

White Paper



Distributed Video-on-Demand – A grid based VoD solution

Grid Computing Focus Group,
Software Engineering and Technology Labs

Multimedia communication has been in a continuous state of evolution over the past few years which have been clearly driven by an exponential rise in consumer demands. Today's multimedia industry caters to a multitude of consumers with highly disparate taste. Thus broadcast services which do not let the user to actively participate and control his subscription at his will, pave the way for its on demand counterparts.

Video-on-Demand - Background

Pay-per-view (PPV) services could be considered a primitive form of distributing media on demand. This requires the subscriber to sign-up for an account and thus enabling him access the service. The subscriber is being charged for installation and a periodic rental. This scheme is different from pure broadcast in the sense that it provides the subscriber the control to receive according to his subscription.

Quasi Video-on-Demand (Q-VoD) services, takes selective subscription a little more ahead by multicasting media content amongst a group of users who share a common set of interests. To access media content that is not available in a particular group a subscriber belongs to, he can switch between groups. Near video-on-demand (N-VoD) services simulate media access control functions like forward and reverse in discrete time intervals. This capability is usually facilitated by providing multiple channels with the same media content, skewed in time. All these concepts collate to the introduction of a True Video on Demand system.

To provide control to the subscriber, a True Video-on-demand system requires a feedback mechanism installed at the subscriber device that aids the Video-on Demand service engine control the rate of data transfer over the network. Thus depending on the network bandwidth the service provider, signals the underlying encoding engine to manipulate the media encoding bit rate so that media can be delivered to the subscriber trading off between the quality or the request-response latency.

This process of converting and transmitting a stored media file into a stream of bytes across a computer network; which when received at the client, could be played impromptu, is called streaming. This approach thrusts ahead of the conventional download-and-play approach because with this, subscribers need not wait to view the file until it is completely downloaded from the broadcast site, instead they can start viewing the media as soon as a minimal playable chunk is received by the client. Thus the subscriber feels as if the media file is being played at his own local site. Complimentary services like rewind/fast-forward/random seeking makes the service more subscriber controlled.

The Challenge

A video-on-demand system typically is characterized by the flexibility it offers to its subscribers. A true video-on-demand service should provide its remote subscribers total control over the presentation session. It should incorporate all the functionalities of a local media play such as fast-forwarding, rewinding and randomized frame access with very little response time and nearbroadcast quality.

Some common problems/issues a Video-on-Demand needs to address are:

1. **Load distribution on server:** To support multiple connection requests from user, and facilitate minimum response time.
2. **Media content management:** This includes high storage space, effective content management, replication strategy etc.
3. **Adapt to dynamic network bandwidth:** As the client-server link may not always be consistent, one needs to manage the content corresponding to network change and still maintain quality of the media.
4. **Decide on Buffer/Cache:** To facilitate user with better quality and high-response time, the system may have to decide upon the buffer size and cache.
5. **Rate control:** For adapting to network, the system may need to vary the transport and encoding rates.
6. **Scalability and cost effectiveness.**
7. To provide **reliability and availability**

In addition to these parameters, such a set-up needs to be highly fault-tolerant and fairly scalable to ensure subscriber satisfaction. Several architectures are proposed in this regard which address the above mentioned issues by employing expensive hardware infrastructure.

This document broaches forward a distributed technique to take these problems under consideration to develop a solution prototype.

Motivation

A True Video-on-demand system would require enormous storage facility accoutred with high-speed access and replication facilities. A centralised storage in such a scenario has its own disadvantages and infrastructure costs. Thus, [pre-splitting](#) and [distributing](#) media content across various nodes with substantial storage capacities is a viable strategy.

Along with the storage problems, such set-ups are often hit by performance bottlenecks that arise as a centralised architecture doesn't scale enough while handling simultaneous connection-requests and maintaining sessions. A [distributed](#) architecture, on the other hand is capable of efficiently handling the same.

In addition, delivering streamed media content to various clients is a highly compute-intensive process. Streaming involves decoding huge data and re-encoding the same based on certain parameters in very less time, so distributing the computation efficiently across the various nodes capable of performing computation at average power leads to a scalable and better solution.

