Available Online at www.ijcsmc.com

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X

IJCSMC, Vol. 2, Issue. 5, May 2013, pg.129 – 133

RESEARCH ARTICLE

Multi-Operating Environment System

Aditya Thakare[1], Prachi Deshmukh[2], Shraddha Kawane[3], Yogesh Sarda[4]

[1]Computer Engineering Department, Sinhgad College of Engineering, University of Pune, India
[2]Computer Engineering Department, Sinhgad College of Engineering, University of Pune, India
[3]Computer Engineering Department, Sinhgad College of Engineering, University of Pune, India
[4]Computer Engineering Department, Sinhgad College of Engineering, University of Pune, India

[1] adityabthakare@gmail.com; [2] prachideshmukh05@gmail.com;
[3] shraddhakawane@gmail.com; [4] yogesh.sarda1@gmail.com

Abstract— Operating Systems are designed and optimized based on specific Environmental needs. The mobile operating systems are built to provide real-time applications and the desktop applications are built to provide extensive processing features. A multicore smartphone can be made available with the advantages/features associated with a mobile-specific embedded operating system and other general-purpose operating system without losing performance. The system is a middleware software system which helps user to change the operating environment. It consists of a multicore smartphone with two, co-existing, and independent environments (mobile/desktop) interacting with a common kernel. The system presents the desktop environment to the user when docked to a docking station or runs as a simple smartphone when undocked.

Key Terms: - Embedded Operating System; Real-time; Kernel Sharing; Multi-Operating Environment; Extensive Features; Middleware; Data Sharing; Resource sharing

I. INTRODUCTION

Now-a-days almost every person carries a mobile phone and if it is a smart phone it has considerable computational power. Smart phone provides real-time applications like calling, reading mails, playing music, and opening excel sheets, etc. But extensive feature like editing of an excel sheet on mobile phone is tedious job with a small screen and limited options for editing it. Instead, if you have laptop it would become easier and user friendly to use extensive features like editing an excel sheet. This creates the need of having both real time applications and extensive features on a single computational device like smart-phone.

A user of the computer system is always willing to have a suitable operating environment according to his needs. Users usually have mobile-specific embedded operating system on his mobile device and general-purpose operating system on his laptop. It would be more advantageous if both mobile and desktop operating environment get available on single computational device like a smart-phone. The idea is to provide a mobile device with the advantages/features associated with a mobile-specific embedded operating system and other general-purpose operating system without losing performance. The system is a middleware software system which helps user to change the operating environment according to user's need. The system consists of a mobile device with two, co-existing, and independent environments interacting with a common kernel, and related methods of operations.

II. PROBLEM DEFINITION

General-purpose computer operating systems have an extensive set of features such as file systems, device drives, applications, libraries, etc. Such operating systems allow concurrent execution of multiple programs, and attempt to optimize the response time and CPU usage.

Unfortunately, such operating systems are not generally suitable for providing the features of embedded operating system that resides on mobile device.

Herein generates the need of providing features/advantages associated with mobile specific real time embedded operating system with the General purpose desktop operating system on single computational device that is a smart-phone.

In order to achieve the goal of providing two environments user requires smart phone and docking station known as laptop dock. The Smartphone when docked into docking device a desktop application environment is served on the laptop dock and when undocked the user experiences a regular Smartphone application environment. Fig.1 show Mobile device connected to the Laptop Dock. The Application environments are independent of each other, share resources and data. Switching among application environments will be without rebooting the mobile device.

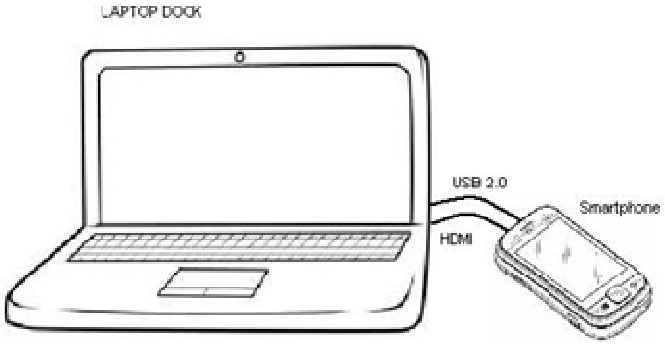

Fig.1 Mobile device connected to the Laptop Dock via HDMI and USB 2.0.

