CloudMoV: Cloud-Based Mobile Social TV

Mrs. Sowbhagya M P

Asst. Professor, Dept. of ISE,
City Engineering College,
Bangalore
Visvesvaraya Technological
University, Karnataka, INDIA
sowbhagya.mp@gmail.com

Ms. Manisha

Student, Department of ISE,
City Engineering College,
Bangalore
Visvesvaraya Technological
University, Karnataka, INDIA
manishanaik7992@gmail.com

Ms. Samata

Student, Department of ISE,
City Engineering College,
Bangalore
Visvesvaraya Technological University,
Karnataka, INDIA
samatasuma@gmail.com

Ms. Harshitha

Student, Department of ISE,
City Engineering College,
Bangalore
Visvesvaraya Technological
University, Karnataka, INDIA
Harshitha_lingesh@yahoo.com

Abstract-The rapidly increasing power of personal mobiledevices (smartphones, tablets, etc.) is providing much richercontents and social interactions to users on the move. This trendhowever is throttled by the limited battery lifetime of mobiledevices and unstable wireless connectivity, making the highestpossible quality of service experienced by mobile users notfeasible. The recent cloud computing technology, with its richresources to compensate for the limitations of mobile devices andconnections, can potentially provide an ideal platform to supportthe desired mobile services. Tough challenges arise on how toeffectively exploit cloud resources to facilitate mobile services, especially those with stringent interaction delay requirements. Inthis paper, we propose the design of a Cloud-based, novel MobilesocialTV system (CloudMoV). The system effectively utilizes both PaaS (Platform-as-a-Service) and IaaS (Infrastructure-as-a-Service) cloud services to offer the living-room experience of video watching to a group of disparate mobile users who can interact socially while sharing the video. To guarantee good streaming quality as experienced by the mobile users with time-varying wireless connectivity, we employ a surrogate for each user in the IaaS cloud for video downloading and social exchanges on behalf of the user. The surrogate performs efficient stream transcoding that matches the current connectivity quality of the mobile user. Given the battery life as a key performance bottleneck, we advocate the use of burst transmission from the surrogates to the mobile users, and carefully decide the burst size which can lead to high energy efficiency and streaming quality.

1. INTRODUCTION

Although many mobile social or media applications have emerged, truely killer ones gaining mass acceptance are stillimpeded by the limitations of the current mobile and wirelesstechnologies, among which battery lifetime and unstable connection bandwidth are the most difficult ones. It is natural to resort to cloud computing, the newly-emerged computing paradigm for low-cost, and 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 at the mobile devices, if a proper design is in place. The big challenge in front of us is how to effectively exploit cloud services to facilitate mobile applications. There have been a few studies on designing mobile cloud computing systems [1]–[3], but none of them deal in particular with stringent delay requirements for spontaneous social interactivity among mobile users. In this paper, we describe the design of a novel mobile social TV system, *CloudMoV*, 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 *CloudMoV*, 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 enjoying the video. It therefore blends viewing experience and social awareness among friends on the go. As opposed to traditional TV watching, mobile social TV iswell suited to today's life style, where family and friends maybe separated geographically but hope to share a *co-viewing* experience. While social TV enabled by set-top boxes over thetraditional TV systems is already available [4], [5], it remains a challenge to achieve mobile social TV, where the concurrently viewing experience with friends is enabled on mobile devices. We design *CloudMoV* to seamlessly utilize agile resource support and rich functionalities offered by both an IaaS (Infrastructure-as-a-Service) cloud and a PaaS (Platform-as-a-Service) cloud. Our design achieves the following goals encoding flexibility, battery efficiency,spontaneous social interactivity,portability.

