New Insights on Internet Streaming and IPTV

Zhen Xiao
Peking University
Beijing, China
xiaozhen@net.pku.edu.cn

Fan Ye
IBM T. J. Watson Research Center
Hawthorne, NY 10532, USA
fanye@us.ibm.com

ABSTRACT

The Internet witnessed its traffic evolved from text and images based traditional Web content to more multimedia rich applications in the past decade. As a result, multimedia and Internet streaming technology have become an increasingly important building block to many Internet applications, ranging from remote education, digital radio, Internet Protocol TV (IPTV), etc. This paper discusses the fundamental issues in Internet streaming delivery and associated technical challenges. It emphasizes the architecture and system differences between streaming applications and traditional Web applications. It reviews our research experience and presents lessons learned in this area as well as points out directions for future work.

Categories and Subject Descriptors

A.1 [**General Literature**]: Introductory and Survey ; H.4.3 [**Information Systems**]: Information Systems Applications—*Communications Applications*

General Terms

Measurement, Performance

Keywords

network measurements, multimedia, Internet streaming, IPTV, mobile video, peer to peer, video on demand

1. INTRODUCTION

The past decade saw the evolution of Internet traffic from mostly text and images to increasingly more multimedia objects such as audio and video. Multimedia applications such as Internet Protocol TV (IPTV) have become increasingly important. Those applications have brought a new set of technical challenges. Due to the large size of media objects, multimedia delivery typically requires significantly more bandwidth than traditional Web content and for a much longer period of time. A streaming service provider can use a Content Delivery Network (CDN) or a Media Delivery Network (MDN) to assist its distribution.

CIVR'08, July 7–9, 2008, Niagara Falls, Ontario, Canada.

Internet streaming technology also brings in more interesting applications. It can be used to transmit traditional TV content but in a much more flexible manner. Due to cost considerations, traditional TV networks typically offer channels only if there are sufficient user bases. For example, a TV network may be willing to offer Chinese programs in New York City where a large Chinese population live, but not in many other parts of the country. In contrast, Internet Protocol TV is more flexible and can be offered virtually anywhere broadband Internet connection is available. It also makes it easier for users to access unpopular video on demand content, such as a legendary movie decades ago which is no longer available in any major TV network. YouTube represents another type of applications for Internet streaming where the users are content providers as well as content consumers [5]. Just like the Web transformed everyone into a potential information publisher on the Internet, IPTV technology like YouTube transformed everyone into a potential media publisher – in the sense you can publish your own TV channel on the Internet.

Internet streaming media delivery can require a significant amount of resources. In fact, many streaming services today offer only a small screen size and relatively low resolution in order to save bandwidth. The quality of those streaming services is typically not comparable to that in traditional TV networks. Resource management is thus a key issue in Internet streaming deployment. To relieve the load on streaming servers and the delivery networks, peer to peer technology has been widely used where a user watching a streaming media can contribute her own resources to the delivery. The interaction between the peer to peer overlay network and the underlying physical network can be quite complex, however. In practice, not all P2P networks are efficient from a resource utilization perspective.

With the deployment of 3G networks, mobile video has become a reality in the past several years. A number of wireless service providers have started video service, ranging from live TV programs, to video on demand or preprogrammed video, for mobile devices such as cell phones. Verizon provides the V Cast video service. With $15 per month the subscriber can view TV programs on the cellphone. There are eight popular channels to choose from, including Fox Mobile, NBC Mobile, ESPN, etc. At $15-25 a month, Sprint has offered more than 50 channels of live TV programs, full-length movies and live concerts in its Power Vision Networks. Its coverage extends to more than 8700 cities and communities nationwide, reaching one million customers in early 2007 [3]. Other wireless service providers such as AT&T, British Telecom have all rolled out similar plans for mobile video. Given the penetration and popularity of cell phones, mobile video is expected to experience tremendous growth in the near future.

The rest of the paper is organized as follows. Section 2 discusses

architecture considerations and resource provisioning strategies for Internet streaming applications. It highlights their differences to traditional Web applications. Section 3 examines the challenges for peer to peer streaming in detail. Mobile video delivery is described in Section 4. Section 5 gives a visionary view on future IPTV environment unified with other communication services such as phones, instant messaging, etc. Related work is discussed in Section 6. Section 7 concludes the paper.

2. ARCHITECTURE AND RESOURCE CONSIDERATIONS FOR INTERNET STREAMING

In this section, we examine the architecture and resource considerations for Internet streaming. We emphasize how those considerations differ from that for traditional Web applications.

2.1 A Tiered Architecture of Internet Services

Internet services are typically organized in a tiered fashion. Figure 1(a) illustrates the commonly used three-tier architecture [8]. The first tier consists of Web gateways which are servers that interface directly with the end users. They receive requests from the users and pass them to the application servers in the second tier for additional processing. Those application servers implement core business logic specific to a particular application (e.g., financial transactions). They often need to query the database servers in the third tier. The database servers themselves can be distributed geographically and extensive work exists in database literature on distributed database processing.

In practice, however, the division of the three tiers is not strict. The processing of user requests and application logic is usually combined into a common "Front End", with the rest of the architecture referred to as the "Back End" database as shown in Figure 1(b). Although not shown in the figure, Web and application servers in the front end do not have to be separate entities – the same server can both communicate to the users and conduct application processing. Additional discussion on the tiered architecture of Internet services can be found in [8].

Internet media delivery, however, brings additional challenges. As discussed earlier, streaming applications have much higher resource requirements than traditional Web applications. Streaming sessions typically last several minutes or even hours and require a high bandwidth pipe between the server and the client. Similar to Web objects, the popularity of streaming objects can be difficult to predict: a suddenly popular streaming object can draw a huge audience from all over the world. Since streaming servers need to keep their connections for a long period of time, the surge in demand or so called "flash crowd" event can make resource provisioning challenging.

