Energy Efficient Cloud based Smart MOV (Mobile social TV system)

Pradeep B.M[1]
Dept of computer science and engineering.
GSS Institute of technology.
Bangalore, Karnataka, India.
[1]Taurus.bmp@gmail.com.

Kruthika S.G[2]
Dept of computer science and engineering.
GSS Institute of technology.
Bangalore, Karnataka, India.
[2]sgkruthi@gmail.com.

***Abstract*— The rapidly increasing power of personal mobile devices such as smart phones, tablets, etc. is providing much richer contents and social interactions to users on the move. This trend however is throttled by the limited battery life of mobile devices and unstable wireless connectivity, achieving the highest possible quality of service experienced by mobile users not feasible. In this paper, we propose the design of a Cloud-based, Mobile *Smart MOV*. This system effectively utilizes both *SaaS (Software-as-a-Service)* and *PaaS (Platform-as-a-Service)* cloud delivery services to offer the living-room experience of video watching to a group of disparate mobile users who can interact socially while watching and sharing the video. We employ a surrogate for each user in the SaaS cloud for video downloading and social exchanges on behalf of the user. These various designs for flexible transcoding capabilities, battery efficiency of mobile devices and spontaneous social interactivity together provide an ideal platform for smart MOV services.**

Keywords: Computers and information processing, Mobile computing, Communications technology, TV, Mobile TV.

I. INTRODUCTION

THANKS to the revolutionary "reinventing the phone" Campaigns initiated by Apple Inc. In 2007, Smartphones and Tablets nowadays are shipped with multiple microprocessor cores and gigabyte RAMs; they possess more computation power than personal computers of a few years ago. On the other hand, the wide deployment of 3G broadband cellular infrastructures further fuels the trend. Apart from common productivity tasks like emails and web surfing, smartphones are flexing their strength in more challenging scenarios such as real time video streaming and online gaming, as well as serving as a main tool for social exchanges. It is natural to resort to cloud computing, the newly-emerged computing paradigm for low-cost agile, scalable resource supply, to support power-efficient mobile data communication. With virtually infinite hardware and software resources, the cloud can offload the computation and other tasks involved in a mobile application and may significantly reduce battery consumption of the mobile devices, if a proper design is in place. Tough challenge in front of us is how to effectively exploit cloud services to facilitate mobile applications. In this paper, we describe the design of a novel mobile social TV system, "***Smart MOV***", which can effectively utilize the cloud computing paradigm to offer a living-room experience of video watching to disparate mobile users with spontaneous social interactions. In *Smart MOV*, mobile users can import a live or on-demand video to watch from any video streaming site, invite their friends to watch the video concurrently, and chat with their friends while watching the video. As opposed to traditional TV watching, mobile social TV is well suited to today's Traditional life style, where family and friends may be distributed geographically but hope to share a *co-viewing* experience. We design *Smart MOV* to seamlessly utilize agile resource support and rich functionalities offered by both an SaaS (Infrastructure- as-a-Service) cloud and a PaaS (Platform-as-a-Service) cloud. Our design achieves the following goals.

A. Encoding Flexibility:

Different mobile devices have various sized displays, customized playback hardwares, and various codecs. Traditional solutions would adopt a few encoding formats ahead of the release of a video program. But even the most generous content providers would not be able to attend to all possible mobile platforms, if not only to the current hottest models. *Smart MOV* customizes the streams for different devices in real time, by offloading the transcoding tasks to an SaaS cloud.

B. Battery Efficiency:

A breakdown analysis conducted by Carroll and G. Heiser [6] indicates that the network modules (such as Wi-Fi and 3G) and the display contribute to a significant portion of the overall power consumption in a mobile device, dwarfing usages from other hardware modules including CPU, memory, etc. We target at energy saving coming from the network module of smart phones through an efficient data transmission mechanism design. We focus on 3G wireless networking as it is getting more widely used and challenging in our design than Wi-Fi based transmissions.

C. Spontaneous Social Interactivity:

Multiple mechanisms are included in the design of *Smart MOV* to enable spontaneous social interactivity, co-viewing experience. *First*, efficient synchronization mechanisms are proposed to guarantee that friends joining in a video program may watch the same portion (if they choose to), and share immediate reactions and comments. Although synchronized playback is inherently a feature of traditional TV, the current

Internet video services (*e.g.*, Web 2.0 TV) rarely offer such a service. *Second*, efficient message communication mechanisms are designed for social, textual interactions among friends, and different types of messages are prioritized in their retrieval frequencies to avoid unnecessary interruptions of the viewing progress.

