ISSN No. 0976-5697

Volume 6, No. 5, May - June 2015

International Journal of Advanced Research in Computer Science

REVIEW ARTICLE

Available Online at www.ijarcs.info

Comparative Study of Memory Management in Different Mobile Os: A Review

Rekha Sisodia
M.Tech Scholar
DCTM College, Palwal (India)

Dr.Kavita Sharma
Assistant Professor (Computer Science)
PT.J.L.N. Govt P.G College Faridabad (India)

Abstract: Mobile phone technology is growing day by day at a very high speed in telecommunication sector. A continuous growth becomes a motto of these industries. Every company is willing to provide new features and easy to use interfaces to their customers. This growth affected everyone's life. In this paper the memory management in various operating systems of the mobiles with their advantages and disadvantages are represented.Here we compare the different memory management system of real life operating system.

Keywords: Memory management, windows, Symbian, blackberry, iOS, Android

I. INTRODUCTION

With increasing demand of mobile phones in people, every company is trying to give their best in the market. Thousands of phones are available in the market with different Mobile Operating System. Scope of this paper represent about the memory management in different mobile operating system. Memory management system is important part of each operating system. The main function of this system is to manage the memory system of random access memory and the other storing devices which are available on the system. Main task that come under the memory management are allocation and deallocation of memory to the various process running on the system .The memory management system should be optimized as its function affects the speed and the performance of the operating system.

Memory management in the mobile Os is the process where we manage the memory of various OS in mobile phone in a controlled way so that all the process can execute without any difficulty. Memory management resides in the OS (operating system), and in programs and applications.

Vendor	Programming Language	Operating system	Memory management	Application store
Symbian	C++	Symbian OS	Flash memory	Nokia OVI store
Open handset alliance	Java	Android OS	DVM(Dalvik virtual machine)	Android market
Apple	C	iPhone Os	ARC(Automatic reference counting) where we have to retain and release the objects	iPhone App store
RIM	Java	Blackberry OS	JVM	Blackberry App world
Microsoft	Visual C#/C++	Windows phone	-Windows working on 32 bit x86 systems accesses up to4GB of physical memory	Windows mobile market place

Memory management is the process of controlling and coordinating computer memory, assigning portions called blocks to various running programs to optimize overall system performance.

A. Binding of Instructions & Data to Memory:

- ***a. Compile time****:* If memory location known in advance, *absolute code* can be generated; must recompile code if starting location changes
- ***b. Load time****:* Must generate *re-locatable code* if memory location is not known at compile time
- ***c. Execution time****:* Binding delayed until run time if the process can be moved during its execution from one memory segment to another. Need hardware support for address maps (*registers).*

B. Logical vs. Physical Address Space:

- ***a. Logical address***– Created by the CPU; also referred to as *virtual address*
- ***b. Physical address***– address seen by the memory unit Logical and physical addresses are the same in compile-time and load-time address-binding schemes; logical (virtual) and physical addresses differ in execution-time address-binding scheme
- ***c. Swapping-*** A process can be swapped temporarily out of memory to a backing store, and then brought back into memory for continued Execution.
- ***d. Backing store–*** fast disk large enough to accommodate copies of all memory images for all users; must provide direct access to these memory images
- ***e. Swap out, Swap in***– Those process which are not in used is swap out to backing store or you can say that lower priority based process are swapped out and higher priority based process are swap in, means come into the main memory.

Modified versions of swapping are found on many systems (i.e., UNIX, Linux, and Windows)

Memory management of different mobile OS are explain here

II.SYMBION

- ***a. Memory Management of Symbian phone:*** Phones with Symbian OS use Flash memory as their store mechanism of system code and user data. Flash

memory is silicon based on non volatile storage medium that can be programmed and erased electronically. Flash memory is of two types: NOR and NAND. The names are just because of their fundamental silicon gate design [1]. Its OS phones make best use of both types of Flash through the Selection of file systems. The built in system code read only drive, known as the Z: drive. The Z: drive is also referring as a ROM image. Various application and user data are stored internally in writable C: derive [1].Today a Symbian phone typically uses between 32 and 64 MB of Flash for the code and user data, which is the total ROM budget. Many techniques are used by the Symbian for minimize the code and lesser data sizes within a phone, such as THUMB instruction set, pre linked XIP images, compressed executables, compressed data formats and coding standards that emphasize minimal code size.