Large streaming sites like YouTube typically host a huge selection of streaming objects whose popularity may change from time to time. Hence, static allocation of servers to streaming applications can be inefficient due to the difficulty to estimate their time-varying popularity. Moreover, user demands tend to have a geographical preference due to time zone and other reasons. For example, a streaming application that provides employee training for a major corporation in New York is likely to experience its peak load during business hours in the east coast, while a similar application for a corporation in Europe will have a different peak time. A streaming application that provides mostly entertainment videos, on the other hand, is likely to experience the highest load at evenings and weekends. [1] Previously, resource provisioning has been done manually by system administrators. By monitoring user demand and server load, they can replicate popular applications to the "hot" spots in the network. For example, if a streaming application experiences a surge in user demand near the east coast, an administrator can instruct the system to deploy more copies of the application in a nearby server farm. This manual approach, however, is labor-intensive and is feasible only when the scale of the system is small. For large systems that contain thousands of servers across a wide operating range, an automated approach for application deployment is needed.

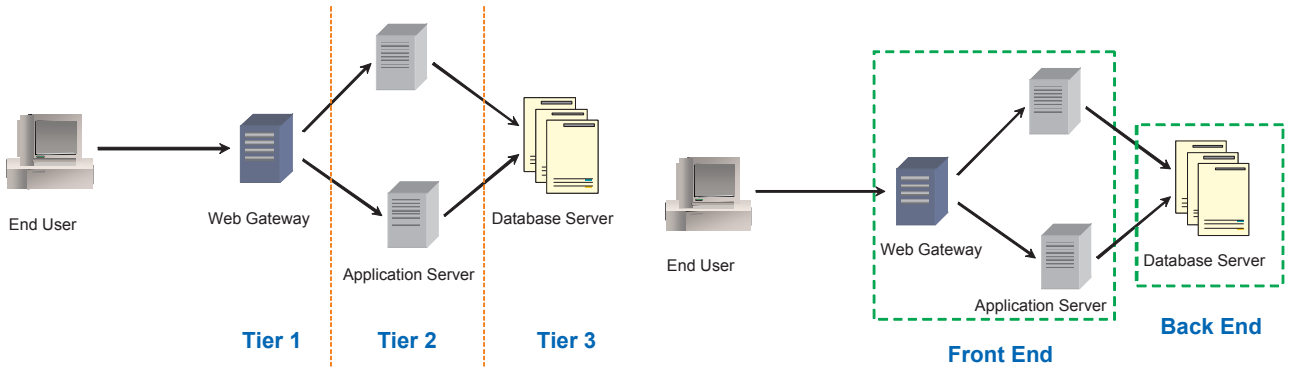
Determining the number and locations of servers for an application has been previously studied in the context of the "content placement" problem. It is a classical problem in distributed systems and has received attention from researchers in graph theory. A classical approach is to model it as the facility location problem: Let $G = (F, C)$ be a bipartite graph, where F is a set of facilities, and C is a set of cities. Building a facility at location i incurs a cost of f_i. Each city j must be connected to one facility i, which incurs a cost of $d_j \cdot c_{ij}$, where d_j denotes the demand from city j, and c_{ij} denotes the distance between i and j. The distance function c satisfy the triangle inequality (i.e. the metric version). The goal is to find the number and location of facilities that should be opened, and a function that assigns cities to open facilities in such a way that the total cost is minimized.

There are several variants of this problem, such as the capacitated version (i.e. each facility can serve no more than a certain number of cities), the fault-tolerant version (i.e. each city must connect to at least a specified number of facilities), etc. There is also a related problem called the k-median problem. It is similar to the facility location problem with two main differences: (1) There is no cost for opening a facility; (2) An upper bound, k, is specified as the maximum number of facilities that can be opened.

Both the facility location problem and the k-median problem are NP-hard. Given their obvious implications on many practical situations (such as the optimal placement of edge servers in a content delivery network), various approximation algorithms have been proposed, such as recent work of [10, 16, 28] based on the use of primal-dual schema and Lagrangian relaxation. Although these solutions are theoretically interesting, they have limited applicability in practice due to several reasons. The first reason is that they assume global knowledge of the demand from each node and the network topology. In practice, such information is difficult to gather in a large system. In addition, they assume the demand and cost functions are fixed, while in a real network the demand for resources as well as the cost for serving user requests can change dynamically. More importantly, these theoretical models abstract away many system details that need to be considered in practice. For example, the content placement decision may need to consider the mutual trust relationship between different servers or the business agreement among peering ISPs, factors that are not considered in those theoretical models.

Another related question is: What does proximity mean? In a multi-tier architecture, servers at different tiers may well be deployed at different locations. The back-end database, for example, can often be found at a remote location from the application server. Hence, proximity to one part of the system does not necessarily mean proximity to the rest of the system. A good algorithm needs to study the traffic characteristics of a given application and optimizes the client-server proximity accordingly. Consider figure 2 as an example. Two Internet Data Centers (IDC) share back-end resources. The application in the first data center is front-end heavy: the traffic between the client and the application server (indicated

[1] This is not strictly true, though. Rumor has it that some video news site receives its peak demand around lunch time.



(a) Standard three-tier architecture

(b) Merged into two-tiers in practice

Figure 1: Three-tier Architecture of Internet Services

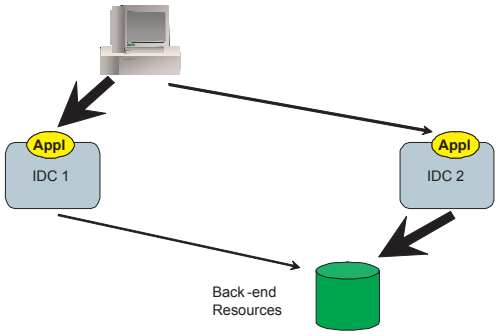


Figure 2: Client proximity

by the wide, heavy arrow) outweighs the traffic between the application server and the back-end resources (indicated by the thin, narrow arrow). Hence, clients accessing this application should be directed to a data center close to the client as shown in the figure. In contrast, the application in the second data center is back-end heavy: the traffic between the application server and the back-end resources outweighs the traffic between the client and the application server. As a result, clients accessing this application should be directed to a data center close to the back-end resources as shown in the figure. In other words, proximity can be application specific and measurement is often needed to understand the characteristic of an application before an appropriate algorithm can be selected [8].