2. RELATED WORK

A number of mobile TV systems have sprung up in recentyears, driven by both hardware and software advances in mobile devices. Some early systems [6], [7] 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 television network and the mobile network, than exploring the demand of "social" interactions among mobile users. There is another trend in which efforts are dedicated to extending social elements to television systems [4], [5], [8]. Coppens *et al.* [4] try to add rich socialinteractions to TV but their design is limited to traditional broadcast program channels. Oehllber*g*[5] conduct a series of experiments on human social activitieswhile watching different kinds of programs. Though inspiring,these designs are not that suitable for being applied directly ina mobile environment. Chuah *et al.* [9] extend the social experiences of viewing traditional broadcast programs to mobile devices, but have yet to deliver a well-integrated framework. Schatz *et al.* [10], [11] have designed amobile social TVsystem, 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 deviceswith HTML5 compatible browsers installed, without any othermandatory component on the devices.A recent work by investigates the media caching management problem under HTTP adaptive bit rate streaming over a wireless network environment, which cancomplement our work when video streams are required to betranscoded into multiple bit rates. Finally, we are aware of the lack of a richly-featured cloud based mobile social TV system in real life. The only system coming close to ours is Live Stream on the iOS platform. This iOS-locked application only supports live video channels, and all its social functions are bound to Facebook open APIs. Conversely, the prototype we implement is browser-based and platform independent; it supports both live channels, VoD channels and even personal channels hosted by any user, with wider usage ranges and flexible extensibility. The framework we propose can be readily applied to other cloud-assistedmobile mediaapplications as well.

3. CLOUDMOV: ARCHITECTURE AND DESIGN

As a novel Cloud-basedMobile socialTV system, *CloudMoV* 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*. The mobile user who initiates a session is the *host* of the session. We present the architecture of *CloudMoV* and the detailed designs of the different modules in the following.
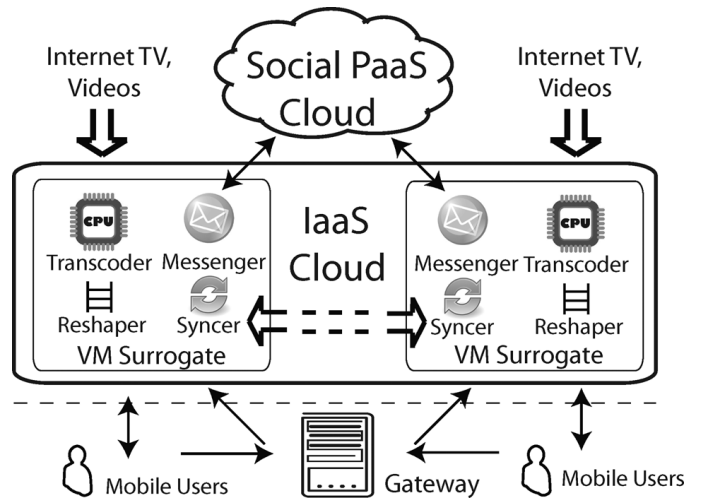

Fig. 3. The architecture of CloudMoV

A. Key Modules

Fig. 1 gives an overview of the architecture of *CloudMoV*.A *surrogate* (*i.e.*, a virtual machine (VM) instance), or a *VMSurrogate*equivalently,iscreatedforeachonlinemobileuserin anIaaS cloud infrastructure. The surrogate acts as a proxy betweenthe mobile device and the video sources, providing transcodingservices as well as segmenting the streaming traffic for bursttransmission to the user. Besides, they are also responsiblefor handling frequently exchanged social messages amongtheir corresponding users in a timely and efficient manner,shielding mobile devices from unnecessary traffic and enablingbattery efficient, spontaneous social interactions. The surrogatesexchange social messages via a back-end PaaS cloud, whichadds scalability and robustness to the system. There is agateway server in *CloudMoV* that keeps track of participatingusers and their VM surrogates, which can be implementedby a standalone server or VMs in the IaaS cloud.The design of *CloudMoV* can be divided into the followingmajor functional modules.

• Transcoder. It resides in each surrogate, and is responsiblefor dynamically deciding how to encode the videostream from the video source in the appropriate format, dimension, and bit rate. Before delivery to the user, thevideo stream is further encapsulated into a proper transport stream. In our implementation, each video is exportedas MPEG-2 transport streams, which is the de facto standardnowadays to deliver digital video and audio streamsover lossy medium.

• Reshaper. The reshaper in each surrogate receives the encodedtransport stream from the transcoder, chops it intosegments, and then sends each segment in a burst to themobile device upon its request (*i.e.*, a burst transmissionmechanism), to achieve the best power efficiency of the device.The burst size, *i.e.*, the amount of data in each burst,is carefully decided according to the 3G technologies implementedby the corresponding carrier.