A. iPhone OS (iOS):

a. Memory management in iOS: ARC the memory management in iOS which means Automatic reference counting where we have to hold and release the objects. Now it supports ARC where we don't need to add hold and release. Actually the Xcode takes care of the job automatically in compile time.

b. Memory Management rules: We possess the objects we create and we have to afterwards release them when they are not in need. Retain can be used gain ownership of an object that we did not create. We have release these objects too when it's not needed. Don't release the objects that we don't own.

B. Handling memory in ARC:

You don't need to use release and hold in ARC. So, all Memory management is the programming discipline of managing the life cycles process of objects and freeing them when they are not in use. Organizing object memory is a matter of act; if an application doesn't free unneeded objects, its memory footprint grows and performance affects. Memory management in a Cocoa application that doesn't use garbage collection which is based on a reference counting model. When you generate or copy an object, its hold count is 1. Thereafter other objects may express an ownership interest in your object, which increments it's retain count or hold count. The owners of an object may also turn down their ownership interest in it, which decrements the retain count. Hold count when becomes zero, the object is destroyed.

C. BlackBerry OS:

a. Memory management in BlackBerry OS:

Memory is usually portioned into three parts:

a) [2]Application Memory (~128MB)-a dedicated memory space for application storage and overhead
b) Device Memory (~850MB)-for storing files and other media
c) Memory Card (optional)-an optional method of file storage[2]

JVM handles the memory management for many third parties application in blackberry Os; otherwise specialized MDS is used.Swaping and handling of data between flash memory and SRAM is also controlled by JVM. It also managed the garbage collection and memory allocation. Limited capacity of memory is handled by the low memory manager which is executing mainly to deal with this problem.

[3]When a free flash memory is going down under a certain threshold or various available objects decreases under a specific limit then the low memory manager frees the accessible memory. Third party applications and standard application must work with this interface in order to delete the low priority data when the low memory manager receives an event of low memory.

D. Windows Phone:

a. Memory management in Window phone: Windows systems works on 32 bit x86 which can use up to 4GB of physical memory. [4]Each process gets 4GB logical space in windows. Lower 2GB is present for the user mode process and upper 2GB is reserved for Kernel mode code of the Windows. Paging feature is used by the windows for memory management.

Paging permits the software to use a logical memory address than the physical memory address, paging unit of the processor translates the logical address into the physical address. This allows ever process in the system to have its own 4GB logical address space.

Windows provides an independent, 2 GB user address space for each and every application (process) in the system. To the application only 2 GB of memory is appeared to exist than the total memory available. When an application requests more memory than the existing memory, Windows NT satisfies the request by paging noncritical pages of memory from this and/or other processes to a page file and freeing those physical pages of memory. Thus a global heap exists for no long in the Windows NT. In windows every process has the private 3bit address space from which all of the memory for the process is allocate including code, resources, data, DLLs (dynamic link libraries), and dynamic memory. The system is still limited by whatever hardware resources are available, but the management of available resources is performed independently of the applications in the system [4].

E. Android:

Operating system of Android phones is open source which is Linux based .It's first version was released on Nov 12th 2007. Android uses its own run time and virtual machine like Java and .NET, to manage application memory. Unlike either of these frameworks, the Android run time also handles the process lifetimes. Android ensures application responsiveness by stopping and killing processes as necessary to free resources for higher-priority applications.

Each Android application has its own separate process within its own Dalvik instance, which holds all responsibility for memory and process management to the Android run times, which stops and destroys processes as necessary to manage resources.

On top of a Linux kernel Dalvik and the Android run time sit, that maintains low-level hardware interaction including drivers and memory management, while a set of APIs provides access to all of the under- lying services, features, and hardware.

DVM is a register-based virtual machine that's been optimized to ensure that a device can run multiple instances

efficiently. It relies on the Linux kernel for threading and low-level memory management.

F. The Dalvik Virtual Machine:

Dalvik virtual machine is one of the main elements of Android OS. Android uses its own VM designed to make sure that many process execute efficiently on a single device [5].

DVM uses the device's underlying Linux kernel to handle low-level functionality including security, threading, and process and memory management.

Dalvik as a middle tier controls all Android hardware and system services. Using a VM to host application execution, developers have an abstraction layer that guarantee they never have to worry about a particular hardware implementation.[3]

The DVM carry out Dalvik executable files, a format optimized to make sure minimal memory foot- print. The **.dex** executables are formed by transforming Java language compiled classes using the tools supplied within the SDK.