D. Portability:

A prototype *Smart MOV* system is implemented following the philosophy of "Write Once, Run Anywhere" (WORA) "100% Pure Java" platform used to implement both the front-end mobile modules and the back-end server modules.

II. RELATED WORK

A huge amount of mobile TV systems has sprung up in recent years, driven by both hardware and software advances in mobile devices. Some early systems bring the "living-room" experience to small screens on the move. But they focus more on barrier clearance in order to realize the convergence of the mobile network and the television network, rather than exploring the demand of "social interactions" among mobile users. Coppens *et al.* [4] try to add rich social interactions to TV, but their design is limited to traditional broadcast program channels. Though inspiring, these designs are not that suitable for being applied directly in a mobile environment. Chuah *et al.* [6] extend the social experiences of viewing traditional broadcast programs to mobile devices, but have yet to deliver a well integrated framework. Schatz *et al.* [7], [8] have designed a mobile social TV system, which is customized for DVB-H networks and Symbian devices as opposed to a wider audience. Compared to these prior work and systems, we target at a design for a generic, portable mobile social TV framework, featuring "*co-viewing* experiences" among friends over geographical separations through mobile devices. Our framework is open to all Internet-based video programs, either live or on-demand, and supports a wide range of devices with HTML5 compatible browsers installed, without any other mandatory component on the devices. For any application targeted at mobile devices, consumes less power is perennially one of the major concerns and challenges. Our work is able to achieve a significant (about 30%) power saving, by opportunistically switching the device between low-power and high-power transmission modes during streaming. Some existing work (*e.g.*, Anastasi *et al.* [9]) have provided valuable guidelines for energy saving over WiFi transmissions; our work focuses on 3G cellular transmissions which have significantly different power models; 3G is a more practical wireless connection technology for mobile TVs on the go at the present time. Cloud computing had its debut with much fanfare and is now deemed a most powerful hosting platform in many areas including mobile computing. Satyanarayan an *et al.* [1] suggest offloading mobile devices computation workload to a nearby resource-rich infrastructure (*i.e., Cloudlets*) by dynamic VM synthesis. Kosta *et al.* [2] propose a virtualization framework for mobile code offloading to the cloud. Zhang *et al.* [10] introduce an elastic mobile application model by offloading part of the applications (*weblets*) to an SaaS cloud. Recently, attentions have been drawn to enabling media applications using the cloud, for both media storage [11] and processing [12] Conversely, the prototype we implement is browser-based and platform independent; it supports both live channels, VoD channels and personal channels hosted by any user, with wider usage ranges and flexible extensibility.

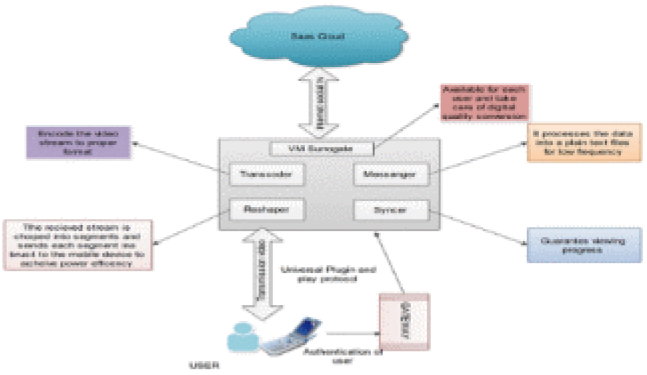


Fig 1: The architecture of *Smart MOV*.

III. Smart MOV: ARCHITECTURE AND DESIGN

As a novel Cloud-based Mobile *Smart MOV* system, *Smart MOV* provides two major functionalities to participating mobile users:

1) Universal streaming: A user can stream a live or on-demand video from any video sources he chooses, such as a TV program provider or an Internet video streaming site, with tailored encoding formats and rates for the device each time.

2) Co-viewing with social exchanges: A user can invite multiple friends to watch the same video, and exchange text messages while watching. The group of friends watching the same video is referred to as a *session*.

Key Modules:

Fig1 gives an overview of the architecture of *Smart MOV*. A *surrogate* (*i.e.*, a virtual machine (VM) instance), or a *VM surrogate* equivalently, is created for each online mobile user in an SaaS cloud infrastructure. The surrogate acts as a proxy between the mobile device and the video sources, providing transcoding services as well as segmenting the streaming traffic for burst transmission to the user. The surrogates exchange social messages via a back-end PaaS cloud, which adds scalability and robustness to the system. There is a gateway server in *Smart MOV* that keeps track of participating users and their VM surrogates, which can be implemented by a standalone server or VMs in the SaaS cloud. The design of *Smart MOV* can be divided into the following major functional modules.

