

Handling Multimedia Objects in Peer-to-Peer Networks

Vana Kalogeraki
HP Labs
Palo Alto, CA
vana@hpl.hp.com

Alex Delis
Univ. of Athens
Athens, Greece
ad@di.uoa.gr

Dimitrios Gunopulos
Univ. of California
Riverside, CA
dg@cs.ucr.edu

Video-on-demand systems and services [4, 2] are predominantly offered over dedicated private networks with the help of large servers [3, 13]. Such systems are restricted by the number of concurrent accesses allowed as well as load balancing issues that ensue when the demand for video objects is skewed [1, 13]. The widespread usage of broadband networks in connection with the low-cost commodity hardware may offer alternative avenues for the delivery of video streams and other multimedia objects. This goal can be achieved with the help of multiple, independent, and inexpensive computing nodes that function using a peer-to-peer P2P protocol [9, 8, 7]. P2P services have so far concentrated in the exchange/sharing of “small” objects including mp3, images, and audio files as well as CPU cycles and memory space [5, 11]. Discovery of objects solely based on the existing message-based schemes may prove rather ineffective [10]. Centralized indexing methods used in early P2P systems are very restrictive [12]. As groups of sites tend to form sizable node clusters [10], the large total number of dispatched messages makes inefficient use of computing resources. It is our position that furnishing video services on a P2P network not only can offer viable alternatives to the mostly proprietary architectures used today for the delivery of video services, but also it can be done in a reliable and scalable manner.

We build upon the approach of ad-hoc P2P networks of resources and propose a new architecture that can support storage and retrieval of movies and/or video-clips. Our proposed configuration exploits the availability of high-performance links to networks, the usage of exclusive and partial indexing in peers, making nodes “aware” of the content in their own vicinity, replication of objects and caching of popular items, as well as full connectivity among servers whenever feasible. The architecture uses efficient indexing mechanisms for the retrieval of the multimedia objects, guarantees continuous operation in light of server failures and allows for the transparent population of new servers as well as the evolution

of the furnished services and/or network resources with minimal disruption to the users. One key provision to realize such a P2P infrastructure is that the core peers (or servers) are expected to be linked via a low-latency and high-bandwidth network capable of effective handling and delivery of voluminous multimedia data. It is also anticipated that end-users should have sufficient connections to object servers. For acceptable quality MPEG-compressed video streams, one would approximately need 30 frames/sec and around 1.5Mbit/sec network connection to the source. This is well within reach as more individuals are connected through specialized modems (cable, DSL, etc.) or T1-level network connections [14, 6].

The system consists of a number of servers that constitute the basis for managing the storage and retrieval of video objects. Each peer functions as an indexing site as it features *local* access structures that help identify and retrieve objects of interest. Servers also maintain *partial* indexes for the video and/or clips held by sibling nodes. Fig. 1 depicts this architecture term Multiple Independent Indexing Servers (MIIS). Peers have a maximum capacity for video objects and each stored object maintains frequency accesses and timestamps for the last time it was accessed.

End-users initiate multimedia object requests by using the payload of *query* messages. When a server receives a request, it searches through its local repository. If the object is found locally, it is presented to the requesting site (through a *query_reply* message). Otherwise, the server uses its partial index to check if the objects are stored in a sibling (peer). There are two options: 1) *First Cache-Then Deliver*: The item is first cached to the server that the requesting site is attached to and then it is distributed. The condition that enables such a copy is that an object has become popular (object has received $k\%$ of the most recent Δ requests). User sites ultimately receive objects from their own respected serving peers. 2) *Forward Object*: The data server managing the item streams the object via the network directly to

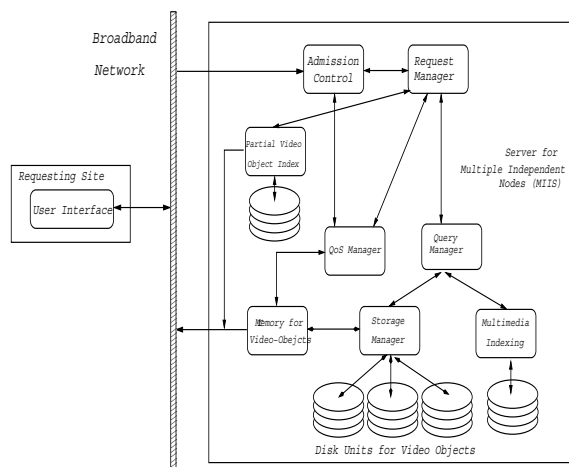


Figure 1. The MIIS Configuration

the requesting site. In this case, users obtain multimedia objects from peers that “anchor” the objects in question.

Whenever multimedia object caching takes place, the peer refreshes its local index and notifies its siblings so that they change the contents of their own *local* indexes. Clearly, not all objects are available in all sites but should the cache first-then deliver option be used, it is guaranteed that at least the most popular video/clips will be present in multiple locations. This offers reduced object access latency and increased reliability. If the server is unable to immediately locate the peer which has the multimedia object, it broadcasts a query message (with a constrained TTL) to all its peers for further processing. The latter will ultimately satisfy the request as it triggers a search of all the possibly related servers in the P2P network.