G. Understanding Application Priority and Process States:

The fashion in which processes are destroying to recover resources is determined by the priority of the hosted applications. An application's priority is equal to its highest-priority component.

The process that has been at a lower priority longest will be killed first, When two applications have the same priority. Process priority is also affected by inter process dependencies; if an application has a dependency on a Service supplied by a second application, the secondary application will have at least as high a priority as the application it supports.

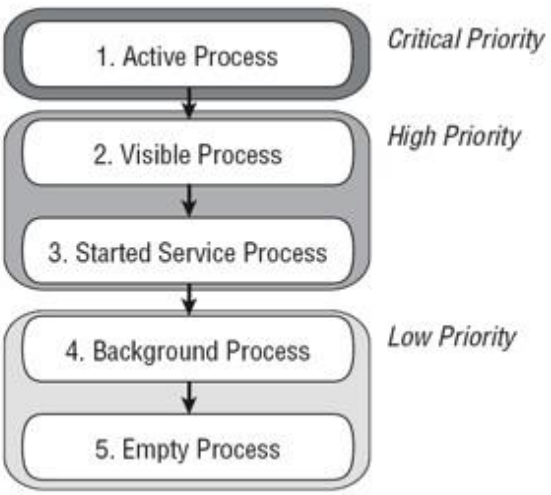

Figure: 1

This is essential to build your application correctly to make sure that its priority is appropriate for the work it's doing. If you don't, your application can be destroy while it's in the middle of something important.

The following list details each of the application states shown in Figure

Active Processes These are the processes which hosting applications with components presently relate with the user. Android is trying to keep responsive by reclaiming resources by these processes. There are generally very few of these processes, and they will be killed only as a last option.

H. Active processes include:

a. If Activities are in the foreground then they are in "active" state; that is, they are responding to user events.
b. Activities, Services, or Broadcast Receivers that are currently executing an onReceive event handler.
c. Services that are executing an onStart, onCreate, or onDestroy event handler.

Visible Processes Visible processes are very few in generally, and they'll only be destroy in severe circumstances to permit active processes to continue. Visible, but inactive processes are those hosting "visible" Activities. As the name suggests, visible Activities are visible, but they aren't in the foreground or responding to user events. This happens when an Activity is only partially hidden.

Started Service Processes hosting Services that have been started. Services hold ongoing processing that should continue without a visible interface. Because Services don't interact directly with the user, they receive a slightly lower priority than visible Activities. They are still considered to be foreground processes and won't be killed unless resources are needed for active or visible processes.

Background Processes hosting Activities that aren't visible and that don't have any Services that have been started are considered background processes. There will generally be a large number of background processes that Android will kill using a last-seen-first-killed pat- tern to obtain resources for foreground processes.

Empty Processes for improving the system performance, Android often keep applications in memory after they have end of their lifecycle. Android sustain this cache to pickup the start-up time of applications when they're re-launched [5].

III. CONCLUSION

Finally we conclude that memory management in various mobile operating systems is different. Every Os has its own unique features, loopholes and has a great evolution chart. Android manages opened applications which are running in the background, so officially you shouldn't care about that. This means that it closes the applications when the system needs more memory. However, most android users are not very satisfied with how it does its things because sometimes it leaves too many processes running which causes sluggishness' in everyday performance. Whereas Symbian are still lacked the updating of latest technologies. Symbian has a code of C++variant which is very difficult and it takes more time to executes his application. Blackberries phone quality is better than iPhone quality. Windows phone are also performing well. But iOS makes the boom in today scenario.

IV. REFERENCES

[1]. www.ijcait.com

[2]. Journal of software Engineering and application vol.08No.)3(215),Article ID:54397,12 pages 10.4236/jsea.2015.83012

[3]. International Journal Of advanced research in computer science and engineering Volume 3,Issue 6,June 2013 ISSN:2277 128X

[4]. "Gaurav Jindal" International Journal of Computer Applications & Information Technology Vol.I Issue III,November 2012(ISSN:2278-7720)

[5]. "T.N.Sharma,Mahender kr.BEniwal,Arpita Sharma" International Journal of Advancements in Research & technology,Volume2,Issue 3,March-2013 ISSSN 2278-7763