• **Transcoder:** It resides in each surrogate, and is responsible for dynamically deciding how to encode the video stream from the video source in the recommended format, dimension, and bit rate. Before deliver to the user, the video stream is further encapsulated into a proper transport stream. In our implementation, each video is exported as MPEG-2 transport streams, which is the de-facto standard nowadays to deliver digital video and audio streams over lossy medium.

• **Reshaper:** The reshaper in each surrogate receives the encoded transport stream from the transcoder, chops it into Segments, and then sends each segment in a burst to the mobile device upon its request (*i.e.*, a burst transmission mechanism), to achieve the best power efficiency of the device.
• **Messenger:** The messenger is the client part of the social cloud, residing in each surrogate in the SaaS cloud. The Messenger periodically queries the social cloud for the social data on behalf of the mobile user and pre-processes the data into a light-weighted format (plain text files), at a much lower frequency. The plain text files (in XML formats) are asynchronously delivered from the surrogate to the user in a traffic-friendly manner, *i.e.*, little traffic is incurred.
• **Syncer:** The syncer on a surrogate guarantees that viewing progress of this user is within a time window of other users in the same session (if the user chooses to synchronize with others). To achieve this, the syncer periodically retrieves the current playback progress of the session host and instructs its mobile user to adjust its playback position. In this way, friends can enjoy the "sitting together" viewing experience.
• **Gateway:** The gateway provides authentication services for users to log in to the *Smart T.V* system, and stores users credentials in a permanent table of a MySQL database it has installed.

A. Loosely Coupled Interfaces

Similar in spirit to web services, the interfaces between different modules in *Smart MOV*, *i.e.*, mobile users, VM surrogates, and the social cloud, are based on HTTP, a universal standard for all Internet-connected devices or platforms. Thanks to the loose coupling between users and the infrastructure, almost any mobile device is ready to gain access to the *Smart MOV* services, as long as it is installed with an HTTP5 browser.

B. Pipelined Video Processing

Both live streaming of real time contents and on-demand streaming of stored contents are supported in *Smart MOV*. Video processing in each surrogate is designed to work on the fly, *i.e.*, the transcoder conducts real time encoding from the video source, the encoded video is fed immediately into the reshaper for segmentation and transmission, and a mobile user can start viewing the video as soon as the first segment is received. To support dynamic bit rate switch, the transcoder launches multiple threads to transcode the video into multiple bit rates once the connection speed between the surrogate and the mobile user changes.

C. Burst Size:

To decide the burst size, *i.e.*, the size of the segment transmitted in one burst, we need to take into consideration characteristics of mobile streaming and energy consumption during state transitions. For video streaming using a fixed device without power concerns, it is desirable to download as much of a video as what the connection bandwidth allows; however, for streaming over a cellular network, leading to a waste of the battery power and the cellular data fee due to their download. Hence, the burst size should be kept small, to minimize battery consumption and traffic charges.

IV. Smart MOV: PROTOTYPE IMPLEMENTATION

Following the design guidelines in Section III, we have implemented a real-world mobile social TV system, and deployed it on the Google App Engine (GAE) and Amazon EC2 clouds, which are the two most widely used public PaaS and SaaS cloud platforms. GAE, as a PaaS cloud, provides rich services on top of Google's data centers and enables rapid deployment of Java-based and Python-based applications. Hence, GAE is an ideal platform for implementing our social cloud, which dynamically handles large volumes of messages. On the other hand, GAE imposes many constraints on application deployment, *e.g.*, lack of support for multi-threading, file storage, etc.