Therefore, there is a need for a distributed architecture for handling the high CPU and storage loads in a VoD system. Grid computing is considered to a highly scalable solution where compute intensive processes are split up into multiple low-cost systems so that the overall load is minimized and cost reduced. Grid works very efficiently in a data parallel scenario, where the data can be split across different nodes constituting the grid. The above paragraph shows that grid indeed is a good candidate to the distributed VoD (DVoD) solution. Also, grid based architecture ensures high availability and reliability through redundancy mechanisms which don't require expensive hardware infrastructure for their deployment.

Proposing DVOD

Objective

Our effort is oriented towards designing a scalable architecture based on grid, where the server itself is distributed among a set of 'non-dedicated' low-cost, off-the-shelf hosts like PCs and workstations

The Components

- [Media splitter](#): This works with the help of the content decoder to convert the media content into splits (atomic playable parts of the media) and distribute among the grid nodes. (It sends the splits to the split agents sitting on each node via TCP connection).
- [Media content decoder](#): This is part of the codec that knows how to split the media file into independent playable splits.
- [Streaming Server](#): This is a typical streaming server that can stream content to the client.
- [Split agent](#): This agent sits on every grid node, and receives the split in the initialization phase. This agent is responsible for passing the split to its replica (replication strategy), sending the resource statistics to scheduler for scheduling.
- [Scheduler](#): The scheduler keeps track of all the media content, its splits, the nodes and it s usage. The scheduler is responsible for sending the collated streamed content from individual grid nodes.
- [Streaming proxy](#): Point of contact for client. Forwards the request from client to scheduler.

Solution

The process involves the following steps/phases.

1. [The initialization/setup phase](#): In this stage the administrator of the system submits the media file(s) on to the splitter where it is split among the nodes in the grid.

Using the media content decoder, the media splitter calculates how many parts the file can be split into, and the administrator can decide upon to which nodes the file will be split into.

[Splitting policy](#): Currently it distributes the file linearly to the nodes

But the policy of splitting can be extended to use the resource statistics of the node also and hence decide what portion of file is worth splitting to the nodes.

Note: 1) The setup assumes that the grid nodes have a split agent and streaming server in each node.

2) We are also working on different splitting policies and strategy's for effective storage management as well as improved scheduling for streaming.

2. **Client request:** The client sees the streaming proxy as the point of contact. The client requests for media using the protocols: HTTP, RTP/RTCP, and RTSP.

Note: We have tested the HTTP implementation; RTP/RTSP is in process.

3. **Request scheduling:** The streaming proxy forwards the request to the scheduler who maintains the metadata about the nodes and the media splits.

Now the scheduler takes over and asks each of the nodes to send the streams of the splitted content available to them.

4. **Streaming and response:** Each of the grid nodes start streaming the content (their part of the media file) to the scheduler.

Grid scheduler collects the streams (as a part of response from grid nodes) and redirects them to the client as a continuous flow.