Streaming applications are often mixed with other applications in Web site design. For example, many streaming applications derive their revenue from online advertisements, which are typically hosted in separate servers and have very different access patterns. The resource decision for the *extraneous* traffic should be made separately from that for the true streaming traffic due to their dramatically different characteristics.

2.2 Request Dispatching

After an appropriate application server is selected based on proximity and other factors, a mechanism is needed to route client requests to that server. One way to accomplish the task is to use the service from the Domain Name System (DNS). DNS provides a mapping from a human readable host name (e.g., www.buystock.com) to a machine readable IP address. It can be configured to return different IP addresses depending on the location of the user who issues the DNS query. DNS-based request dispatching is illustrated in Figure 3.

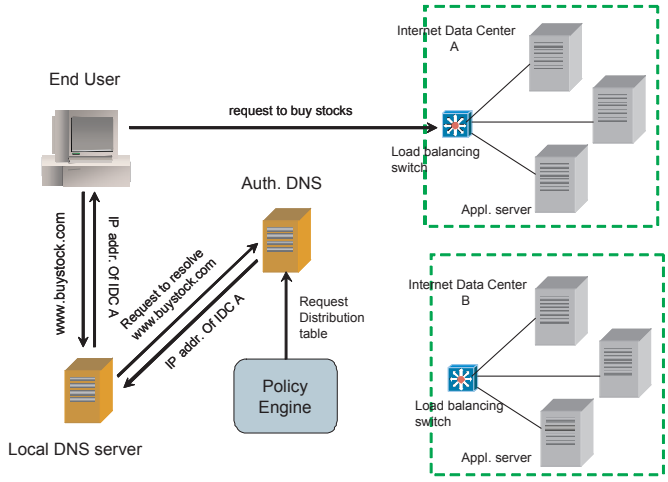


Figure 3: An example of an Internet user buying stocks from an online brokerage site.

Here we use the example of an end user who wants to buy stocks from an online brokerage site www.buystock.com. Each data center consists of a load balancing switch connected to a group of application servers. Only the load balancing switch has an externally visible IP address. The server selection in this case is made in two steps: first at the granularity of data centers, and then at an application server within that data center. For simplicity, we do not show the back-end of the Internet services in the figure. The transaction involves the following steps:

- The end user initiates a request to buy stocks. The computer first sends a DNS query to the local DNS server to resolve host name www.buystock.com.
- The local DNS server sends a request to the authoritative DNS server to resolve www.buystock.com.
- As shown in the figure, the Web site is replicated across multiple data centers. The authoritative DNS server consults a Request Distribution table and selects a data center based on the input of a Policy Engine. Policy considerations include the proximity, load, and other factors for each data center.

- The authoritative DNS server sends a response to the local DNS server with a record containing the IP address of the load balancing switch of the selected data center.
- The local DNS server forwards the response to the end user. It also caches the response for a certain period of time so that future requests can be resolved without contacting the authoritative DNS server again.
- The end user sends the request to the server with the IP address she received. The request arrives at the load balancing switch of the data center.
- The load balancing switch forwards the request to an appropriate application server based on load and other factors. That application server will complete the transaction with the user from now on (i.e., session stickiness).

Despite some pitfalls with DNS based request dispatching, this architecture is widely used for Web services. It works because the typical processing time for a Web transaction is short. Hence, resource allocation at request boundaries is sufficient.

One important issue is application migration or replication. Moving a live application from one server to another can involve a significant amount of overhead, especially if the two servers are located at different sites. If we allow the server migration to happen at arbitrary time, then the amount of state that needs to be transferred can include (among others) the entire memory footprint of the running application. For Web services, we and others proposed a solution which restricts migration to happen only during request boundaries when the amount of state that needs to be transferred is small [33]. This is again based on the observation that Web transactions are typically short. Hence, instead of migrating a live application, we can simply deploy a new instance of the application at the destination server and start redirecting new requests there. After the effect of DNS caching wanes off, we can optionally decommission the old application instance [7, 34].

However, this does not work well with streaming applications. Streaming connections can last for a long time during which resource conditions at application servers can change substantially. A server which is lightly loaded at the beginning of a two-hour movie may become quite heavily loaded in the middle of the session due to other processing. Moreover, due to their real time delivery requirement (a late frame is as bad as a lost frame), migrating applications without causing noticeable disruption to user viewing experience is challenging.

One solution for this problem is to utilize multiple servers concurrently. Both system and encoding support exists for a user to receive data from multiple sources simultaneously and then merge the data into a single media stream before displaying it to the user. In this case, we can deploy the streaming application in the new server and have both of them feeding the data to the user initially. Gradually, we shift the load from the old server to the new one and eventually decommission the old server. It has been suggested that having multiple servers in different geographical locations is an advantage due to the path diversity it creates: if one part of the network is congested for some reason, packets can still go through via alternative paths.

3. PEER TO PEER STREAMING

Peer to peer technology is one way to address the resource problem with Internet streaming. Due to the large sizes of many multimedia objects, it can be highly expensive to store them in a centralized data center. Even though the cost of storage per gigabyte is getting cheaper with modern technology, the size of media objects is getting larger as high definition video content becomes more common. In addition, the bandwidth requirement to stream video content from centralized location to end users can be overwhelming. The real time delivery of a media stream with the quality of a standard definition TV, for example, can require up to 2Mbps bandwidth, while that for a high definition TV can be as high as 6Mbps [27] (The required delivery rate of DVDs or other stored media is lower because it is compressed offline.) The demands on the amount of processing power at streaming servers can also be prohibitive.

