Journal of Data Mining and Knowledge Discovery
ISSN: 2229–6662 & ISSN: 2229–6670, Volume 3, Issue 1, 2012, pp.-36-39.
Available online at http://www.bioinfo.in/contents.php?id=42

ANDROID BASED SERVER FOR SHARING BACKUP AND RESTORING DATA

PAWADE P.P.[1] AND KATHALKAR A.A.[2]

[1]Sipna's College of Engineering & Technology, Amravati, MS, India.
[2]Sinhgad College of Engineering, Pune, MS, India.
*Corresponding Author: Email- pparnal@gmail.com, aniket.kathalkar@gmail.com

Received: February 21, 2012; Accepted: March 15, 2012

Abstract- The number of smartphone users and mobile applications are growing rapidly. Though smartphones are expected to have PC-like functionality, hardware resources such as CPU's, memory and batteries are still limited. To solve this resource problem, many researchers have proposed architectures to use server resources in the cloud for mobile devices. While many solutions for making backups and restoring data are known for servers and desktops, mobile devices pose several challenges, mainly due to the plethora of devices, vendors, operating systems and versions available in the mobile market. In this paper, we propose a conceptual architecture of Android as a Server Platform, which enables multiple user Android applications on cloud server via network and backup and restore approach for mobile devices, which helps to reduce the effort in saving and restoring personal data. Android's two other features are useful to construct a server platform, i.e. Android is open source product and runs on an x86 CPU. Another feature of our approach lies in the capability of sharing information in mobile devices among a group of selected persons. This can be useful in many situations e.g., in creating a mobile business network among a group of people.
Keywords- Android, multi-tenant; cloud, backup and restore, mobile devices.

Citation: Pawade P.P. and Kathalkar A.A. (2012) Android Based Server for Sharing Backup and Restoring Data. Journal of Data Mining and Knowledge Discovery, ISSN: 2229–6662 & ISSN: 2229–6670, Volume 3, Issue 1, pp.-36-39.

Introduction

The number of smartphones users and mobile applications are growing rapidly. There are several mobile Operating Systems, such as Symbian, iOS, Android, and Windows Mobile. Though smartphones are expected to PC-like functionality, hardware resources such as CPU's, memory, and batteries are still limited. To solve this resource problem, some researchers have proposed using server resources in the cloud for smartphones. From this background, Android as a Server Platform is proposed that enables many users to use resources on remote cloud servers. Android is an open source mobile OS initiated by Google. The main reason to use Android as a server platform is that it is able to run not only for smartphones but also for the x86 platform including servers.

Backup is a crucial task, since hardware faults and software or human errors can lead to the loss of important information. In addition to faults, backups are even more important for devices such as laptops and smartphones, since they are more prone to loss or to theft. Currently, smartphones are used more as handheld computers than as mobile phones, and consequently a lot of data is stored in those devices. This makes more critical the need to keep data stored on those devices safe from losses. In addition, the rapid technological evolution in mobile devices makes it more difficult to restore data saved from old devices to new ones. Thus, mobile devices pose new challenges for the backup and restore problem.

Making backups on external memory devices, such as on Secure Digital (SD) cards or on laptop disks, suffers from the same risks of failure or loss. As smartphones tend to be always connected to the Internet, it seems natural to move the information online and to provide backup and restore services based on the cloud computing paradigm, which is considered to be more reliable and less expensive by end users [1, 2]. This approach reduces also the risk of data loss and decouples the data from a specific device.

Once information about backups moves online, it can be used in shared applications. In an enterprise scenario, for example, it can be useful for users to share business or personal data contained in their mobile's backups, such as calendar or business cards, with some selected contacts of their choice. In such scenario, it is easy to imagine a community of people willing to share some of their data within their mobile network. A backup that allows data sharing, however, can suffer the same security and privacy issues present in social networks [3]; such limitations can be approached in different ways depending on the environment where the system is used. In an enterprise scenario, data sharing can be monitored by administrators which can enforce the company privacy policies. In a general purpose environment, like a mobile social network, ownership of data must be verified and sharing must be allowed only for the data owner. Security can be ensured by deploying secure connections and data encryption. The main goal of this work is a backup system for smartphones that allows users to share part of their personal data in the backup with a selected set of contacts. In order to be platform independent, this approach is based on a novel management of data, and hinges on a data model which abstracts from the underlying platform and focuses on the data type. The same backup and restore method can be applied both on mobile and on desktop platforms. With such system, users can manage different devices, under different operating systems, and keep data synchronized across different platforms.

Materials and Methods

Mobile Application Platform on Cloud Server

As a number of service providers such as Dropbox [4] and Zumodrive [5] provide online storage services, the architecture for remotely using mobile application on server has many benefits for users. This approach, called Mobile Application Platform on Cloud Server, intends to handle not only user data but also user applications in a cloud server. This approach changes the application lifecycle as follows. "Write once, run everywhere. Install once, use everywhere." By executing a mobile application in the cloud server, users and developers free from device limitation such as CPU power, memory and battery and from device software environment such as OS or version.