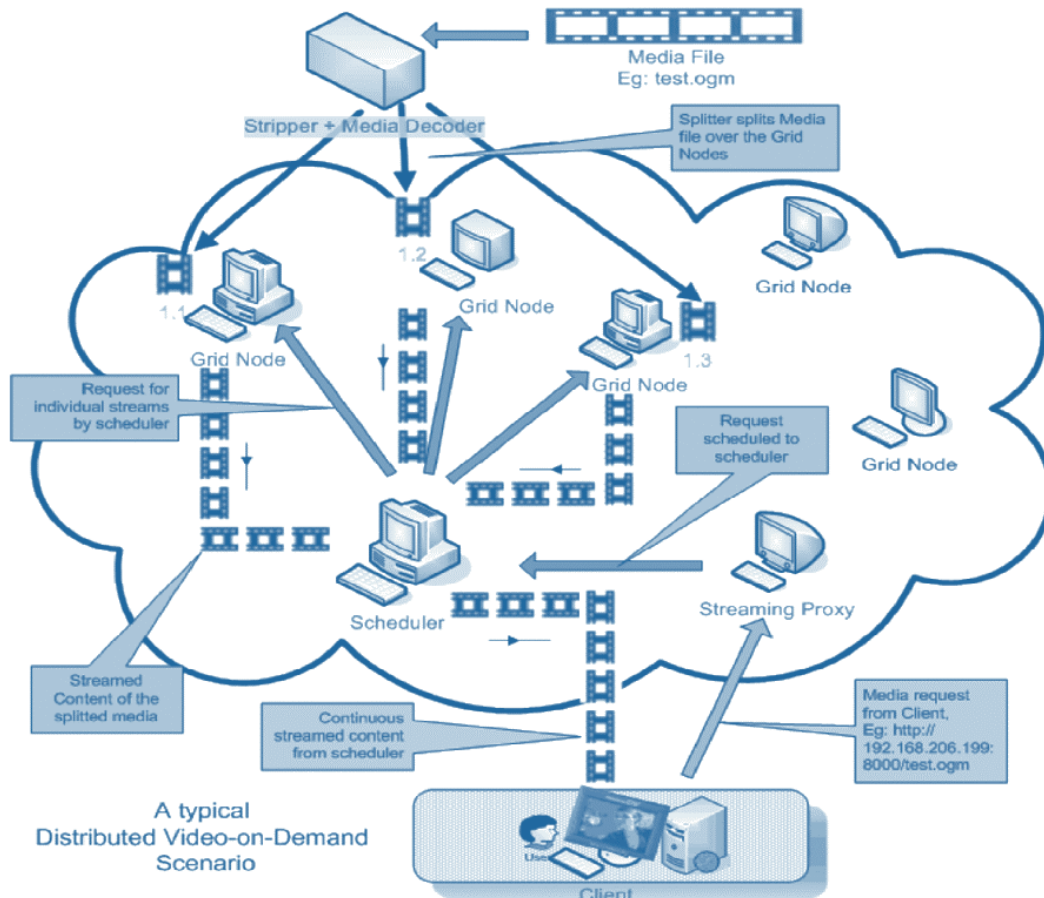
Note: 1) The process of streaming at the nodes may not be simultaneous. They can be timed to send the streams after a particular delay to avoid buffering, research is ongoing for designing this strategy.

2) The nodes requested may not always stream their content. We are working on a replication strategy to help 'busy' nodes forward the request to their replica nodes to do the streaming.

3) We are considering an alternative approach where the proxy/scheduler can be implemented in the client.

5. **Serving On-Demand video:** We are yet to get to implement a on demand system. But splitting the media content over the nodes would facilitate the seeking of media by client. The VCR type operations like, forward, rewind would be easier as the content is distributed among nodes and seeking to a particular time can be resolved easily.

The following diagram illustrates a typical DVOD system.



The diagram illustrates a typical scenario where a distributed Video-on-Demand can be applied and the way it works,

- The media file *test.ogm* is split to three parts, 1.1, 1.2, and 1.3.
- There is request from client, an *http* request in this case, for media. This request is forwarded to scheduler. The scheduler asks for streamed content from grid nodes, which stream the individual parts to scheduler.
- The scheduler sequences them back to client, where it is viewed.

Most of the stream-able file formats available are proprietary and use of these incur patent intensive disputes. Hence a completely open, patent-free, professional audio/video encoding and streaming technology is natural research alternative. OGG is one such implementation from Xiph.org which is the encapsulation format for Xiph's open-source audio encoding library (Vorbis) and video encoding library (theora).

Both vorbis and theora achieve significant compression without much loss to quality. These formats also employ variable-bit-rate encoding too.

The above stream-able formats are supported by Icecast which is an open source streaming server. Together with a source streamer (ices or ezstream), Icecast broadcasts streamed media content to the requesting clients. Streaming internally is done by modified implementation of libshout API.

Related Work

Following are an enumeration of some of the works done towards designing a scalable architecture, reducing server load/storage and network adaptation.