III. RELATED WORK

In order to achieve multiple operating environments, the first and foremost technique developed is the Virtualization Technique [3]. In virtualization method, stack structure is used. The second application environment is loaded with its own kernel over the first application environment which has its own kernel. When we are running an application in second application environment the application gives call to its kernel, the kernel again gives call to the kernel of first application environment which in turn calls the hardware. This function calling creates a heavy load on to the system and reduces the system performance. To achieve virtualization we have to build an application which:

- Gets the root permissions.
- Creates the loop device
- Mounts the desktop system image on that loop device
- Forwards the network to the desktop system
- Load desktop system in the RAM
- Start desktop system

After that to access the desktop system the VNC silent is needed to access the system at the specified port.

Another technique to achieve the multiple operating environments is the Motorola WebTop application [7]. In Motorola WebTop application the Mozilla Firefox is provided as the desktop application. The Motorola WebTop shares the address space which is used by the mobile application environment.

IV. KERNEL SHARING SYSTEM ARCHITECTURE

In kernel sharing technique both mobile and desktop environment must use common kernel. Both the environments reside in RAM at the same time.

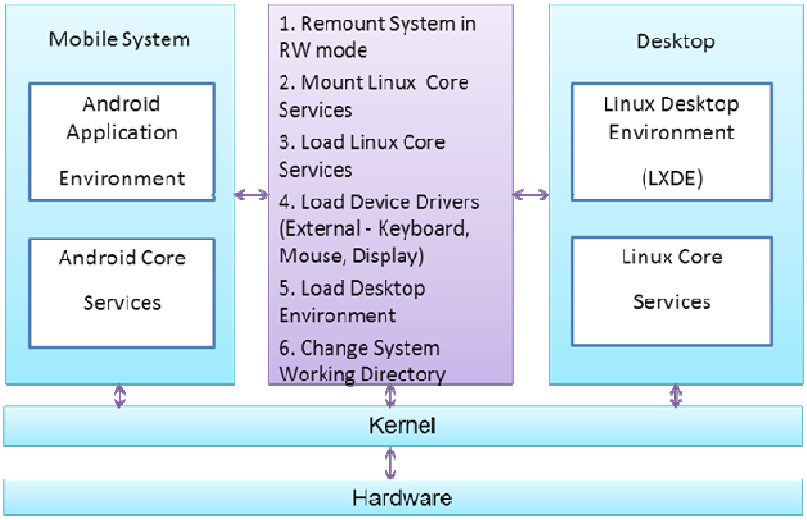

Fig.2 Kernel Sharing System Architecture

The system uses Linux kernel. When android Smartphone is started it first loads the Linux kernel then android core services and android applications are started. But the desktop environment must be loaded when smart-phone is docked by simultaneously loading Ubuntu core services and Ubuntu desktop environment. To load core services and desktop environment of Ubuntu the following steps need to be followed:

- Remount the system in RW mode: In normal user mode, change in system is not permitted. So system has to log in as root user and remount system in RW mode to make changes in the system.
- Mount Linux core services: Every operating environment requires different partition. So, system has to mount Linux core services.
- Load Linux core services: System has to load Linux core services in previously mounted partition.
- Load device drivers: The drivers of external keyboard, mouse and display have to be loaded in the system.
- Load desktop environment: System has to load Linux desktop environment above the Linux core services.
- Change system working directory.

The middle part of the system architecture acts as an interface for communication between mobile and desktop environment.

V. IMPLEMENTATION OF KERNEL SHARING SYSTEM

Normal user doesn't have permission to make change in the system. To get those permissions system user access is needed. For that purpose user has to install Super user and Busybox.