• Social Cloud. The social cloud is built on top of any general PaaS cloud services with BigTable-like data store toyield better economies of scale without being locked downto any

specific proprietary platforms. Despite its implementationon Google App Engine (GAE) as a proof of concept,our prototype can be readily ported to other platforms.It stores all the social data in the system, including the onlinestatuses of all users, records of the existing sessions,and messages (invitations and chat histories) in each session.The social data are categorized into different *kinds*and split into different *entities* (in analogy to *tables* and*rows* in traditional relational database, respectively) [13].The social cloud is queried from time to time by the VMsurrogates.

• Messenger. It is the client side of the social cloud, residingin each surrogate in the IaaS cloud. The Messenger periodicallyqueries the social cloud for the social data on behalfof the mobile user and pre-processes the data into alight-weighted format (plain text files), at a much lowerfrequency. The plain text files (in XML formats) are asynchronouslydelivered from the surrogate to the user in atraffic-friendly manner, *i.e.*, little traffic is incurred. In thereverse direction, the messenger disseminates this user'smessages (invitations and chat messages) to other users viathe data store of the social cloud.

• Syncer. The syncer on a surrogate guarantees that viewingprogress of this user is within a time window of other usersin the same session (if the user chooses to synchronize withothers). To achieve this, the syncer periodically retrievesthe current playback progress of the session host and instructsits mobile user to adjust its playback position. In thisway, friends can enjoy the "sitting together" viewing experience.Different from the design of communication amongmessagers, syncers on different VM surrogates communicatedirectly with each other as only limited amounts oftraffic are involved.

• Mobile Client. The mobile client is not required to installany specific client software in order to use *CloudMoV*, aslong as it has an HTML5 compatible browser (*e.g.*, MobileSafari, Chrome, etc.)and supports the HTTP Live streaming protocol [14]. Both are widely supported onmost state-of-the-art smartphones.

• Gateway. The gateway provides authentication servicesfor users to log in to the *CloudMoV* system, and storesusers' credentials in a permanent table of a MySQL databaseit has installed. It also stores information of the poolof currently available VMs in the IaaS cloud in anotherin-memory table. After a user successfully logs in to thesystem, a VM surrogate will be assigned from the pool tothe user. The in-memory table is used to guarantee smallquery latencies, since the VM pool is updated frequentlyas the gateway reserves and destroys VM instances accordingto the current workload. In addition, the gatewayalso stores each user's friend list in a plain text file (in XMLformats), which is immediately uploaded to the surrogate

after it is assigned to the user.

We describe the key designs in *CloudMoV*in the following.

B. Loosely Coupled Interfaces

Similar in spirit to web services, the interfaces between differentmodules in *CloudMoV*, *i.e.*, mobile users, VM surrogates,and the social cloud, are based onHTTP, a universal standard forall Internet-connected devices or platforms. Thanks to the loosecoupling between users and the infrastructure, almost any mobiledevice is ready to gain access

to the *CloudMoV* services, aslong as it is installed with an HTTP browser. The VMsurrogatesprovisioned in the IaaS cloud cooperate with the social cloudimplemented on a PaaS cloud service via HTTP as well, with noknowledge of the inner components and underlying technologiesof each other, which contributes significantly to the portabilityand easy maintenance of the system.For socialmessage exchanges among friends, *CloudMoV* employsasynchronous communication. All the exchanged messagesare routed via the surrogates to the social cloud, whichefficiently organizes and stores the large volumes of data in a Big Table-like data store. The VM surrogates query the socialcloud frequently and processes the retrieved data into XMLfiles, for later retrieval by users in an asynchronous fashion.Such a design effectively separates the mobile users from thesocial cloud to significantly simplify the architecture, while theextra delay introduced at the VM surrogates is ignorable, asshown in Section V.

C. Pipelined Video Processing

Both live streaming of real time contents and on-demandstreaming of stored contents are supported in *CloudMoV*.Video processing in each surrogate is designed to work onthe fly, *i.e.*, the transcoder conducts real time encoding fromthe video source, the encoded video is fed immediately intothe reshaper for segmentation and transmission, and a mobileuser can start viewing the video as soon as the first segmentis received. To support dynamic bit rate switch, the transcoderlaunches multiple threads to transcode the video into multiplebit rates once the connection speed between the surrogate andthe mobile user changes. The IaaS cloud where the surrogatesare deployed, represents an ideal platform for implementingsuch computation intensive jobs.

