

[13] H. Schulzrinne, A. Rao, and R. Lanphier. RFC 2326: Real time streaming protocol (RTSP), Apr. 1998.

[14] H. Sun, W. Kwok, and J. Zdepski. Architectures for MPEG compressed bitstream scaling. *IEEE Transactions on Circuits Systems and Video Technology*, April 1996.

[15] S. J. Wee, J. G. Apostolopoulos, and N. Feamster. Field-to-frame transcoding with temporal and spatial downsampling. In *Proceedings of the IEEE International Conference on Image Processing*, Kobe, Japan, October 1999.

[16] T. Yoshimura, Y. Yonemoto, T. Ohya, M. Etoh, and S. Wee. Mobile Streaming Media CDN enabled by Dynamic SMIL. In *International World Wide Web Conference*, May 2002.