The benefit of P2P is to use the otherwise idle resources in end users' computers (CPUs, bandwidths, disks, etc.) to offload the central servers. It also has the potential of reducing the user perceived latency by pushing the data and computation to a place closer to the users. We use the P2P networks in the popular BitTorrent system as an example. Peers organize into an overlay network. Each peer holds a portion of the file and exchanges file chunks with other peers. Many studies have been done on P2P streaming. A nice feature is the self-scaling property: peers join the network contributing additional resources. Several popular P2P streaming systems report better viewing experience when there are more users in the systems. The focus of this section is not to re-exam the wide variety of P2P algorithms and systems, but to discuss several commonly overlooked issues when going from a centralized system into a P2P architecture. Sometimes the implications are unexpected.

3.1 Encoding for heterogeneous receivers

The connection speed of Internet users ranges widely from slow dial-up connection to broadband DSL and cable modem or even high speed Ethernet (e.g., university users) or optical fiber. This trend is expected to continue with the wide deployment of broadband Internet access (e.g., Verizon offers several different speeds in its DSL plan with different pricing). It is undesirable to use a fixed encoding rate to accommodate a heterogeneous set of streaming clients. If a client does not have enough bandwidth, she may experience frequent delays during the media playback as the media player has to stop to buffer for the new data.

Various techniques have been proposed to address this issue. Layered encoding is a technique that divides a media stream into a base layer and an ordered set of enhancement layers. A client receiving the base layer can decode the media stream with basic quality. The more enhancement layers she receives, the better the quality. There is a partial dependency among those layers in that lower layers are needed for higher layers to be decoded. In contrast, Multiple Description Coding (MDC) divides the media into a set of *descriptions*, each of which is individually decodable. The more descriptions a receiver gets, the better the quality. Compared to layered encoding, MDC is more flexible since there is no dependence between descriptions. On the downside, it has a higher overhead than layered encoding. In practice, however, neither of them is used in any major Internet streaming system due to the encoding overhead and complexity.

Multiple-Bit-Rate (MBR) encoding is a technique that is actually used in practice. It encodes the same media content multiple times with different bit rates to accommodate the heterogeneous set of receivers. It also supports stream switch: when a network congestion causes the available bandwidth between the client and the server to drop below a certain threshold, the client and the server can coordinate to switch to a low quality stream. MBR is supported by both Windows Media [4] and Real Media [2]. It is called Intelligent Streaming in Windows Media and SureStream in Real Media.

As we can see, MBR is similar to MDC in that each bit rate stream in MBR is individually decodable. MBR is also simpler than MDC because MBR encoding merely bundles several bit rate streams together while its decoding just extracts one of them. The price paid is the extra space needed to store all the streams: unlike layered encoding or MDC where the space overhead is a fraction of the original (full quality) media size, the size of the MBR is the sum of the sizes of individual streams. In a client server environment, as long as the server has enough storage to accommodate the extra space required, MBR can work well. When P2P is used, the situation is very different.

In P2P streaming, each peer possesses certain file chunks and swaps them with other peers. The exchange in almost all existing P2P systems works in the time domain. When MDC is used, the exchange can happen in the quality domain as well. Hence, MDC extends naturally to a P2P environment. The situation with layered encoding is similar except that exchange in the quality domain needs to observe the partial order in encoding (e.g., getting an enhancement layer without the base layer is not useful). When MBR is used, however, peers receiving different bit rate streams cannot exchange file chunks directly. Rather, they are partitioned into subgroups based on their bit rates. Unlike MDC or layered encoding, there is no sharing across different groups. While seemingly obvious, we believe we are the first to point out this phenomenon explicitly. The impact of this to P2P performance is debatable. Some people argue that with "tit-for-tat" incentive as in BitTorrent-like systems peers will optimize their uploading destinations to peers who can reciprocally upload the most to themselves. Hence, peers will naturally form several subgroups based on their network bandwidth even if they all receive the same bit rate stream.

3.2 Set top box vs. computers

We can divide IPTV services into two categories based on how a user watches it: a user can watch it using a computer or using a TV through a set top box. The content in the first category can be either user generated (which is the case in YouTube [5]) or provided by some content providers (such as the official Chinese TV channels in PPlive [1]). Many early IPTV services fall into this category. The content in the second category is almost always provided by content providers. The IPTV services envisioned by telecommunication companies like AT&T and Verizon fall into this category.

At a first glance, it should not matter from a system design point of view whether a user watches the program using a computer or a TV. In practice, however, we need to consider the difference in computing resource between the TV and the computer system. When IPTV delivery is made via set top boxes to the TVs, the algorithm and implementation is limited by the available resource in each set top box, which is typically much smaller than that in a computer. For example, fast streaming is a technique from Windows Media designed to smooth the jitter in streaming delivery due to network bandwidth fluctuation in a stream session. It includes several techniques: fast start, fast cache, fast recovery, and fast reconnect. With fast start, the server will deliver the media object to the client as fast as the underlying TCP allows until the play out buffer in the media player is filled. After that, it enters the fast cache phase where the server can deliver a media object at a rate up to five times its encoding rate. The client maintains a buffer at its side for the early arrived data. This is much more aggressive than the traditional streaming method where the media object is streamed at its encoding rate. In fact, it is possible for the entire media object to be delivered to the client side in the middle of a playback session.

Hence, a computer can cache the full movie in its local disk during the middle of a streaming session and hence protect against the possible bandwidth fluctuation during media playback. When the same movie is re-played, the computer does not need to fetch the data from the streaming server again. In contrast, the set top boxes deployed by most existing companies are designed for prefix caching where only the initial portion of the media is cached to reduce start up latency, thus requiring a higher predictability on the amount of streaming bandwidth available.

The limited resources in set top boxes has other important implications for P2P algorithm design. For computers, a peer that has the full media object stored on its disk can share all chunks of the object other peers. [2] Previous studies found that some altruistic peers stay in the system long after its own downloading has completed and keep their downloaded objects for a long time [15]. Incentive mechanism has been proposed to encourage swapping file chunks even across different media objects [21]. The situation can be different when set top boxes are used. Due to the limited amount of storage space, the P2P algorithm has to be designed to facilitate sharing of the current segment of the media object, since other segments are typically not available.