Multitenant for Android

Multi-tenancy, which means that software running on a server provides services to many users, is one of the important features for cloud computing. From the viewpoint of both economy and ecology, it is beneficial to share hardware resources among users. Using a mobile OS would be more effective than using a desktop OS because the resource requirements of mobile Oss are smaller.

Backup Methods

According to [6], backups can be classified in several types; it is possible to distinguish the data repository model in full backups vs. incremental backups, data can be stored in a file-based or a device-based style, and the data repository management can be classified as on-line vs. off-line; those approaches can be combined in different ways, according to accessibility, security and cost needs. On-line backups permit to save and restore data while the system is running, while off-line backups require the system to be idle. In all cases, backups can be stored locally, e.g., on an external device, or remotely, on a remote server.

Results and Discussions

Multi Tenant Architecture for Android

This section discusses to construct multi-tenant architecture for Android. Figure 1 shows an overview of the architecture Android on a server. Android system info is drawn upon Google's "Application Developers" document [7]. There are three types of approach, hypervisor-layer, kernel-layer, and framework-layer, for multi-tenant architecture.

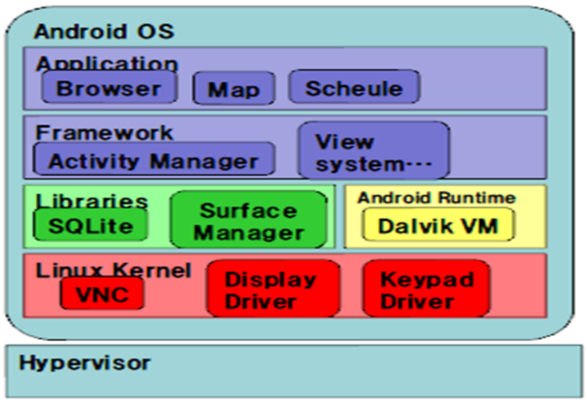

Fig. 1- Overview of Android on a Server

The hypervisor-layer approach uses the Virtual Smartphone over IP system. Each user owns his/her Android OS image on a server and freely runs his/her application in a separate VM Multi-tenancy is achieved by running multiple users VMs in a server via a hypervisor.

This approach has the advantage of application usability and maintenance. From the viewpoint of application usability, every mobile application that can run on Android-x86 is usable because each Android OS runs only one application. Android has different versions and version up is currently on going.

The second approach implements multi-tenant function in kernel-layer, as illustrated in Figure 2.This approach changes Android OS to run multiple user applications in separate processes. This approach is similar to an ordinary thin client server running multiple user applications in a server. The main challenge is that original Android supports only one display and keypad device since Android is mainly designed to work on a smartphone.

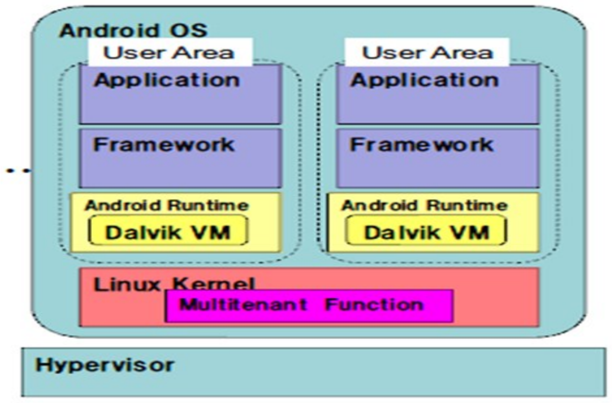

Fig. 2- Kernel-layer approach

Another approach is to create a multi-tenant function at framework -layer, similar to existing a Java-based multi-tenant framework. This approach remodels Android the framework and APIs to support multiple user applications. The main challenge is how to run existing Android applications in modified framework.

As shown in [8], the hypervisor-layer approach is feasible and good for maintenance. However, it seems to have a scalability limitation caused by a hypervisor. Because each VM try to separately maintain their resources, it is difficult to control unused resources. The other two approaches have an advantage in scalability but have a disadvantage in maintenance because they change the Android OS. From the viewpoint of running existing application, the kernel-layer approach is better because it does not changes Android runtime environment. Moreover, we assume that the kernel-layer approach is easy to develop because Android is implemented based on the Linux kernel so that can support multiple displays, keypads, and applications.

A New Approach for Mobile Backup/Restore

In this work, we try to overcome the limitations in saving and restoring data from mobile devices, by using online backups as a uniform interface for sharing data among different users and multiple platforms. In particular, an online backup system based on a Service Oriented Architecture (SOA) is presented: the services offered by this solution allow to backup and restore not only files but also more structured data such as contacts, calendar events, and text messages (SMS). In order to be able to access those services, a mobile device must be equipped with a client capable of retrieving internal data from the device and sending them to the server via a common interface. This interface is designed so as to exploit the common features of mobile data models: e.g., independently of the platform used, a contact in an address book is always identified by fields such as first name, last name, address, phone numbers, etc. All the communication exchanged between the client and the server is based on an extensible standard language (i.e. XML).