D. Burst Transmissions

1) 3G Power States: Different from Wi-Fi which is moresimilar to the LANed Internet access, 3G cellular services sufferfrom the limited radio resources, and therefore each user equipment

(UE) needs to be regulated by a Radio Resource Control(RRC) state machine [15]. Different 3G carriers may customizeand deploy complex states in their respective cellularnetworks. Different states indicate different levels of allocated radio resources, and hence different levels of energy consumptions.For ease of implementation, we consider three basic statesin our design, which are commonly employed by many carriers,namely CELL_DCH (a dedicated physical channel is allocated

to the UE in both the uplink and the downlink), CELL_FACH (no dedicated channel is allocated but the UE is assigned a defaultcommon transport channel in the uplink), and IDLE, in decreasing order of power levels [15]. Contrary to intuition,the energy consumption for data transmission depends largelyon the state a UE is working in, but has little to do with thevolume of data transmitted, *i.e.*, a UE may stay at a high powerstate (CELL_DCH) for data transmission even the data rate isvery low [15] (this has also been verified in our experiments inSection V).A 3G carrier may commonly transfer a UE froma high-powerstate to a low-power state (state demotion), for releasing radiochannels allocated to this UE to other users. For example, ifa UE working at a high-

power state does not incur any datatraffic for a pre-configured period of time (measuredby a criticalinactivity timer), the state of the UE will be

transferred toa low-power one; when the volume of data traffic rises, the UE"wakes up" from a low-power state and moves to a high-powerone. Timeouts of the critical inactivity timers for state transitionsare properly set by the carrier to guarantee performancein both delay and energy consumption, since extra delay andenergy consumption are potentially incurred for acquiring newradio channels when the UE transits from a low-power state toa high-power one later (state promotion).

*2) Transmission Mechanism:*In *CloudMoV*, we aimatmaximumconservation of the battery capacity of the mobile device,and design a burst transmission mechanism for streamingbetween the surrogate and the device. Using the HTTP livestreaming protocol [14], the mobile device sends out requestsfor the next segment of the video stream from time to time. Thesurrogate divides the video into segments, and sends each segmentin a burst transmission to the mobile device, upon sucha request. When the mobile device is receiving a segment, itoperates in the high-power state (CELL_DCH); when there is

nothing to receive, it transfers to the low-power state (IDLE) viathe intermediate state (CELL_FACH), and remains there untilthe next burst (segment) arrives.

*3) Burst Size:*To decide the burst size, *i.e.*, the size of thesegment transmitted in one burst, we need to take into considerationcharacteristics of mobile streaming and energy consumptionduring state transitions. For video streaming using a fixeddevice without power concerns, it is desirable to download asmuch of a video as what the connection bandwidth allows; however,for streaming over a cellular network, we should avoiddownloading more than what is being watched for one mainreason: usersmay switch among channels from time to time andthose prefetched contents are probably never watched, leadingto a waste of the battery power and the cellular data fee due totheir download. Hence, the bursty size should be kept small,to minimize battery consumption and traffic charges. On theother hand, state transitions introduce latency and energy overheads,so the burst should be large enough to avoid frequentstate transitions; otherwise, such overheads may diminish theenergy saving achieved by an intelligent state transition mechanism. We next derive a lower bound on the burst size, which guarantees positive energy saving by such intelligent state transition.

Let B be the average available bandwidth over a wireless connection, S be the burst size in the CELL_DCH state.P_{DCH}, P_{FACH} and P_{IDLE} denote the power levels at states CELL_DCH, CELL_FACH, and IDLE, respectively $t_{DCH \to FACH}$ is the timeout of the critical inactivity timer (the of the inactivity timer for state transition from CELL_FACH to IDLE.Let$P_{IDLE \to FACH}$, $P_{FACH \to DCH}$ and $P_{IDLE \to DCH}$ be the energy needed for state promotion denoted by the respective subscript. We ignore the delay overhead in the state promotion from a low-power state to a high-power state. An illustration of power consumption in both cases is given in Fig.3.1. The burst transmission operates at state CELL_DCH to send a total amount of data for a duration of; then it transits to state IDLE via state CELL_FACH, and remains there for duration (is the time taken for the mobile user to play the segment of size).