On the positive side, since a single company controls all set top boxes, the deployment of software and hardware can be done homogeneously. Built-in P2P algorithms can be used to ensure all peers help each other sharing their content and bandwidth in a fair manner. (More complicated algorithms are investigated in [12].) In contrast, free riding is a major problem in most computer based P2P file sharing systems and various incentive mechanisms are proposed to address that. The tit-for-tat incentive strategy in BitTorrent works well for single torrent file, but not across multiple torrents [21].

The processing power of a modern PC is likely to supersede that of a TV set top box, thus facilitating the implementation of complicated algorithms. On the other hand, a set top box can have hardware support for special encoding or decoding algorithms. [3] What is clear is that set top boxes need to have a much friendlier user interface since all operations have to be done by remote controls – the users do not have access to full fledged keyboard and mouse input. Algorithm changes require remote upgrading of the software on the set top boxes.

3.3 Digital rights management

Digital right management is important to protect the interest of content providers. It can be accomplished by encrypting part of a media object with a licensing key which is available only to authorized users. Even when the data of a movie is fully cached in the user's disk, the content provider may require the user to reauthenticate with a licensing server before the media can be replayed.

Now the question is: how do peers exchange file chunks encrypted with different license keys? If they cannot decrypt the data received from other peers, then the whole P2P scheme will break. One possibility is to encrypt a small fraction of each media segment with a key while leaving most data unencrypted. Hence, it can still reap the benefit of P2P for delivering most of the data, but requires the user to obtain the license key before decrypting any segment.

[2]Such aggressive caching is not without its own problems. Previous studies have found that in most media sessions the users only watch the beginning portion of the media objects. If the content is not interesting, a user will stop the playback. As a result, the early arrived data for the rest of the media session is simply wasted. Our early work has found that this can contribute to significant amount of over-supplied data [24].

[3]The price for that is flexibility: updating the algorithm might require upgrading the hardware.

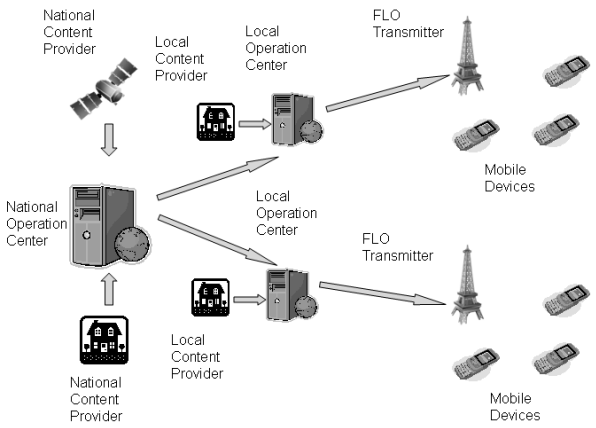


Figure 4: A FLO system consists of national and local Network Operation Centers that receive content from providers via the Internet or satellite. The FLO transmitter broadcasts video content in FLO waveforms to FLO-enabled mobile devices. The 3G network provides the network infrastructure and reverse link for user feedback and control.

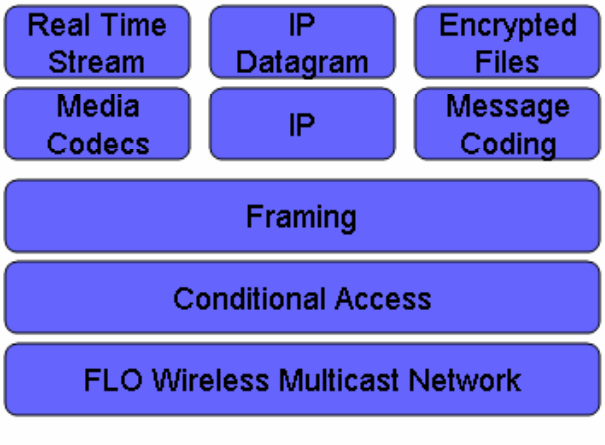


Figure 5: FLO supports three types of transport mechanisms for encrypted data, video stream and IP datagram. Each mechanism is suitable for one type of content.

4. MOBILE VIDEO

Mobile video has become increasingly popular in the past years. There exist a number of mobile video technologies and they are mostly variants or derivatives of existing digital television broadcast formats. We will explain in detail one of them, MediaFLO [32]. MediaFLO is a technology developed by Qualcomm and used in Verizon's V Cast. We will then briefly describe another technology, DVB-H [35], supported by an alliance, including handset makers such as Motorola, Nokia and device manufacturers such as Intel.

FLO and DVB-H share many similarities. They both use Orthogonal Frequency Division Multiplexing (OFDM) and transmit video streams of up to 15 frames per second on a QVGA screen. They both adopt Time Division Multiplexing techniques to conserve power at mobile devices. Technical wise, FLO has certain advantages over DVB-H, such as more efficient use of the spectrum, thus fitting more streams into the same channel (e.g., 20 vs. 6), faster switching time when choosing different video streams [18]. On the other hand, they take quite different market approaches. FLO was designed from scratch, with the intention to be spun off as a standalone technology and licensed to vendors. DVB-H was supported by traditional GSM vendors from the start, aiming at creating a technology that may not be most technically advanced, but can be supported by a wide base of carriers and device manufacturers.

MediaFLO is capable of delivering a video size of 240×320 at 30 frames per second, together with stereo audio. It can provide up to 19 live streaming video channels, including 14 national and 5 local ones; 50 nationwide pre-programmed and 15 local channels, each of which having 20 minutes of content per day. Non real time content can be delivered concurrently, allowing access to music, news or weather broadcast. FLO service provides viewing experience similar to that of TV. Users must have a V Cast mobile TV handset that has a dedicated key to receive and view the content. Users can quickly change between channels through a program guide user interface. They can also select a package that includes a group of programs and view the content at any time.

There are four major components in the system architecture of FLO (see Figure 4): the Network Operation Center, FLO Transmitters, 3G Networks and FLO-enabled mobile devices. The Network Operation Center is responsible for the content distribution and management of the network. It receives program from content providers through the Internet or satellites, and delivers the content throughout the network. Usually there is a Nation NOC that receives program from national content providers and sends the content to many Local NOCs. Local NOCs are responsible for further distributing the content through FLO Transmitters, thus end users can receive the content on their mobile devices. In addition, local content providers such as local TV stations can also deliver region-targeted programs to local NOCs. The reverse link in 3G networks allows users to communicate with the National NOC for operations such as subscriptions, access key distribution.

