International Journal For Technological Research In Engineering
Volume 2, Issue 6, February-2015 *ISSN (Online): 2347 - 4718*

ENHANCING SOCIAL INTERACTION IN MOBILE USING CLOUD SERVICES

K. Kavitha[1], D. Geetha[2]
[1]Associate Professor, [2]PG Scholar
Department of IT, Annai Mathammal Sheela Engineering College, Namakkal-637013

Abstract: The rapidly increasing power of personal mobile devices (smart phones, tablets, etc.) is providing much richer contents and social interactions to users on the move. The recent cloud computing technology, with its rich resources to compensate for the limitations of mobile devices and connections, can potentially provide an ideal platform to support the desired mobile services. Tough challenges arise on how to effectively exploit cloud resources to facilitate mobile services, especially those with stringent interaction delay requirements. This system is based on the design of a Cloud-based, novel Mobile sOcial tV 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. Given the battery life as a key performance bottleneck, the use of burst transmission from the surrogates to the mobile users is enabled, and carefully decides the burst size which can lead to high energy efficiency and streaming quality. Social interactions among the users, in terms of spontaneous textual exchanges, are effectively achieved by efficient designs of data storage with BigTable and dynamic handling of large volumes of concurrent messages in a typical PaaS cloud.

I. INTRODUCTION

Cloud Computing is the delivery of computing resources over the internet. Cloud services allows businesses and individuals to use hardware and software that managed by the third parties at remote locations. A mobile user exploits VM technology to rapidly instantiate customized service software on a nearby cloudlet. Think air is the frame work for migrating Smartphone applications to the cloud. Although many media and mobile social applications have emerged, truly destroyer ones gaining mass acceptance are still impeded by the limitations of the present wire technologies and mobile devices, among which unstable wireless connection and battery life time are challenging one. In now a day's every Smartphone users need the fastest technologies like 3G, Wi-Fi for fast web access & chatting. These technologies focus more on the challenging scenarios such as real-time video streaming and online gaming, for social apps, and emails. The Cloud- MoV utilizes agile resource support and the functionalities which are provided by both an Infrastructure-as-a-Service (IaaS) cloud and a Platform-as-a-Service (PaaS) cloud. It therefore blends social awareness and co-viewing experience among friends on the go. As opposed to traditional Television watching, mobile social Television is well suited to today's life style, where family and friends may be separated geographically but hope to share a co-viewing experience. While social Television Social TV is any technology that enables social interaction in the context of watching TV content. Nowadays it is not always possible to meet to watch TV together, and that is where the idea of synchronous Social TV comes in, trying to simulate a collocated watching experience while in different locations. A number of mobile TV systems have 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 television network and the mobile network, than exploring the demand of "social" interactions among mobile users.

II. PROPOSED SYSTEM

This system proposes the design of a Cloud-based, novel Mobile social TV system. 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, system employ a surrogate for each user in the IaaS cloud for video downloading and social exchanges on behalf of the user. GAE, as a PaaS cloud, provides rich services on top of Google's data centres and enables rapid deployment of Java based and Python-based applications. Data store, a thin layer built on top of Google's famous Big Table handles "big" data queries well with linear and modular scalability even for high throughput usage scenarios. 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. which may hinder both computation-intensive jobs and content distribution applications. Amazon EC2 is a representative IaaS cloud, offering raw hardware resources including CPU, storage, and networks to users. A surrogate (i.e., a virtual machine (VM) instance), or a VM surrogate equivalently, is created for each online mobile user in an IaaS 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. Besides, they are also responsible for handling frequently exchanged social messages among their corresponding users in a timely and efficient manner, shielding mobile devices from unnecessary traffic and enabling battery efficient, spontaneous social interactions. 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 CloudMoV that keeps track of participating users and their VM surrogates, which can be implemented by a standalone server or VMs in the IaaS cloud.