Note that the power consumption level during transition periods $t_{DCH \to FACH}$ and $t_{FACH \to IDLE}$, remains at P_{DCH} and P_{FACH} , respectively, although no data is transmitted then. The continuous transmission always operates at the high-power state CELL_DCH with power level $P_{cont}(t) = P_{DCH}$ .We calculate the overall energy saving (ΔE) by burst transmission of the video over the time span (multiples of s/b), as compared to the continuous transmission, as follows:

$$
\begin{aligned}
\Delta E &= \int_0^T (P_{cont}(t) - P_{burst}(t)) dt \\
&= \int_0^{S/b} (P_{cont}(t) - P_{burst}(t)) dt \times \frac{T}{\frac{S}{b}} \\
&= \frac{T \times b}{S} \int_0^{S/b - S/B} (P_{cont}(t) - P_{burst}(t)) dt \\
&= \frac{T \times b}{S} \times \Big((P_{DCH} - P_{FACH}) \times t_{FACH \to IDLE} \\
&\quad + (P_{DCH} - P_{IDLE}) \times \Big(\frac{S}{b} - \frac{S}{B} - t_{FACH \to IDLE} \\
&\quad - t_{DCH \to FACH} \Big) - P_{IDLE \to FACH} - P_{FACH \to DCH} \Big)
\end{aligned} \quad (1)
$$

The burst size should be chosen such that positive energy saving, can be achieved. A lower bound of the burst size can be decided using$\Delta E > 0$. We also see that the larger S is, the more energy saving we can achieve using burst transmission. However, with a large S, a user has to wait for a long time before the first segment is transcoded and delivered, and there is also the risk that it may download contents it will never watch.
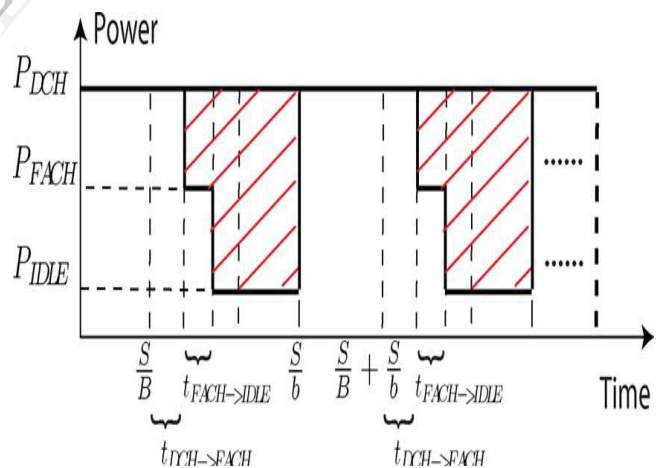

Fig. 3.1 Power consumption over time.

4. CLOUDMOV: PROTOTYPE IMPLEMENTATION

Following the design guidelines in Section III, we have implementeda real-world mobile social TV system, and deployedit on the Google App Engine (GAE) and Amazon EC2 clouds,which are the two most widely used public PaaS and IaaS cloudplatforms.

A. Client Use of CloudMoV

All mobile devices installed with HTML5 compatible browsers can use *CloudMoV* services, as long as the HTTPLive Streaming (HLS) [14] protocol is supported. The user firstconnects to the login page of *CloudMoV*, as

illustrated in the topleft corner of Fig. 3. After the user successfully logs in throughthe 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, are maintained in an in-memory tableof a MySQL database deployed in the gateway). Then theuser is automatically redirected to the assigned VM