1 Super user: It gives root user access.
2 Busybox: It provides the set of commands to communicate with kernel.

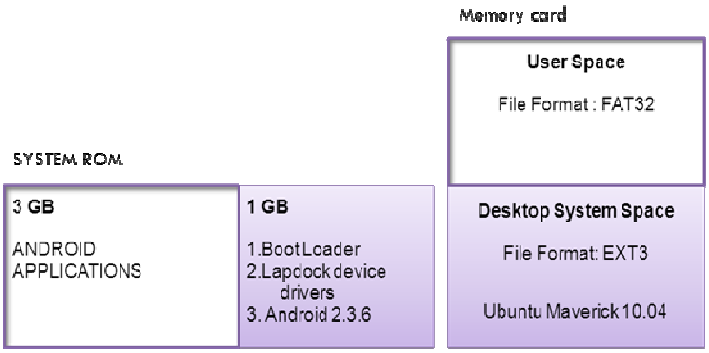

Fig.3 Memory Card Partition

In General smart-phone has 4GB of ROM in which 1GB is used for boot loader, device drivers and android image. Other 3 GB is used by android application. So there is no sufficient space to install ubuntu desktop system. Therefore memory card is used to extend system's ROM. As Linux understands EXT3 format, partitioning of memory card into FAT 32 and EXT3 format is done. FAT32 is used to store user data and EXT3 format is used to install ubuntu desktop system. Then installation of Ubuntu Desktop Image on the SD card partition at the /osh is done.