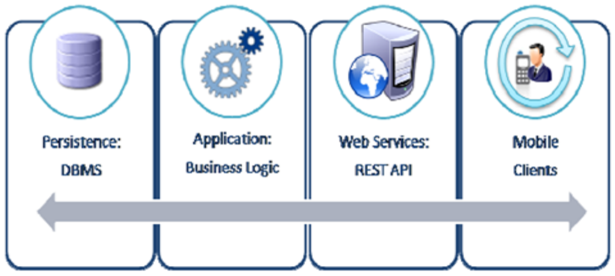

Fig. 3- Backup and Restore System Architecture

In this architecture (see Figure 3), the server provides his services using a representational state transfer (REST) architecture [11, 12]. For each type of platform, a different client is implemented: each client connects to the server using the HTTP protocol to exchange information in XML format. In the following, we describe in more detail the functionalities of the server and the client.

A. Server

The server has been designed as RESTful: in REST architectures requests and responses are built around the transfer of representations of resources. In this case, a resource is the XML representation of its state.

REST architectures are based on the HTTP protocol and use all the HTTP facilities, such as the security layer provided by HTTPS in a transparent way. The server allows mobile clients to perform full backups and incremental backups. When a user performs a backup, all the user's data previously stored on the server are still accessible from the mobile client; old data are kept on the server, and made accessible to the mobile client, to allow the user to revert to old backups in case of loss or failure.

B. Client

The client can be implemented for different types of devices (mobile, desktop, game console, Internet TV etc…). The software should be implemented to access private data residing on the device and to send such data on a remote server which will store these data. Clients must be able to handle HTTP messages bodies, get data sent by the server and store them into the device, for example in the address book, in the device specific format.

Usually devices need to be built on purpose to interact with a backup server; in some cases they need to handle dirty flags in order to manage the status of the resources to be saved. In this approach, in order to interact with the server, clients need only to be able to read and write resources to be saved and to implement just some basic HTTP methods.

Use Cases

Social backup in business environment

In an enterprise where people collaborate daily, it could be important for employees to share commonly useful information e.g., calendar, part of the address book, templates for presentations or documents etc.

Moreover if a new employee joins the team, his/her contacts are added to the common address book and shared with selected users of his/her new team; his/her new business device is added to a specific closed group and all data updated to the last changes are kept from the shared backup and saved on it. If somebody's device is lost, or stolen, or the employee leaves the company, the group administrator can disconnected it from the social backup and the privacy of the group members is granted. Using this approach, all these updates are directly exchanged and notified on employees' smartphones.

Conclusion

In this paper, we proposed Android as a server platform system that enables the use of saving, recovering and sharing personal information into closed groups of smartphones. We also showed the technical difficulty and approaches related to multi-tenant architecture for Android OS. We plan to develop a prototype system about proposed multi-tenant Android architecture. We believe that proposed architecture shows high performance on virtual image-based virtualization for mobile applications.

Acknowledgements

We wish to thank many people who have extended their kind cooperation & guidance to make this paper successful. We sincerely acknowledge our indebtedness to our guides Dr. R.S. Kawitkar and Prof. V.T. Gaikwad, without their valuable guidance and encouragement this work would not have been possible. We also mention our sincere thanks to the Principal for giving all possible infrastructure and facilities to complete this work within the prescribed time. We can never forget our parents and our family members whose blessings & best wishes inspired us to present this in a flamboyantly way.

References

[1] Buyya R., Chee Shin Yeo and Venugopal S. (2008) *10th EE International Conference*, 5-13.
[2] Wei-Tek Tsai, Xin Sun and Balassooriya J. (2010) *Seventh International Conference*, 684-689.
[3] ENISA (2007) *Giles Hogben*.
[4] *http://www.dropbox.com/*.
[5] *http://www.zumodrive.com/*.
[6] Chervenak A., Vellanki V. and Kurmas Z. (1998) *Joint NASA and IEEE Mass Storage*.
[7] Android Developers. *http://developer.android.com/index.html*.
[8] Chen E.Y. and Ito M. (2010) *IEEE WOWMOM*.
[9] Agarwal S. and Starobinski D. and Trachtenberg A. (2002) *Department of Electrical and Computer Engineering, Boston University*.
[10] *http://www.openmobilealliance.org/tech/affiliates/syncml/syncmlindex.html*.
[11] Fielding R. (2000) *Architectural Styles and the Design of Network-based Software Architectures*.
[12] Fielding R.T. and Taylor R.N. (2000) *Software Engineering, International Conference*, 407-416.