surrogate,and welcomed by a portal page as shown on the right-handside of Fig. 4. Upon user login, the portal collects the deviceconfiguration information by examining the "User-Agent"header values, and this information will be sent to its surrogatefor decision making of the video encoding formats. The usercan enter the URL of the video or live broadcast he wishes towatch, on the "Subscribe" tab of the portal; after he clicks the"Subscribe" button, the address of the video is sent to the VMsurrogate, which downloads the stream on the user's behalf,transcodes and sends properly encoded segments to the user.
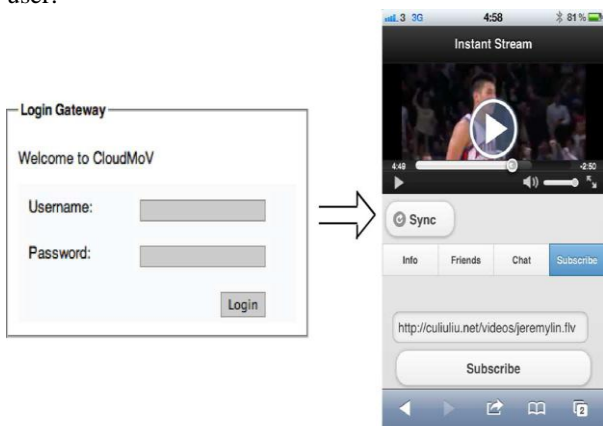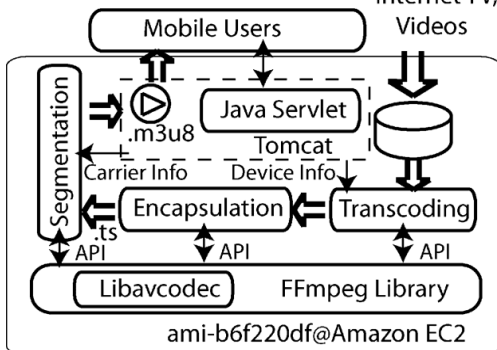

Fig. 4. Client UI of *CloudMoV*

Fig. 4.1. Streaming architecture in each customized VM image (ami-b6f220df).
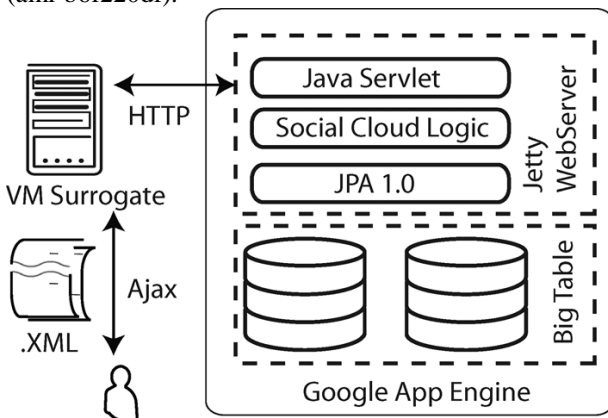

Fig. 4.2. Social message exchanges via Google App Engine.

B. VM Surrogates

All the VM surrogates are provisioned from Amazon EC2web services and tracked by the gateway. We create ourown AMI (ami-b6f220df) based on Linux kernel 2.6.35.14,the default image Amazon provides. Due to the intensivecomputation involved, we propose to implement all thevideo processing related tasks using ANSI C, to guaranteethe performance. In particular, we install FFmpeg togetherwith libavcodec as the groundsill library to develop thetranscoding, segmentation and reshaping modules on the VMsurrogates.We have also installed a Tomcat web server (version6.5) to serve as a Servlet container and a file server on eachsurrogate. Both FFmpeg and Tomcat are open source projects.Once a VM surrogate receives a video subscription requestfrom the user, it downloads the video from the source URL, andprocesses the video stream by transcoding and segmentation,based on the collected device configurations by the portal.For example, in our experiments, the downloaded stream istranscoded into a high-quality stream and a low-quality streamin real time with H264/AAC codecs. The high-quality streamhas a "480× 272" resolution with 24 frames per second,while the low-quality one has a "240 ×136" resolution with10 frames per second. A mobile user dynamically requestssegments of these two different video streams, according to hiscurrent network connection speed. The transcoded stream isfurther exported to an MPEG-2 transporting stream (.ts), whichis segmented for burst transmission to the user. The burst sizesdepend on both the network bandwidth and video bit rate.

C. Data Models in the Social Cloud

We use GAE mainly as the back-end data store to keep thetransient states and data of *CloudMoV*, 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 toGAE, under limited usage constraints, where no platform-specificAPIs are enforced for the deployment. GAE provides bothits Java Persistence API (JPA 1.0, part of JSR 220) adapter and aset of proprietary low-level APIs to map the relational data.Wechoose to use the former only in *CloudMoV* such that *CloudMoV*can be easily migrated to other PaaS clouds as well.