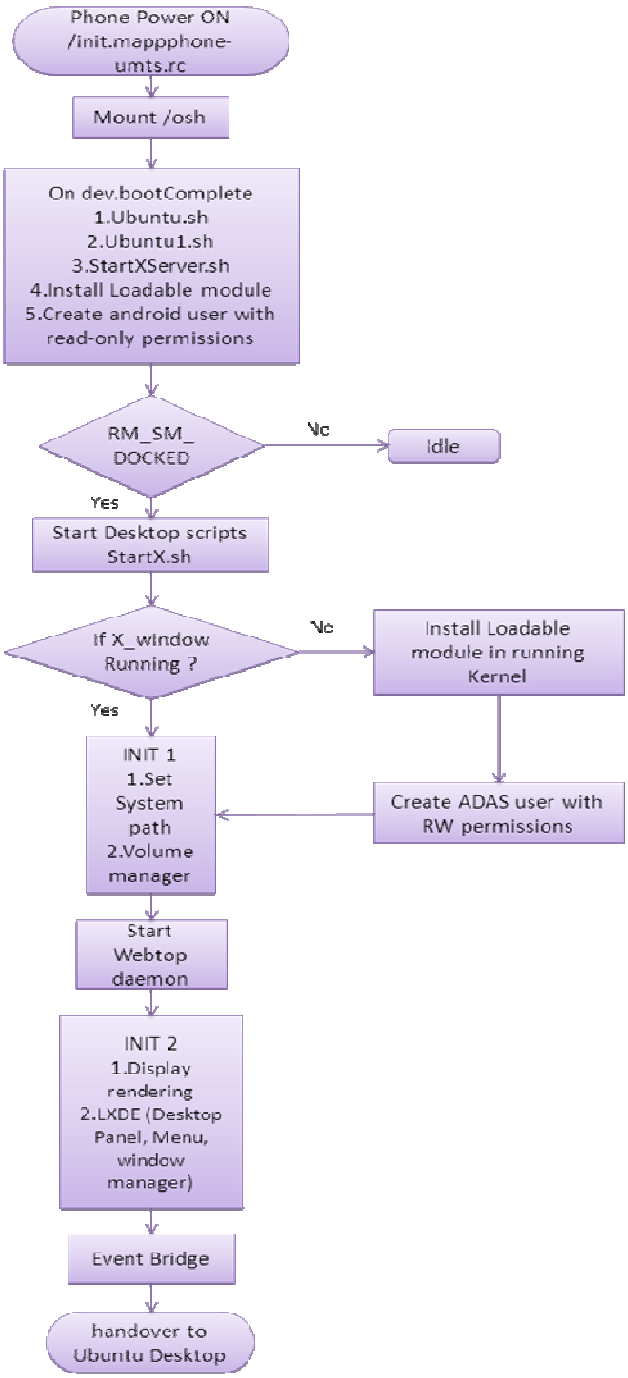

Fig.4 Overall System Flow

Fig.4 shows Overall System Flow. When smartphone is powered on /init.maphone-umts.rc script is run. Then mounting of EXT3 partition at /OSH is done. On dev.boot complete (system kernel loading variable is set),

ubuntu.sh script is executed which forks the ubuntu1.sh in the background. The ubuntu1.sh is responsible for installing the loadable modules in the running kernel, creating the android users read only permissions and forks the StartXServer.sh in background. The StartXServer.sh is responsible for loading the drivers, setting the system path and starting the android desktop environment. When the smartphone is docked into the docking device sys.SystemMode is set to RM_SM_DOCKED. This forks /etc/init.d/StartX.sh. StartX.sh is responsible for checking if the X_window system is running or not. X_window is a desktop window management unit. If the X_window is not running, then it installs the desktop loadable modules in running kernel. After that it creates the ADAS desktop user with read and write permissions. Then it forks the INIT1 script in the background. The INIT1 script sets the desktop system path, fork the volume manager and start Webtop daemon. Webtop daemon launches and mounts the phone file system in ubuntu and forks the INIT2 script. INIT2 script renders the display. It starts LXDE (Light Weight Desktop Environment) with its components like desktop panel, menu, window manager, session manager etc. Event Bridge is responsible for passing the keyboard, mouse events to the ubuntu system. After that we hand over to ubuntu desktop system.

VI. CONCLUSIONS

The paper represents the system which provides multiple operating environments that is desktop environment and mobile environment on single computational device, a Smart-phone, using kernel sharing technique. Both the operating environments have the common kernel. Hence the system provides the functionality of changing environment between mobile environment and desktop environment as per user's need with punctual performance.

The system also achieves data sharing and resource sharing as it uses the single computational device having both the operating environments residing in the device memory at the same time which ultimately removes the need of rebooting the system each time when operating environments needs to be changed.

ACKNOWLEDGEMENT

The authors' wishes express true sense of gratitude towards our project guide Prof. A.R.Joshi., Computer Engineering Department, Sinhgad College of Engineering. She contributed her valuable guidance and gave plenty of her precious time to solve every problem that arose.

REFERENCES

[1] Doug Abbott, Linux for embedded and real time application, 3rd ed., Newnes, 2012.
[2] Gene Sally, Pro Linux Embedded System, Apress, 2009.
[3] Zacthespack. "Linux on Android", [Online]. Available: http://linuxonandroid.org/what-is-linuxonandroid/
[4] Joshua D. Galicia, and Andrew N. Tzakis, "Multi-Environment Operating System", United States Patent Application Publication Galicia et al., 2011.
[5] Canonical Ltd. (2012) "Ubuntu for Android", [Online]. Available: http://www.ubuntu.com /devices/android.
[6] Kosmaz Technologies. "NEX PHONE". [Online]. Available: http://www.nexcrea.com/
[7] Motorola Inc. (2012) "Motorola Laptop Dock with Firefox Browser", [Online]. Available: http://www.motorola.com/Consumers/US-EN/Consumer-Product-and-Services/Atrix-Accessories-Page/Atrix-Accessories.
[8] Always Innovating Inc. (2012) "Always Innovation OS", [Online]. Available: http://www.alwaysinnovating.com/products/aios.htm.
[9] Always Innovating Inc. "HDMI Dongle",[Online]. Available: http://www.alwaysinnovating.com /products/hdmidongle.htm
[10] M3 Inc. "In Dash Android/Ubuntu Car PC Via HDMI", [Online]. Available: http://www.m3forum.net/m3forum/showthread.php?t=411567