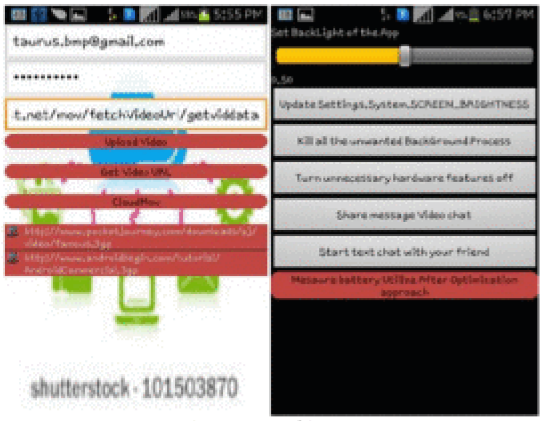
A. Client Use of Smart MOV

All mobile devices installed with HTML5 compatible browsers can use *Smart MOV* services, as long as the HTTP Live Streaming (HLS) [13] protocol is supported. The user first connects to his Gmail page from *Smart MOV*, as illustrated in the top left corner of Fig 2(a). After the user successfully logs in through the gateway, he is assigned a VM surrogate from the VM pool (the hostnames of available VMs, *e.g.*,ec2-50-16-xx-xx.compute-1.amazonaws.com, maintained in an in-memory table of a MySQL database deployed in the gateway). Then the user is automatically redirected to the assigned VM surrogate, and welcomed by a portal page as shown on the right-hand side of Fig. 2. Upon user login, the portal collects the device configuration information by examining the "User-Agent" header values, and this information will be sent to its surrogate for decision making of the video encoding formats. The user can enter the URL of the video or live broadcast he wishes to watch, on the "Subscribe" tab of the portal; after he clicks the "Subscribe" button, the address of the video is sent to the VM surrogate, which downloads the stream on the user's behalf, transcodes and sends properly encoded segments to the user. Invite one or more friends to join him in watching the video. Users in the same session can exchange opinions and comments on the "Chat" tab (a snapshot is given in Fig. 3(a) and (b)), where new chat messages can be entered and the chat history of the session is shown.

B. VM Surrogates

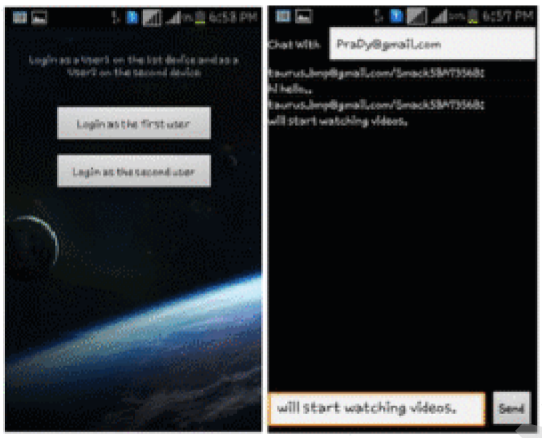
All the VM surrogates are provisioned from Amazon EC2 web services and tracked by the gateway. We create our own AMI (ami-b6f220df) based on Linux kernel 2.6.35.14, the default image Amazon provides. Due to the intensive computation involved, we propose to implement all the video processing related tasks using ANSI C, to guarantee the performance. We have also installed a Tomcat web server (version 6.5) to serve as a Servlet container and a file server on each surrogate. Both FFmpeg and Tomcat are open source projects. Once a VM surrogate receives a video subscription request from the user,

it downloads the video from the source URL, and processes the video stream by transcoding and segmentation, based on the collected device configurations by the portal.



(a) (b)

Fig 2: UI Screen of "**Smart MOV**" (a) UI screen (b) Options menu



(a) (b)

Fig 3: Video and text chat tabs of *Smart T.V* (a) "Video chat tab" (b) "Text Chat tab".

C. *Data Models in the Social Cloud:*

We use GAE mainly as the back-end data store to keep the Transient states and data of *Smart MOV*, including users online presence status, social messages (invitation and chat messages) in all the sessions. With Jetty as the underlying Servlet container, most Java-based applications can be easily migrated to GAE, under limited usage constraints, where no platform-specific APIs are enforced for the deployment. GAE provides both its Java Persistence API.

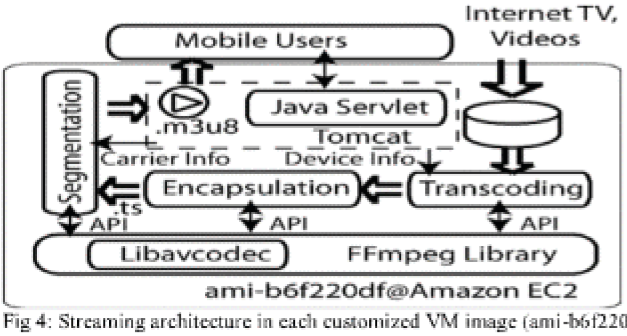


Fig 4: Streaming architecture in each customized VM image (ami-b6f220df).

V. REAL-WORLD EXPERIMENTS

We carry out both unit tests and performance evaluations of *Smart MOV*. The experiments are conducted over the 3G cellular network of 3HK, which is one of the largest Telecom operators in Hong Kong. Fig. 5 shows the power consumption levels on the phone over time, in terms of portions of the highest device power level. The red vertical lines represent the starting points of playback periods when the Safari runs in the foreground, and the green lines represent the finish times of playback periods when the Safari is suspended in the background. When there is data transmission, the device operates at the high power mode; when data transmission stops, the transmission power of the device first decreases to an intermediate level, and then to a very low level.