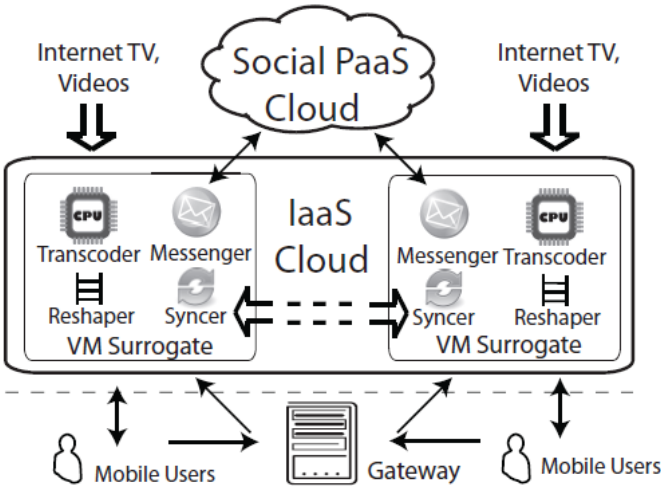


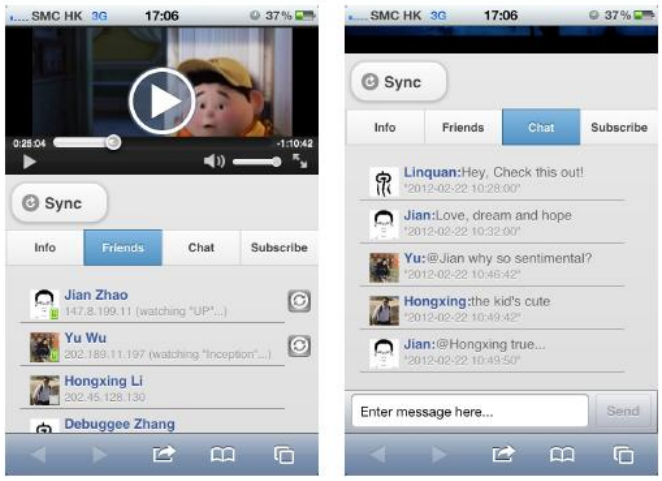
Fig. 1.The Architecture of CloudMoV

A. Client Use of CloudMov

All mobile devices installed with HTML5 compatible browsers can use CloudMoV services, as long as the HTTP Live Streaming (HLS) protocol is supported. The user first connects to the login page of CloudMoV, as illustrated in the top left corner of Fig. 2. 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, are 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. 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. From the surrogate to the mobile device, the video stream delivered using HLS is always divided into multiple segments, with a playlist file (.m3u8) giving the indices. When the mobile client subscribes to a video, the playlist is first downloaded and individual segments are requested by the client.

Fig. 2. Client UI of CloudMov

When watching a video, the user can check out his friends' status (online or offline, which video they are currently watching) on the "Friends" tab (a snapshot is given in Fig. 3(a)), and invite one or more friends to join him in watching the video. When a user receives an invitation from a friend (profile pictures of those friends who have sent invitations will be highlighted on the "Friends" tab) and decides to join the session, he can choose to watch from the start, or catch up with the viewing progresses of others by clicking the "Sync" button, which triggers the Syncer functionality in the surrogate. Users in the same session can exchange opinions and comments on the "Chat" tab (a snapshot is given in Fig. 3(b)), where new chat messages can be entered and the chat history of the session is shown. The "Info" tab shows an abstract of the video, as edited by the session host.



(a) "Friend tab" (b) "Chat tab"

Fig. 3. UI of "Friend" and "Chat" tabs.

B. VM Surrogates

All the VM surrogates are provisioned from Amazon EC2 web services and tracked by the gateway. Due to the intensive computation involved, this system propose to implement all the video processing related tasks using ANSI C, to guarantee the performance. In particular, system

installs FFmpeg together with libavcodec as the groundsill library to develop the transcoding, segmentation and reshaping modules on the VM surrogates. 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. For example, in our experiments, the downloaded stream is transcoded into a high-quality stream and a low-quality stream in real time with H264/AAC codecs. The high-quality stream has a "480x272" resolution with 24 frames per second, while the low-quality one has a "240x136" resolution with 10 frames per second. A mobile user dynamically requests segments of these two different video streams, according to his current network connection speed. The transcoded stream is further exported to an MPEG-2 transporting stream (.ts), which is segmented for burst transmission to the user. The burst sizes depend on both the network bandwidth and video bit rate. The evaluation of impact of different burst sizes on the streaming quality and energy consumption is done here. Fig. 4 shows the streaming architecture in our customized VM image. Here, the modules on social message exchanges are omitted.