There has been other promising works in designing effective codecs, their encoding and transmissions rates, and design optimal buffer/cache and utilizing the network protocols like ATM and frame relay.

1. Server-less VoD architecture: Using a set of set-top-boxes for media storage and delivery.
(http://www.mcl.ie.cuhk.edu.hk/lee_leung_icme2002.pdf)
Jack Y.B Lee and Raymond W.T Leung, Department of Information Engineering, The Chinese University of Hong Kong
2. Distributing streaming media content using cooperative networking: The clients form a cooperative network to reduce the load on server. An architecture complimenting client-server architecture.
(<http://research.microsoft.com/~helenw/papers/msr-tr-2002-37.pdf>)
Venkata N. Padmanabhan, Helen J. Wang, Philip A. Chou, Microsoft Research
Kunwadee Sripanidkulchai, Carnegie Mellon University
3. P2VOD: Providing fault tolerant Video-on-Demand streaming in p2p environment. Using the P2P architecture to solve the problem of
 - Network bandwidth,
 - Using IP multicast,
 - High maintenance/deployment of dedicated overlay routers.(<http://www.cs.huji.ac.il/labs/danss/p2p/resources/p2vod-providing-fault-tolerant-videoon-demand-streaming-in-p2p-environment.pdf>)
Tai T. Do, Kien A. Hua, Mounir A. Tantaoui,
School of Electrical Engineering and Computer Science, University of Central Florida, Orlando
4. Design and implementation of a low-cost clustered video server using a network of personal computers
(<http://doi.ieeecomputersociety.org/10.1109/MMSE.2002.1181634>)
Apostolos Papagiannis, University of Patras Dimitrios Lioupis,
Computer Technology Institute Stylianos Egglezos, University of Patras
5. A distributed file system for Video-on-Demand system: The VODFS serves as the interface between clients and the VOD system, and streaming, replication, traffic loadbalancing and indexing functionalities are encapsulated in the file system design.

The novel aspect of VODFS is its file format level knowledge about supported video format and can deliver them in a streaming fashion transparently. The distributed file system adopts a hybrid of centralized and a peer-to-peer networking model with a central server (or a server cluster).

Based on the loads of the server and the availability of replicated copies, the server could either serve the file itself or broker a client to one or more replicated copies of the requested video available on the network. The server can configure the caching and replication throughout nodes in the network to achieve desired level of reliability and data redundancy.

(http://www-scf.usc.edu/~yuanchuc/555_VOD_paper.doc)

Yuan-Chun Chiu, Computer Science Department, University of Southern California.

6. Network-adaptive video coding and transmission: In this paper, instead of using mechanisms to ensure reliable transmission of video packets and improve error resilience, the focus is on end to end adaptation to network congestion, which can be used in conjunction with error recovery

(<http://amp.ece.cmu.edu/Publication/Tsuhanspie.pdf>)

Kay Sripanidkulchai and Tsuhan Chen. Department of Electrical and Computer Engineering, Carnegie Mellon University

7. On transmission scheduling in a server-less VoD system: Addressing the problem of transmission scheduling in a server-less VoD System.

(<http://www.mcl.ie.cuhk.edu.hk/slvods07.pdf>)

C.Y. Chan and Jack Y.B Lee

Department of Information Engineering, the Chinese University of HongKong.

References

[1] Icecast official documentation <http://www.icecast.org>

[2] Xiphophorous,OGG Project initiative <http://www.xiph.org>

[3] OGG theora specification <http://www.theora.org>

[4] OGG vorbis documentation <http://www.vorbis.org>

[5] RFC 3533 - The OGG Encapsulation Format Version 0 <http://www.faqs.org/rfcs/rfc3533.html>



For more information, contact askus@infosys.com

About Infosys

Many of the world's most successful organizations rely on Infosys to deliver measurable business value. Infosys provides business consulting, technology, engineering and outsourcing services to help clients in over 30 countries build tomorrow's enterprise.

For more information about Infosys (NASDAQ:INFY), visit www.infosys.com.