Traditional vs. Mobile Operating Systems

CS-502
Fall 2011

James Process
Andrew Sawchuk
Jeffrey Martin
Mike Sandman

Overview

- Introduction
- Design
- Development
- Test
- Packaging/Distribution
- Wrap-up

Introduction

- Discuss the differences between Traditional and Mobile Operating Systems
- Focus will be on Android vs. Linux and iOS vs. Mac OS** X
- Cross compare all four, Android vs. Linux vs. iOS vs. Mac OS X from the developer perspective
- Objective - What is the mental model a developer must have when developing for mobile vs. traditional operating systems

A New Age of Operating Systems

- Why create new operating systems for phones, tablets and other devices? Why not just use existing ones?
  - End user has different expectations/requirements from their mobile device than their laptop/desktop
  - More simple/cleaner design that can be driven from a very limited number of peripherals i.e. only has a touch screen, and maybe a keyboard
  - Added reliability needed by user as users depend on their mobile devices for daily functions (e.g., calling, email, calendars, etc.)

Design

Android vs. Linux

- Languages
  - Linux – Most modern languages available: C/C++, Java, JavaScript, Python, Ruby, etc.
  - Android – Must be coded using Java. JNI allows the use of other languages, but all system APIs must be called from Java code
- Integrated Development Environment (IDE)
  - Linux – Commonly text editors are used in conjunction with cmdline compilers. But several other options are available based on language
  - Android – SDK is available as an integrated tool for Eclipse and also a cmdline based set of tools for emulation, debug, and compilation are available

Design

Android vs. Linux

- Defining your audience
  - Version Compatibility
    - Android – different versions and upgrade not available for all devices
    - Must weigh increased feature set of newer versions versus audience size of including older versions
    - Linux – Must consider various architectures and target distributions (e.g. RHEL, OpenSUSE, Ubuntu, etc.)
  - Peripherals
    - Linux – Keyboard, mouse, monitor, all others should be available for purchase.
    - Android – No guarantee on hardware, though most mobile devices have at least a touch screen. May have others, physical keyboard, trackball, front facing camera, etc.
      - Most likely not to upgrade to a new device for a specific peripheral for your app

Design

iOS vs. Mac OS X

- Languages
  - Mac OS** X Based on OPENSTEP, Mach, BSD and Mac OS*:
    - Apple** SDK: Xcode
    - Darwin kernel
    - Native language/SDK is Objective-C based
    - Unix languages (C, Objective-C, java, scripting languages …)
  - iOS is OS X Based but supports Objective-C as only language
    - Library support provided by "Frameworks" ***
- Integrated Development Environment (IDE)
  - Mac OS X: Xcode provided by Apple**
  - Third-party IDEs (such as Eclipse) supported

Design

iOS vs. Mac OS X

- Integrated Development Environment (cont.)
  - iOS SDK provided and controlled by Apple Inc.
    - Includes Xcode which provides the editor, debugger and compiler
    - Interface builder –Program for creating the Graphical User Interface and associating to the application code
    - Simulator – iOS virtual machine allows testing of applications on a MAC**
    - Tracing and Profiling (Instruments) – Application profiler providing details on memory usage and system performance. Based on Sun Microsystems Dtrace package.

Design

iOS vs. Mac OS X

- Defining your audience
  - Version Compatibility
    - iOS – Version Restricted and controlled by Apple Inc.
      - Apple is the sole OS and hardware platform developer
      - Application developers need to consider the iOS version they are developing to
      - Newer features may change development approach (Ex: Automated Reference counting*)
    - Mac OS* X– Versions restricted and controlled by Apple**
      - Only one manufacturer of hardware/OS
      - (Basically the same thing as iOS)

Design

iOS vs. Mac OS X

- Defining your audience (cont.)
  - Peripherals
    - Mac OS* X
      - FireWire Reference Platform 2.0
      - Bluetooth (Apple's Bluetooth Stack, based on Bluetooth SIG Standard)
      - Multiple USB devices (camcorders, digital cameras, cell phones)
      - DVD-ROM drive, mouse, keyboard, monitor
    - iOS
      - Supports "Bonjour"* for network device discovery
      - Bluetooth for Peer-to-Peer connectivity
      - USB Cable can be used

Design

Summary

- Linux/Mac OS* X
  - Freedom to choose IDE/Language
  - More peripherals/upgrades available
  - Operating systems released as standalone software
  - Different Linux architectures and distributions, not so for Mac
- Android
  - Restricted to Java, minimal IDE flexibility
  - Non-standard hardware
  - Varying OS version per device
  - OS is distributed by hardware mfg
- iOS
  - Restricted to Objective C
  - Slight hardware variation between older/newer devices but for the most part standardized
  - Single hardware vendor, standardized hardware
  - Latest version generally available