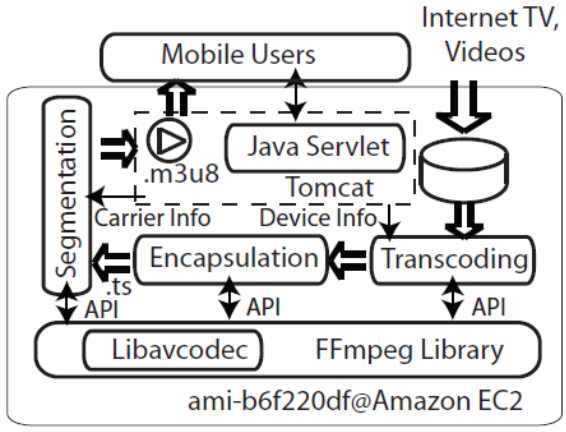


Fig. 4. Streaming architecture in each customized VM image

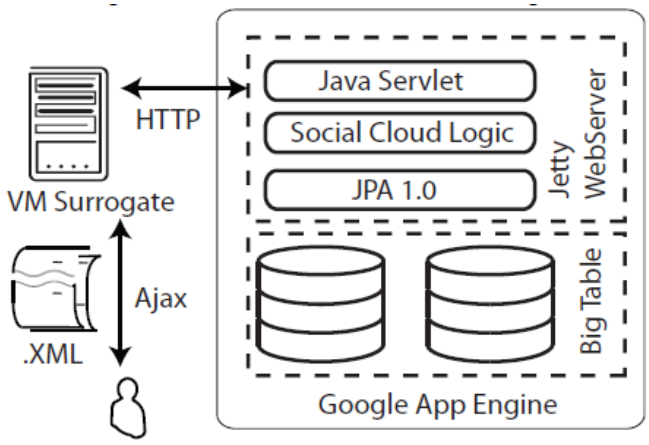


Fig. 5. Social message exchanges via Google App Engine.

III. SYSTEM DESIGN

A. Transcoder

It resides in each surrogate, and is responsible for dynamically deciding how to encode the video stream from the video source in the appropriate format, dimension, and bit rate. Before delivery to the user, the video stream is further encapsulated into a proper transport stream. 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.

B. Social Cloud

Social network is a dynamic virtual organization with inherent trust relationships between friends. This dynamic virtual organization can be created since these social networks reflect real world relationships. It allows users to interact, form connections and share information with one another. This trust can be used as a foundation for information, hardware and services sharing in a Social Cloud.

C. Messenger

It is the client side of the social cloud, residing in each surrogate in the IaaS 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 are asynchronously delivered from the surrogate to the user in a traffic-friendly manner, i.e., little traffic is incurred. In the reverse direction, the messenger disseminates this user's messages (invitations and chat messages) to other users via the data store of the social cloud.

D. Gateway

The gateway provides authentication services for users to log in to the CloudMoV system, and stores users' credentials in a permanent table of a MySQL database it has installed. It also stores information of the pool of currently available VMs in the IaaS cloud in another in-memory table. After a user successfully logs in to the system, a VM surrogate will be assigned from the pool to the user. The in-memory table is used to guarantee small query latencies, since the VM pool is updated frequently as the gateway reserves and destroys VM instances according to the current workload. In addition, the gateway also stores each user's friend list in a plain text file (in XML formats), which is immediately uploaded to the surrogate after it is assigned to the user.

E. Subscribe

In this module user can download the video. Subscribe module download video in high speed and clear video streaming. Authorized user every one download and watch the videos.

F. Transcoding Mechanism

It resides in each surrogate, and is responsible for dynamically deciding how to encode the video stream from

the video source in the appropriate format, dimension, and bit rate. Before delivery to the user, the video stream is further encapsulated into a proper transport stream. 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.

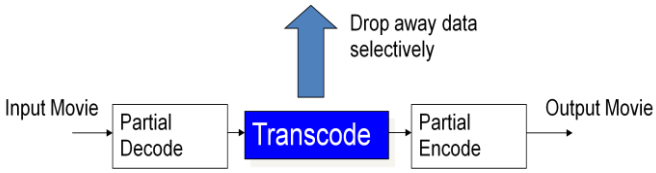


Fig. 6. Control flow of Transcoding mechanism

IV. SYSTEM SPECIFICATION

A. Android

Android is an operating system based on Linux with a Java programming interface. The Android Software Development Kit (Android SDK) provides all necessary tools to develop Android applications. This includes a compiler, debugger and a device emulator, as well as its own virtual machine to run Android programs. Android is currently primarily developed by Google. Android allows background processing, provides a rich user interface library, supports 2-D and 3-D graphics using the OpenGL libraries, access to the file system and provides an embedded SQLite database. Android applications consist of different components and can re-use components of other applications. This leads to the concept of a task in Android; an application can re-use other Android components to archive a task. For example you can trigger from your application another application which has it registered with the Android system to handle photos. In this other application you select a photo and return to your application to use the selected photo.

B. Android Goals Security and Permissions

During deployment on an Android device, the Android system will create a unique user and group ID for every Android application. Each application file is private to this generated user, e.g. other applications cannot access these files. In addition each Android application will be started in its own process. Therefore by means of the underlying Linux operating system, every Android application is isolated from other running applications. If data should be shared, the application must do this explicitly, e.g. via a service or a Content Provider. Android also contains a permission system. Android predefines permissions for certain tasks but every application can define additional permissions. An Android application declares its required permissions in its AndroidManifest.xml configuration file. For example an application may declare that it requires access to the Internet. Permissions have different levels. Some permission is automatically granted by the Android system, some are automatically rejected. In most cases the requested permissions will be presented to the user before installation of the application. The user needs to decide if these permissions are given to the application. If the user denies permission required by the application, this application cannot be installed. The check of the permission is only performed during installation; permissions cannot be denied or granted after the installation. Not all users pay attention to the required permissions during installation. But some users do and they write negative reviews on Google Play.