Newly-arrived objects are published by client sites to corresponding peers by issuing upload messages. Peers are then responsible for not only updating their own multimedia indexing structures but also alerting their network counterparts as soon as possible. This notification can be carried out with a low-entropy protocol. Arrival of nodes in the P2P network can be facilitated in a rather straightforward manner. As soon as a new (server) peer joins in, it gets populated either by caching objects from other servers or by processing client requests for uploads.

To ensure reliable operation of the network in light of node failure and/or time outs, we propose a simple yet effective replication policy that calls for the mirroring of an object to at least another node. The selection of the new node can be done randomly or based on load-balancing heuristics. In conjunction with the possible caching and subsequently floating copies (due to First Cache-Then Deliver operation above), the P2P system will be able to recover from more than two site failure. As peers reach their storage capacity, objects can

be eliminated in a least recently requested object fashion provided that at least two copies exist in the network.

The emerging peer-to-peer (P2P) distributed computing protocols in conjunction with ever increasing network capabilities offer new and exciting opportunities for a wide range of applications including tele-medicine, news delivery, digital libraries and distance learning. The proposed architecture termed Multiple Independent Indexed Servers (MIIS) is an option that can offer good performance behavior and effective video and multimedia object delivery to end-users. Preliminary experimental results indicate that on average, the MIIS architecture achieves a small number of message before the streaming of a server object commences. As the replication degree of popular video-objects quickly enlarges, the corresponding server “cache hit” rates for such items continuously increase. Consequently, the number of messages needed to find a movie/clip invariably decreases over time and the reliability of the overall network is enhanced.

References

- [1] B.Ozden, A. Biliris, R. Rastogi, and A. Silberschatz. A Disk-Based Storage Architecture for Movie On Demand Servers. *Information Systems*, 20(6):465–482, 1995.
- [2] S.R. Carter, J.F. Paris, S. Mohan, and D.D.E. Long. A Dynamic Heuristic Broadcasting Protocol for Video-On-Demand. In *Proceedings of the 21st IEEE International Conference on Distributed Computing Systems*, Phoenix, CA, May 2001.
- [3] L. Golubchik, R.R. Muntz, C.-F. Chou, and S. Berson. Design of Fault-Tolerant Large-Scale VOD Servers with Emphasis on High-Performance and Low-Cost. *IEEE Transactions on Parallel and Distributed Systems*, 12(4):363–386, April 2001.
- [4] J. Hsieh, M. Lin, J.C.L. Liu, D.H.-C. Du, and T. Ruwart. Performance of a Mass Storage System for Video-On-Demand. In *Proceedings of the IEEE INFOCOM Conference*, Boston, MA, 1995.
- [5] J. Kubiatowicz, D. Bindel, Y. Chen, S. Czerwinski, P. Eaton, D. Geels, R. Gummadi, S. Rhea, H. Weatherspoon, W. Weimer, C. Wells, and B. Zhao. OceanStore: An Architecture for Global-Scale Persistent Storage. In *Proceedings of ASPLOS*, Cambridge, MA, 2000.
- [6] R. Lienhart, M. Holliman, Y.-K. Chen, I. Kozintsev, and M. Yeung. Improving Media Services on P2P Networks. *IEEE Internet Computing*, 6(1):73–77, January/February 2002.
- [7] Sun Microsystems. Jxta. <http://www.jxta.org>.
- [8] Gnutella Home Page. Gnutella.Com. <http://www.gnutella.com>.
- [9] Napster Home Page. Napster.Com. <http://www.napster.com>.
- [10] M. Ripeanu, A. Iamnitchi, and I. Foster. Mapping the Gnutella Network. *IEEE Internet Computing*, 6(1):50–57, January/February 2002.
- [11] A. Rowstron and P. Druschel. Storage Management and Caching in PAST, a Large-scale Persistent Peer-To-Peer Storage Utility. In *Proceedings of the 18th SOSP*, Toronto, Canada, 2001.
- [12] I. Stoica, R. Morris, D. Karger, M.F. Kaashoek, and H. Balakrishnan. Chord: A Scalable Peer-to-peer Lookup Service for Internet Applications. In *Proceedings of ACM SIGCOMM Conference*, San Diego, CA, August 2001.
- [13] M. Vernick, C. Venkatramani, and T. Chiueh. Adventures in Building the Stony Brook Video Server. In *Proceedings of the Forth ACM International Conference on Multimedia*, Boston, MA, November 1996.
- [14] D. Wu, Y.T. Hou, W. Zhu, Y.-Q. Zhang, and J.M. Peha. Streaming Video over the Internet: Approaches and Directions. *IEEE Transactions on Circuits and Systems for Video Technology*, 11(1):1–20, February 2001.