Development

Linux

- Memory Management
  - Large address spaces via virtual memory
  - Each process has its own virtual address space, and cannot touch others
  - Virtual memory mapping is managed by the Linux kernel
- Process Lifecycle
  - Each process has a corresponding metadata structure within the Linux kernel
  - Processes are started, scheduled, and destroyed by the Linux kernel

Development

Android

- Process and Memory Management
  - Each Android application runs in its own Dalvik VM
    - Dalvik VM is memory-optimized so that multiple instances may be run on the same device
    - Threading and low-level memory management is done by the Linux kernel
  - Android Runtime
    - Manages processes and memory at a higher level
    - Each process is assigned a state (and associated priority)
    - Android runtime kills tasks to free up memory based on priority of task

Development

Application Security

- Linux
  - Running application inherits privileges of the user running it
  - Every file has permissions and filetype embedded straight into the file
- Android
  - Each application must request needed permissions (e.g., read/write storage, access contact information, access the Internet, etc.)
  - When installing, end user must “accept” list of requested permissions

Development

User Interface

- Linux
  - Command shell
  - Various graphics libraries available (e.g., OpenGL, TK, etc.)
  - Several GUIs (window managers)
    - Different window managers are bundled with different distributions
- Android
  - Standardized GUI provided by the Android platform
  - GUI may be tweaked slightly by device manufacturers

Development
Mac OS X

- Memory Management
  - Sparse virtual memory scheme (one of the major upgrades from Mac OS9)
  - Garbage collection
- Process Lifecycle
  - Multiple processes allowed
  - Unix/Linux style process IDs and management, processes managed within kernel
  - Processes can be started/terminated/force quit from command line or Desktop

Development

iOS

- Memory Management
  - Same virtual memory scheme as Mac OS** X
  - Garbage collection
    - Viewed differently by the beholder
    - If memory is low, app is requested to release, if it does not it is terminated (See process life cycle).
    - Application memory has to be either marked for release (ARC) or manually released (MRR)*
      - Automated reference counting (new)
      - Manual retain and release

Development

iOS

- Process Lifecycle
  - “Multitasking” is supported in newer iOS versions*
    - Restrictions are applied to back ground tasks (Playing audio or cell calls)
    - App operations are expected to be short in duration if running in the background (exception with previous statement).
    - Processes not in the foreground (only running app) are suspended
    - If memory is running low on the system these apps are stopped

Development

Application Security

- Mac OS X
  - Running application inherits privileges of the user running it
  - Every file has permissions and filetype embedded straight into the file
  - Some changes require administrator permissions (different from "root" user)
- iOS
  - Application installation done via an App store (Apple iTunes* or private)
  - Data encryption done via KeyChain and Cryptography Services
  - Access provided by application signature
  - Keychain items can be shared across applications
  - Only the keychain item is encrypted on backups
  - Don't store the password information directly (and always look up latest threats)
  - No actual "users" are present on the system

Development

User Interface
Mac OS Xand iOS

- Mac OS X
  - GUI is Apple* desktop with dock and application bar
  - Command line interface (Terminal) similar to Linux shell, runnable from GUI
- iOS
  - Graphical User interface only
  - Single Application user window displayed at a time
    - Note multiple views can be provided via the application
  - Different hardware has different resolutions

Development

Summary

- Linux/Mac OS* X
  - Separate virtual memory per process
  - Large address space
  - Graphical and command line interface
  - Large number of concurrent processes
  - Application privileges match those of user running it

Development

Summary

- iOS
  - Same virtual memory management as Mac OS* X
  - Memory resources limited, multitasking limited
  - Security is restricted to Keychain and cryptography methods
- Android
  - Each process runs in its own VM
  - Process lifecycle managed by Android runtime
  - Each application has its own permissions

Test and Debug

Android vs. Linux

- Similar mechanics for both Linux and Android
- Linux
  - Bare metal or Virtual Machine
  - Can print messages to stdout
  - GNU Debugger (GDB)/Kernel Debugger (KDB)
- Android
  - Physical device
  - SDK provides emulator
    - Can configure OS Version, peripherals, screen size, disk size
  - Can print messages out to system log, viewable using ADB logcat
  - Android Debug Bridge (ADB)

Test and Debug

Android vs. Linux

Key difference between Linux and Android debugging…

- Client-server debugging vs. local debugging
- Both Android and Linux support client-server debug models
- Linux supports local debugging
- Very limited debug tools on standalone Android device

Test and Debug

Mac OS X

- LLDB
  - Part of LLVM open source project
  - Included as part of Xcode v4
- Apple Developer Tools
  - Suite of test tools provided by Apple Inc.
  - 10.6 and later
  - Packaged with Xcode but not installed by default
- Use own debugger of third-party IDEs