Once a user logs in to the system and enters the URL ofa video to watch, a session ID is generated for the new session(corresponding to viewing of this video), by combining theuser's "username" in the system with the time stamp when the

session is created. The gateway delivers an HTTP request to a Servlet listener running on GAE, to notify it that an entry for thenewly joined user should be added, with the user's "username"as the key and other information (URL of the subscribed video,the session ID, etc.) as the value.

5. CONCLUDING REMARKS AND FUTURE WORK

This paper presents our view of what might become a trendfor mobile TV, *i.e.*, mobile social TV based on agile

resourcesupports and rich functionalities of cloud computing services.We introduce a generic and portable mobile social TV framework,*CloudMoV* that makes use of both an IaaS cloud and aPaaS cloud. The framework provides efficient transcoding servicesfor most platforms under various network conditions andsupports for co-viewing experiences through timely chat exchangesamong the viewing users. By employing one surrogateVM for each mobile user, we achieve ultimate scalability of thesystem. Through an in-depth investigation of the power statesin commercial 3G cellular networks, we then propose an energy-efficient burst transmission mechanism that can effectivelyincrease the battery lifetime of user devices.We have implemented a realistic prototype of *CloudMoV*, deployedon Amazon EC2 and Google App Engine, where EC2instances serve as the mobile users' surrogates and GAE asthe social cloud to handle the large volumes of social messageexchanges. We conducted carefully designed experimentson iPhone 4S platforms. The experimental results prove thesuperior performance of *CloudMoV*, in terms of transcodingefficiency, power saving, timely social interaction, and scalability.The experiments also highlight the drawbacks of the currentHTTP Live streaming protocol implementation on mobiledevices [14] as compared to our proposed burst transmissionmechanism which achieves a 29.1%increase of battery lifetime.Much more, however, can be done to enhance *CloudMoV* tohave product-level performance. In the current prototype, we donot enable sharing of encoded streams (in the same format/bitrate) among surrogates of different users. In our future work,such sharing can be enabled and carried out in a peer-to-peerfashion, *e.g.*, the surrogate of a newly joined user may fetch thetranscoded streams directly from other surrogates, if they areencoded in the format/bit rate that the new user wants.For implementing social interactions, most BigTable-likedata stores (including GAE) support memcache which is ahighly efficient secondary storage on the data stores. We seekto integrate memcache support into *CloudMoV*, by possiblymemcaching the data (*e.g.*, chat histories) of sessions wherefriends chat actively, so as to further improve the query performance.To sustain the portability of the system, we will stick tostandard API interfaces, *i.e.*, JCache (JSR 107), in our system.

REFERENCES

[1] M. Satyanarayanan, P. Bahl, R. Caceres, and N. Davies, "The case forVM-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] Z. Huang, C. Mei, L. E. Li, and T. Woo, "Cloud stream: Delivering high-quality streaming videos through a cloud-based SVC proxy," in *Proc. INFOCOM'11*, 2011, pp. 201–205.

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

[5] 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.

[6] J. Santos, D. Gomes, S. Sargento, R. L. Aguiar, N. Baker, M. Zafar,and A. Ikram, "Multicast/broadcast network convergence in next generation mobile networks," *Comput. Netw,* vol. 52, pp. 228–247, Jan. 2008.

[7] DVB-H. [Online]. Available: http://www.dvb-h.org/.

[8] K. Chorianopoulos and G. Lekakos, "Introduction to social TV: Enhancing the shared experience with interactive TV," *Int. J. Human-Comput. Interaction*, vol. 24, no. 2, pp. 113–120, 2008.

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

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

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

[12]Live stream. [Online]. Available:http://itunes.apple.com/us/app/live stream/id379623629?

[13] NoSQL Date Base. [Online]. Available: http://nosql-database.org/.

[14] HTTP Live Streaming. [Online]. Available: http://tools.ietf.org/html/Draft-pantos-http-live-streaming-01.

[15] 3GPP TS 25.331. [Online]. Available:http://www.3gpp.org/ftp/Specs/html-info/25331.htm.