In FLO networks, a real time channel is usually streamed directly from content providers to Network Operation Centers. This can be done through C-band satellite in MPEG2 format, which is the most common format for most content providers. Due to the higher resolution (e.g., 720×480 pixels), the content will go through some transcoding to H.264 QVGA resolution to suit the screen size of mobile devices. For pre-recorded programs, the NOC usually receives them from the Internet. The program will be reformatted into FLO packet streams and redistributed by the FLO Media Distribution System throughout the network, based on pre-arranged schedules. The FLO transmitters will broadcast corresponding FLO waveforms. The content from local providers will be combined with nationwide content and broadcast simultaneously to end users.

An FLO network can deliver multiple types of content, including real time video, IP datagrams and encrypted files. Each type utilizes a different transport mechanism that is most suitable for its requirements (see Figure 5). For example, real time content is fed to a media codec that minimizes the effects of lost packets in streaming. Thus the occasional loss of data does not affect the viewing experience too much. In contrast, IP datagrams are usually for best effort data that can tolerate losses naturally.

After the data is properly coded or packaged, it is fed to the frame layer. The physical signal of FLO is organized into the so called "super frames". Each frame consists of 4 frames of data, with TDM pilot frame, Overhead Information Symbols (OIS) frame. For every

MHz of allocated bandwidth, a superframe has 200 OFDM symbols. At 6 MHz, the total length of a superframe is 1200 OFDM symbols and takes 1 second's time for transmitting. The purpose of TDM pilots is to allow receivers to quickly acquire the OIS frame, which contains information about the location of the data in each of the data frames. A data frame consists of both wide-area and local-area data. Inside the data frame, every OFDM symbol consists of 7 interlaces of active subcarriers. These interlaces are assigned to logical channels. Depending on the data rate of the channel, it may receive more or less interlaces. A high data rate channel can utilize more interlaces so as to reduce the power consumption at the mobile device, while a low data rate channel will receive less interlaces so as to improve the time diversity.

The logical channel carries real time stream data at variable rates. One channel can have multiple codecs that produces data at variable rates, reliability and quality of service. Depending on the application or users' requirement, different codecs are chosen. Another reason to utilize multiple codecs is to support statistical multiplexing gain. A codec of variable rates produces a data stream that changes its rate temporally. When multiple such codecs are used together, their data streams can use the unoccupied capacity from each other, within the same channel. The mobile device can decode a single logical channel it is interested in, it can also decode multiple channels such that video and audio can be carried in different channels.

The modulation of streams takes a layered approach. A base layer that can be decoded by all users provides 15 frames per second video quality. It has wider coverage and can be received under most Signal to Noise Ratios (SNR). An enhancement layer can be received only under higher SNR levels. It provides additional information to the base layer to provide 30 frames per second video quality. Such a layered approach allows the graceful degradation of quality of service under varying reception conditions, when the SNR can change due to the signal coverage or user mobility.

A big challenge for mobile video is the power management of mobile device. Streaming video consumes one magnitude higher bandwidth than a voice call, consequently the radio power on time at the mobile device is much higher. One technique FLO employs to reduce the power consumption is through Time Division Multiplexing (TDM). Each content stream uses a distinct time slot and is transmitted at a specific interval within the time frame. A mobile device can power up its radio to receive the stream it is interested in at those specific time intervals, and power down the radio at other times.

The DVB-H standard [31] is an extension of DVB-T specification for terrestrial digital television broadcast. Its system architecture is similar to that of the FLO. It also consists of a program production center and a distribution network. At the TV production center, multiple program mixes are generated from national and local channels. Each mix is intended for a specific region and contains a few common national channels and several local channels for that region. The program mixes can be distributed to many local towers in two ways. One is through a satellite. The mixes will be transmitted to a satellite uplink earth station over a terrestrial network. The earth station will transmit the program mixes to a satellite, which in turn broadcasts the program to many local towers. Another way is using a content distribution network over the Internet. In either case, the local towers can receive the program mixes. A tower will then demodulate the program mix intended for its local region. End users in that region can receive and select among the channels within that program mix.

DVB-H also uses Time-Division Multiplexing to reduce the power consumption for mobile devices. The channel is divided into multiple time slots and a data stream uses one specific slot. During each time slot, up to two megabits of data can be transmitted. The mobile device needs to power up its radio during only the specific slots. Depending on how many streams are multiplexed over the same channel, it can achieve power saving of up to 90%.

5. A UNIFIED IPTV SYSTEM

We envision that future IPTV will be offered not as a standalone application, but rather as a unified service bundled with other applications such as Voice over IP (VoIP) and instant messaging (IM). Remember the movie "Click"? While a scientific fiction, it does illustrate the convenience of having a single point of control in life. The past decade saw the convergence of voice, video, and data onto a single communication network. Telecomm companies, for example, began to offer high speed Internet access and IPTV services, while cable companies fight back by offering VoIP and cable modem access. Unfortunately, different communication media still rely on different sets of devices for control and we are far from giving the users a unified control over their total life experiences.

As a concrete example, suppose the phone rings when a user is in the middle of watching a TV show. Currently, the user has to pause or mute the TV, goes to her phone and checks who is calling via Caller-ID, and decides if she wants to answer. In other words, the phone and the TV are offered as two independent services, despite coming from the same cable network. In a unified IPTV and VoIP environment, when the phone rings, a dialog box will pop up on the TV screen to indicate who is calling. The user can then decide if she wants to answer the call. If so, she should be able to just press a button in the same TV remote control. Then the TV show will stop and the phone call will be connected. When the call finishes, the user just needs to press another button in the remote control to end the call and the screen will switch back to the TV show and resume from where it left.