Test and Debug

iOS

- Xcode debugging interface provided
  - SDK version should be greater then or equal to the iOS version developed to.
  - Console provided, logging facilities should be used
- USB Connection needed for debugging
- Simulator included in Apple Inc. SDK, runs on Mac OS X
- Use of each i[Device]* should be tested on

Test and Debug

Summary

- Linux/Mac OS X
  - Debuggers as part of IDE
  - Multiple choices
  - Traditional test and debug approach
- Android
  - Single debugger
  - Debugging easier in client/server environment*
- iOS
  - Single debugger (Xcode)
  - Debugging can be done on phone or on iOS Simulator*

Packaging

Android vs. Linux

- Linux – Several different packaging types
  - RHEL/CentOS - .rpm file format
  - Debian/Ubuntu - .deb file format
  - Others…
- Android – One package type
  - .apk files for all android releases/platforms

Packaging

iOS vs. Mac OS X

- Mac OS X- .pkg files
  - Installer – installation wizard for Mac
  - Often packaged in a .dmg (disk image)
  - Sometimes need to drag executable into Applications folder manually
- iOS – Application bundles
  - Inventory list of the files for the Apps “Information Property list”*
  - Application content files (Program, data files “Resources”)

Distribution

Linux

- Various software repositories, based on Linux distribution (e.g., Yum, Apt-Get, Pacman, etc.)
  - Private and public entities can create repositories, which end users then use that repository's client to connect and download software packages
  - Developer must manually package and submit software for each repository
- Source code, binaries, and other packaging may optionally be distributed via other means

Distribution

Android

- Original app repository is the Android Market, which is hosted by Google
- Third-party app repositories are also available, such as the Amazon Appstore for Android
- Both Google and Amazon repositories:
  - offer license enforcement mechanisms (copy protection)
  - actively police apps in their repositories
  - take a cut of app sales revenue
- Developer may distribute source and/or .apk via other means

Distribution

Mac OS X

- Third-party distribution
  - Developers make software available on website/in stores
  - No approval with Apple needed
- App store in later versions
  - Through iTunes
  - Modeled after iOS App store
  - Need to submit through Apple* development process

Distribution

iOS

- Apps registered through Apple Inc. iTunes*
- Must be a registered developer with Apple Inc.
  - Additional work beyond registration may be needed.
- Can distribute beta versions of app to a limited audience
- Private distributions can be done in Enterprises
- Educational access is available

Packaging/Distribution

Summary

- Linux
  - Multiple repositories (based on distribution)
  - Multiple package types
  - Option to release independently
- Mac OS* X
  - Independent distribution
  - Core OS from Apple*
  - iOS-like App Store in later versions

Packaging/Distribution

Summary

- Android
  - Single format (.apk)
  - Typically distributed through Google on Android Market
  - Option to distribute third-party e.g. Amazon Market
  - Independent distribution possible
- iOS
  - Single package format
  - Apps must be submitted to Apple for distribution (approval)
  - No third-party commercial distribution

Wrap-up

- Less flexibility in development environment on iOS/Android
- iOS and Android require tighter memory management/more controlled access to devices
- Mobile app development targeted for a more specific and known set of devices
- Application distribution has tighter regulation on iOS and Android

Resources/References

Linux/Android

- Android
  - Android Developers
    - http://developer.android.com/index.html
  - Android Open Source Project
    - http://source.android.com/index.html

- Linux
  - The Linux Documentation Project
    - □ http://tldp.org/LDP/tlk/mm/memory.html
  - IBM on Linux Memory Management
    - http://www.ibm.com/developerworks/linux/library/l-linux-process-management/
  - ResearchBooth.com on Linux Security
    - http://www.researchbooth.com/categories/computers/open_source/understanding_linux_security.php

Resources/References

iOS/Mac OS X

- Apple Inc. Development portal
  - http://developer.apple.com/
- Apples' Open Source resources
  - http://www.opensource.apple.com/
  - http://developer.apple.com/opensource/
- MAC OS Forge:
  - http://www.Mac OSforge.org/
- Other Useful resources:
  - http://cocoadevcentral.com/
  - http://boredzo.org/cocoa-and-cocoa-touch-intro/
  - http://www.w3.org/Consortium/
  - http://www.raywenderlich.com/

Resources/References

iOS/Mac OS X

- Publications
  - The iOS 5 Developer's Cookbook: Core Concepts and Essential Recipes for iOS Programmers, Third Edition, Erica Sadun, Addison-wesley Professional, November 14 2011 ISBN-13 978-0-321-75426-4
  - iOS5 Programming Cookbook, Vandad Nahavandipoor, O’Reilly Media, Inc., Updated November 2, 2011, ISBN-13 978-1-4492-1143-8
  - Introducing Xcode 4 Tools for iOS Development. Xcode 4 iOS Development, Steven F. Daniel, August 25, 2011 ISBN: 9781849691307
  - iOS Development Bibliography, Safari Content Team, Safari Books Online August 1, 2011