C. Android Applications and Tasks

An Android application consists out of different Android components and additional resources. The Android system knows activities, services, broadcast receiver and content provider as components. Android application components can connect to components of other Android applications to create tasks. For example an application which allows you to make a photo can start an email application and instruct this application to create a new email and attach a photo to this email. The following description gives a overview of the most important user interface related component and parts of an Android application. Activity An activity represents the visual representation of an Android application. Activities use views, i.e. user interface widgets as for example buttons and fragments to create the user interface and to interact with the user. An Android application can have several activities. Fragments Fragments are components which run in the context of an Activity. A Fragment encapsulates application code so that it is easier to reuse it and to support different sized devices. Fragments are optional components which allow you to reuse user interface and non user interface components for different devices configurations. Views and layout manager Views are user interface widgets, e.g. buttons or text fields. The base class for all views is the android.view. View class. Views have attributes which can be used to configure their appearance and behavior. A layout manager is responsible for arranging other views. The base class for these layout managers is the android. view. View Group class which extends the View class. Layout managers can be nestled to create complex layouts. You should avoid nestling them to deeply too deeply as this has a negative impact on the performance. The user interface for Activities is typically defined via XML files (layout files). It is possible to define defined layout file for different device configuration, e.g. based on the available width of the actual device running the application. Fragments are designed to support such a setup. The Android Software Development Kit (SDK) contains the necessary tools to create, compile and package Android application. Most of these tools are command line based. The Android SDK also provides an Android device emulator, so that Android applications can be tested without a real Android phone. You can create Android virtual devices (AVD) via the Android SDK, which run in this emulator. The Android SDK contains the Android debug bridge (adb) tool which allows connecting to a virtual or real android device.

V. CONCLUSION AND FUTURE ENHANCEMENT

This system presents 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. This system introduces a generic and portable

mobile social TV framework, CloudMoV that makes use of both an IaaS 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, here ultimate scalability of the system is achieved. Through an in-depth investigation of the power states in commercial 3G cellular networks, an energy-efficient burst transmission mechanism that can effectively increases the battery lifetime of user devices is proposed here. This system implemented a realistic prototype of CloudMoV, 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. Carefully designed experiments are conducted on iPhone 4S platforms. The experimental results prove the superior performance of CloudMoV, in terms of transcoding efficiency, power saving, timely social interaction, and scalability. The experiments also highlight the drawbacks of the current HTTP Live streaming protocol implementation on mobile devices as compared to our proposed burst transmission mechanism which achieves a 29.1 % increase of battery lifetime. In the current prototype, sharing of encoded streams (in the same format/bit rate) among surrogates of different users is not enabled. 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.

REFERENCES

[1] Carroll .A and Heiser .G (2010), "An analysis of power consumption in as smartphone," in Proc. of USENIXATC.
[2] Coppens .T, Trappeniners .L, and Godon .M (2004), "AmigoTV: towards a social TV experience," in Proc. of EuroITV.
[3] Ducheneaut .N, Moore .R .J, Oehlberg .L, Thornton .J .D, and Nickell .E (2008), "Social TV: Designing for Distributed, Sociable Television Viewing," International Journal of Human-Computer Interaction, vol. 24, no. 2, pp. 136–154.
[4] Huang .Z, Mei .C, Li .L. E, and Woo .T, "Cloudstream: Delivering high-quality streaming videos through a cloud-based svc proxy," in INFOCOM'11, 2011, pp. 201–205.
[5] Flinn .J and Satyanarayanan .M (1999), "Energy-aware adaptation for mobile applications," in Proceedings of the seventeenth ACM symposium on Operating systems principles, ser. SOSP '99, pp. 48–63.
[6] Santos .J, Gomes .D, Sargento .S, Aguiar .R. L, Baker .N, Zafar .M, and Ikram .A (Jan 2008), "Multicast/broadcast network convergence in next generation mobile networks," Comput. Netw., vol. 52, pp. 228–247.
[7] Schatz .R and Egger .S (2008), "Social Interaction Features for Mobile TV Services," in Proc. of 2008 IEEE International Symposium on Broadband Multimedia Systems and Broadcasting.
[8] Schatz .R, Wagner .S, Egger .S, and Jordan .N (2007), "Mobile TV becomes Social - Integrating Content with Communications," in Proc. of ITI.
[9] Zhu .W, Luo .C, Wang .J, and Li .S(2011), "Multimedia cloud computing," IEEE Signal Processing Magazine, vol. 28, pp. 59–69.