Achieving a unified IPTV environment requires a unified communication interface. In the above example, the phone interface is unified into the TV interface: the TV screen will become a big digital phone screen during the phone call showing the caller photo (if available), name, phone number, call duration, etc. On the other hand, if several people are watching the TV show at the same time and the user does not want to disturb others' viewing experience, she might decide to pick up the traditional phone in the plain, old fashion.

An implication of unified interface is unified presence. The availability of a user should be unified so that the appliance, rather than humans, can find the appropriate method to locate the user. The well-known application with presence information is instant messaging: you can find the availability of your buddy in real time. Integrated into an IPTV environment, a unified interface should be able to indicate if and how the user can be reached. For example, a user in the middle of watching a hot movie may want to set her status to "Do not disturb". When a user sees her friends is available through the TV screen, she can initiate a video conference call with them through digital cameras and TVs instead of through computers. In some sense, the TVs are becoming more like computers or we can say the computers are becoming more like TVs.

6. RELATED WORK

Several measurement studies have been conducted on Internet media delivery in the educational [6, 14] or enterprise environment [13]. Characteristics such as object popularity distribution and evolution, media session duration, sharing patterns, etc., were modeled and analyzed. We and others characterized the media workload

collected from a large number of commercial Web sites hosted by a major ISP and that collected from a large group of home users connected to the Internet via a well-known cable company [22]. We found that the majority of media contents at that time were still delivered via downloading from Web servers and that a substantial percentage of media downloading connections were aborted before completion due to the long waiting time. Interactive operations in streaming sessions were studied in [6, 17, 25] as well as in our own work [23]. We found that jumps and pauses were the most frequent interactive operations initiated by clients and that jump operations often experience significant delays. Most of the Internet streaming media traffic today is delivered either through Windows media services or RealNetworks media services. Measurement studies in this area include analysis of RealNetwork audio [29] and video [36] as well as Windows media workloads [30]. User contributed content in YouTube has been analyzed in [9] and the benefit of P2P techniques is discussed.

Much work exists to measure and model P2P systems. Authors in [20] discovered the "download at most once" phenomenon in KaZaa traffic and believed it contributed to the fundamental difference in the observed traffic pattern to traditional Web traffic. BitTorrent is a new generation of P2P system which has become very popular. Most studies found that BitTorrent had good performance during "flash crowd" events, such as during the initial release of a popular software. All these studies, however, are based on the peer behaviors in a single torrent, while our analysis found that most peers participate in multiple torrents. We also revealed several limitations of BitTorrent-like systems: even though the system can handle flash crowd well, the duration of the flash crowd is actually quite short. After that there is an exponentially long tail during which the number of peers and seeds both decrease rapidly. Hence, if a peer joins a torrent late, it may have a hard time to locate and download a file. (The downloading failure ratio in such systems is non-trivial.) In addition, we found that the client performance in such systems can fluctuate substantially with the torrent population [24].

Related work has also been conducted in peer to peer streaming. Work [26] has shown that peer assisted VoD can offload server and reduce its bandwidth requirement significantly. It also explores the so-called "ISP-friendly" P2P algorithm – an approach that aims to minimize cross-ISP traffic if possible. Using the real network architecture in the AT&T Light Speed project, we and others have shown that P2P is not always beneficial and its benefit depends greatly on where the bottleneck of the system is [12]. If the capacity of the streaming servers is the bottleneck, then P2P can help. On the other hand, if the capacity of the local cable head-end is the bottleneck, then P2P can actually reduce system throughput. In our follow up work, we explored the idea of striping the data of a media object across multiple peers, essentially treating a P2P network as a large RAID array and using erasure code to overcome transient failures [11].

Internet resource provisioning was previously studied in the context of grid computing [19]. Unlike grid computing which mainly targets batch oriented scientific applications, our focus is on the three-tier based Internet streaming applications described earlier in the paper. Compared to traditional scientific applications, our applications are interactive in nature and must keep the response time short. A video stream that has a long start up latency or excessive jitters is unlikely to attract users. In addition, the computational requirement for each request is typically small, but an application may receive lots of simultaneous requests. An important metric in streaming server performance is the number of concurrent connections it can handle while satisfying a specific QoS requirement. Finally, our applications can be expected to run continuously: a popular streaming service receives requests from all over the world 24 by 7. Hence, any decision to replicate or migrate the application or to re-allocate the resources must be done in a seamless manner without disrupting the normal operation of the running application. This is very different from batch oriented scientific applications where a task is dispatched to a node and is expected to run there until completion.

7. CONCLUSION

The rapid growth of streaming traffic on the Internet and mobile networks brings a set of technical challenges. This paper has investigated some of those challenges with a focus on resource provisioning and utilization as well as the unique characteristics of mobile devices. As we pointed out throughout the paper, the requirements for streaming applications are significantly different from those for traditional Web applications. We believe that peer to peer technologies hold great promise to support novel applications like IPTV. In the future, we plan to develop and evaluate real Internet streaming services.

8. REFERENCES

[1] PPLive. http://www.pplive.com/.

[2] Realnetworks. http://www.real.com/.

[3] Sprint Mobile Video Coverage. `http://www2.sprint.com/mr/news_dtl.do?id=14980`.

[4] Windows Media home. http://www.microsoft.com/windows/windowsmedia/default.aspx.

[5] YouTube. http://www.youtube.com/.

[6] J. M. Almeida, J. Krueger, D. L. Eager, and M. K. Vernon. Analysis of educational media server workloads. In *Proc. of ACM Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV)*, June 2001.

[7] C. Canali, S. Fisher, M. Rabinovich, O. Spatscheck, and Z. Xiao. Snap: A shared network-aware platform for internet applications. Technical report, AT&T Research, 2005.

[8] C. Canali, M. Rabinovich, and Z. Xiao. *Recent Advances on Web Content Delivery*, chapter Utility Computing for Internet Applications. Springer-Verlag, Sept. 2005.

[9] M. Cha, H. Kwak, P. Rodriguez, Y. Ahn, and S. Moon. I tube, you tube, everybody tubes: Analyzing the world's largest user generated content video system. In *Proc. of ACM SIGCOMM Internet Measurement Conference*, Oct. 2007.