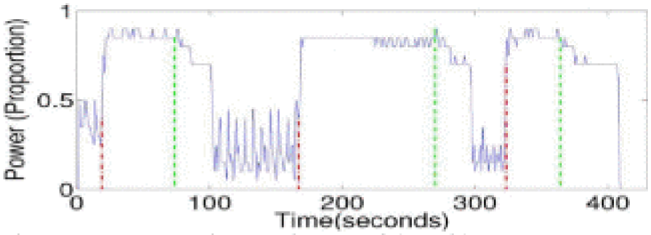


Fig 5: Power consumption over time on an iPhone 4S.

VI. CONCLUDING REMARKS AND FUTURE WORK

This paper presents our view of what might become a trend for mobile TV, *i.e.*, mobile social TV based on agile resource supports and rich functionalities of cloud computing services. We introduce a generic and portable mobile social TV framework, *Smart MOV* that makes use of both an SaaS cloud and a PaaS cloud. The framework provides efficient transcoding services for most platforms under various network conditions and supports for co-viewing experiences through timely chat exchanges among the viewing users. By employing one surrogate VM for each mobile user, we achieve ultimate scalability of the system. Through an in-depth investigation of the power states in commercial 3G cellular networks, we then propose an energy- efficient burst transmission mechanism that can effectively increase the battery lifetime of user devices. We have implemented a realistic prototype of *Smart MOV*, deployed on Amazon EC2 and Google App Engine, where EC2 instances serve as the mobile users' surrogates and GAE as the social cloud to handle the large volumes of social message exchanges. The experimental results prove the superior performance of *Smart MOV*, in terms of transcoding efficiency, power saving, timely social interaction, and scalability. In our future work, such sharing can be enabled and carried out in a peer-to-peer fashion, *e.g.*, the surrogate of a newly joined user may fetch the transcoded streams directly from other surrogates, if they are encoded in the format/bit rate that the new user wants. For implementing social interactions, most BigTable- like data stores (including GAE) support memcache which is a highly efficient secondary storage on the data stores.

REFERENCES

[1] M. Satyanarayanan, P. Bahl, R. Caceres, and N. Davies, "The case for VM-based Cloudlets in mobile computing," *IEEE Pervasive Comput.*, vol. 8, pp. 14–23, 2009.'

[2] S. Kosta, A. Aucinas, P. Hui, R. Mortier, and X. Zhang, "Thinkair: Dynamic resource allocation and parallel execution in the cloud for mobile code offloading," in *Proc. IEEE INFOCOM*, 2012.

[3] T. Coppens, L. Trappeniers, and M. Godon, "AmigoTV: Towards a social TV experience," in *Proc. EuroITV*, 2004.

[4] N. Ducheneaut, R. J. Moore, L. Oehlberg, J. D. Thornton, and E. Nickell, "Social TV: Designing for distributed, sociable television viewing," *Int. J. Human-Comput. Interaction*, vol. 24, no. 2, pp. 136–154, 2008.

[5] What is 100% Pure Java. [Online]. Available: ttp://www.javacoffeebreak.com/faq/faq0006.html.

[6] M. Chuah, "Reality instant messaging: Injecting a dose of reality into online chat," in *CHI '03 Extended Abstracts on Human Factors in Computing Syst.*, 2003, ser. CHI EA '03, pp. 926–927.

[7] R. Schatz, S. Wagner, S. Egger, and N. Jordan, "Mobile TV becomes social – Integrating content with communications," in *Proc. ITI*, 2007.

[8] R. Schatz and S. Egger, "Social interaction features for mobile TV services," in *Proc. 2008 IEEE Int. Symp. Broadband Multimedia Syst. And Broadcasting*, 2008.

[9] G. Anastasi, M. Conti, E. Gregori, and A. Passarella, "Saving energy in Wi-Fi hotspots through 802.11 psm: An analytical model," in *Proc Workshop Linguistic Theory and Grammar Implementation, ESSLLI 2000*, 2004, pp. 24–26.

[10] X. Zhang, A. Kunjithapatham, S. Jeong, and S. Gibbs, "Towards an elastic application model for augmenting the computing capabilities of mobile devices with cloud computing," *Mobile Netw. Applicat.*, pp. 1–15, Apr. 2011.

[11] W. Zhu, C. Luo, J. Wang, and S. Li, "Multimedia cloud computing," *IEEE Signal Process. Mag.*, vol. 28, pp. 59–69, 2011.

[12] R. Pereira and K. Breitman, "A cloud based architecture for improving video compression time efficiency: The split & merge approach," in *Proc. DCC'11*, 2011, pp. 471–471.