[10] M. Charikar and S. Guha. Improved combinatorial algorithms for the facility location and k-median problems. In *Proceedings of the 40th Annual Symposium on Foundations of Computer Science*, 1999.

[11] Y.-F. Chen, Y. Huang, R. Jana, H. Jiang, M. Rabinovich, J. Rahe, B. Wei, and Z. Xiao. Towards capacity and profit optimization of video-on-demand services in a peer-assisted IPTV platform. *To appear in the Multimedia Systems Journal*, 2008.

[12] Y.-F. Chen, Y. Huang, R. Jana, H. Jiang, M. Rabinovich, B. Wei, and Z. Xiao. When is P2P technology beneficial to iptv services? In *Proc. of the 17th International workshop on Network and Operating Systems Support for Digital Audio & Video (NOSSDAV'07)*, June 2007.

[13] L. Cherkasova and M. Gupta. Characterizing locality, evolution, and life span of accesses in enterprise media server workloads. In *Proc. of ACM Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV)*, May 2002.

[14] M. Chesire, A. Wolman, G. Voelker, and H. Levy. Measurement and analysis of a streaming media workload. In *Proc. of the 3rd USENIX Symposium on Internet Technologies and Systems*, March 2001.

[15] Y. Chu and H. Zhang. Considering altruism in peer-to-peer Internet streaming broadcast. In *Proceedings of ACM NOSSDAV*, June 2004.

[16] F. A. Chudak and D. B. Shmoys. Improved approximation algorithms for capacitated facility location problem. In *Proceedings of the 10th ACM Symposium on Discrete Algorithms*, 1999.

[17] C. Costa, I. Cunha, A. Borges, C. Ramos, M. Rocha, J. Almeida, and B. Ribeiro-Neto. Analyzing client interactivity in streaming media. In *Proc. of the International World Wide Web Conference*, May 2004.

[18] K. Fitchard. Tv wars go wireless. `http://telephonyonline.com/wireless/technology/telecom_tv_wars_go/`.

[19] I. Foster, C. Kesselman, and S. Tuecke. The anatomy of the Grid: Enabling scalable virtual organizations. *International J. Supercomputer Applications*, 15(3), 2001.

[20] K. P. Gummadi, R. J. Dunn, S. Saroiu, S. D. Gribble, H. M. Levy, and J. Zahorjan. Measurement, modeling, and analysis of a peer-to-peer file-sharing workload. In *Proc. of ACM Symposium on Operating Systems Principles (SOSP)*, October 2003.

[21] L. Guo, S. Chen, Z. Xiao, E. Tan, X. Ding, and X. Zhang. Measurements, analysis, and modeling of bittorrent systems. In *Proc. of ACM SIGCOMM Internet Measurement Conference (IMC'05)*, Oct. 2005.

[22] L. Guo, S. Chen, Z. Xiao, and X. Zhang. Analysis of multimedia workloads with implications for Internet streaming. In *Proc. of the 14th International World Wide Web Conference (WWW'05)*, May 2005.

[23] L. Guo, S. Chen, Z. Xiao, and X. Zhang. DISC: Dynamic interleaved segment caching for interactive streaming accesses. In *Proc. of the 25th International Conference on Distributed Computing Systems (ICDCS'05)*, June 2005.

[24] L. Guo, E. Tan, S. Chen, Z. Xiao, O. Spatscheck, and X. Zhang. Delving into internet streaming media delivery: A quality and resource utilization perspective. In *Proc. of ACM SIGCOMM Internet Measurement Conference (IMC'06)*, Oct. 2006.

[25] L. He, J. Grudin, and A. Gupta. Designing presentations for on-demand viewing. In *Proc. of ACM Conference on Computer Supported Cooperative Work*, December 2000.

[26] C. Huang, J. Li, and K. W. Ross. Can internet video-on-demand be profitable. In *Proceedings of the ACM SIGCOMM*, Aug. 2007.

[27] Y. Huang, Y.-F. Chen, R. Jana, H. Jiang, M. Rabinovich, A. Reibman, B. Wei, and Z. Xiao. Capacity analysis of mediagrid: a P2P iptv platform for fiber to the node (fttn) networks. *IEEE Journal on Selected Areas in Communication (JSAC), Special Issue on Peer-to-Peer Communications and Applications*, 25(1), Jan. 2007.

[28] K. Jain and V. V. Vazirani. Approximation algorithms for metric facility location and k-median problems using the primal-dual schema and lagrangian relaxation. *Journal of the ACM*, Mar. 2001.

[29] A. Mena and J. Heidemann. An empirical study of real audio traffic. In *Proc. of IEEE INFOCOM*, March 2000.

[30] J. Nichols, M. Claypool, R. Kinicki, and M. Li. Measurements of the congestion responsiveness of windows streaming media. In *Proc. of ACM Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV)*, June 2004.

[31] R. Pieck. DVG-H Broadcast to Mobile Devices. `http://www.newtec.eu/index.php?id=485`, 2005.

[32] Qualcomm. FLO Technology Overview. `http://www.qualcomm.com/common/documents/brochures/tech_overview.pdf`, 2007.

[33] M. Rabinovich, Z. Xiao, and A. Aggarwal. Computing on the edge: A platform for replicating Internet applications. In *Proceedings of the Eighth International Workshop on Web Content Caching and Distribution*, Sept. 2003.

[34] M. Rabinovich, Z. Xiao, F. Douglis, and C. Kalmanek. Moving edge-side includes to the real edge—the clients. In *Proceedings of the 4th USENIX Symposium on Internet Technologies and Systems*, Mar. 2003.

[35] G. Raria, J. Henriksson, E. Stare, and P. Talmola. DVB-H, Digital Broadcast Services to Handheld Devices. *Proceedings of the IEEE*, January 2006.

[36] Y. Wang, M. Claypool, and Z. Zuo. An empirical study of realvideo performance across the internet. In *Proc. of the ACM SIGCOMM Internet Measurement Workshop (IMW)*, November 2001.