THE GROWTH OF ANDROID IN EMBEDDED SYSTEMS

Benjamin Zores

Android Builder Summit 2014

1st May 2014 - San Jose, USA

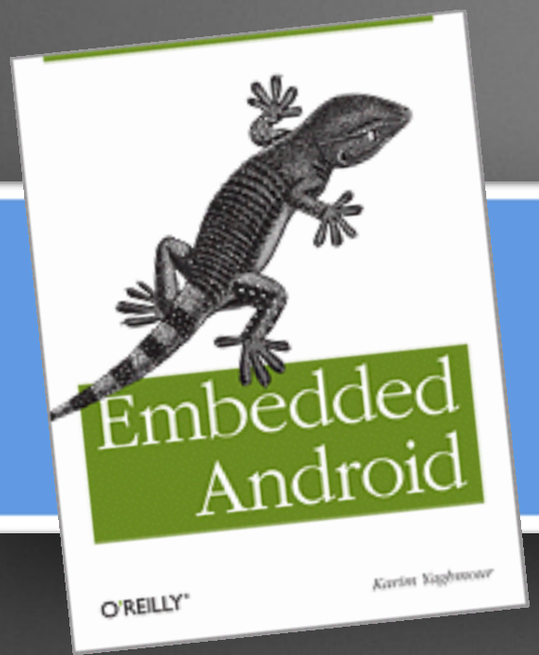
>OPERSYS

Benjamin Zores


benjaminzores


@gxben


#Benjamin Zores



Embedded Android

Karim Yaghmour, O'Reilly - Mar 2013

Android 4: Fondements Internes

Benjamin Zores, Ed. Diamond - Q3'2014

The Android

Operating System

Android Chronology

- Early development at Android Inc. in early 2000s.
- Android Inc. got purchased by Google in 2005 (not Linux-based at this time).
- Architecture revamping lead to HTC G1 first Android smartphone in 2008.
- You know the rest ;-)

Android in Embedded Systems

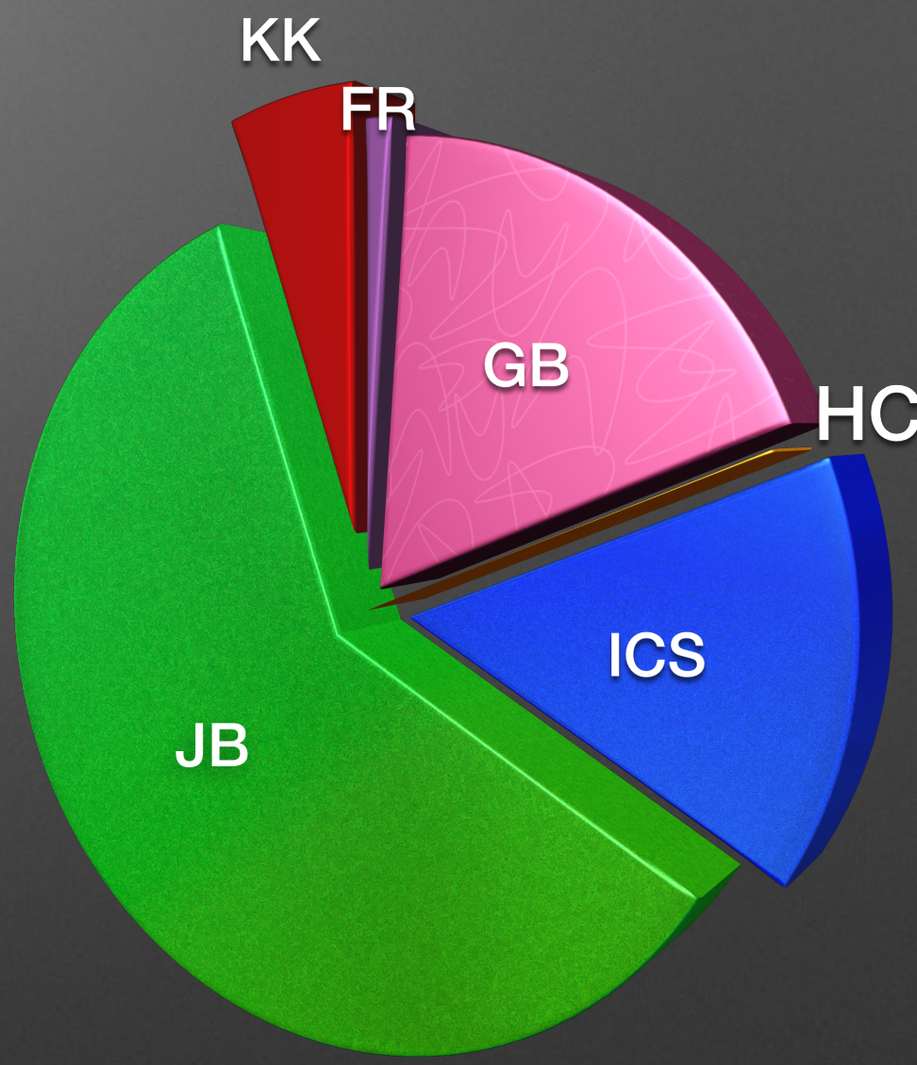
- The industry is (fortunately) not only composed of smartphones and tablets ;-)
- 34% of embedded engineers are considering using Android in 2013.
- May sounds appealing for domestic use markets (STB, IVI …)
- Under the hood, Android however can be a burden for device manufacturers.

Releases History

NAME	VERSION	SDK RELEASE	KERNEL	SDK API	NDK API
N/A	1.0	September 2008	2.6.25	1	N/A
PETIT FOUR	1.1	February 2009	2.6.25	2	N/A
CUPCAKE	1.5	April 2009	2.6.27	3	1
DONUT	1.6	September 2009	2.6.27	4	2
ECLAIR	2.0	October 2009	2.6.29	5	2
	2.0.1	December 2009	2.6.29	6	2
	2.1	January 2010	2.6.29	7	3
FROYO	2.2	May 2010	2.6.32	8	4
GINGERBREAD	2.3 - 2.3.2	November 2010	2.6.35	9	5
	2.3.3 - 2.3.7	February 2011	2.6.35	10	5
HONEYCOMB	3.0	February 2011	2.6.36	11	6
	3.1.x	May 2011	2.6.36	12	6
	3.2.x	June 2011	2.6.36	13	6
ICE CREAM SANDWICH	4.0 - 4.0.2	October 2011	3.0.1	14	7
	4.0.3 - 4.0.4	December 2011	3.0.1	15	7
JELLY BEAN	4.1.1 - 4.1.2	June 2012	3.0.31	16	8
	4.2	November 2012	3.0.31	17	8
	4.3	July 2013	3.0.31	18	9
KIT KAT	4.4	October 2013	3.4.0	19	9

Android Fragmentation (Apr. 14)

VERSION	CODENAME	API	DISTRIBUTION
2.2	Froyo	8	1.1%
2.3.X	Gingerbread	10	17.8%
3.2	Honeycomb	13	0.1%
4.0.X	Ice Cream Sandwich	15	14.3%
4.1.X	Jelly Bean	16	34.4%
4.2.X		17	18.1%
4.3		18	8.9%
4.4	KitKat	19	5.3%



A Life

Without GNU

Unique System & Software Architecture

- Android is based on **modified Linux kernel** and **300+ OpenSource software components**.

- There ends the ressemblance with any other embedded and/or desktop Linux distribution.

- **Redesign** or replacement of **fundamental building blocks**

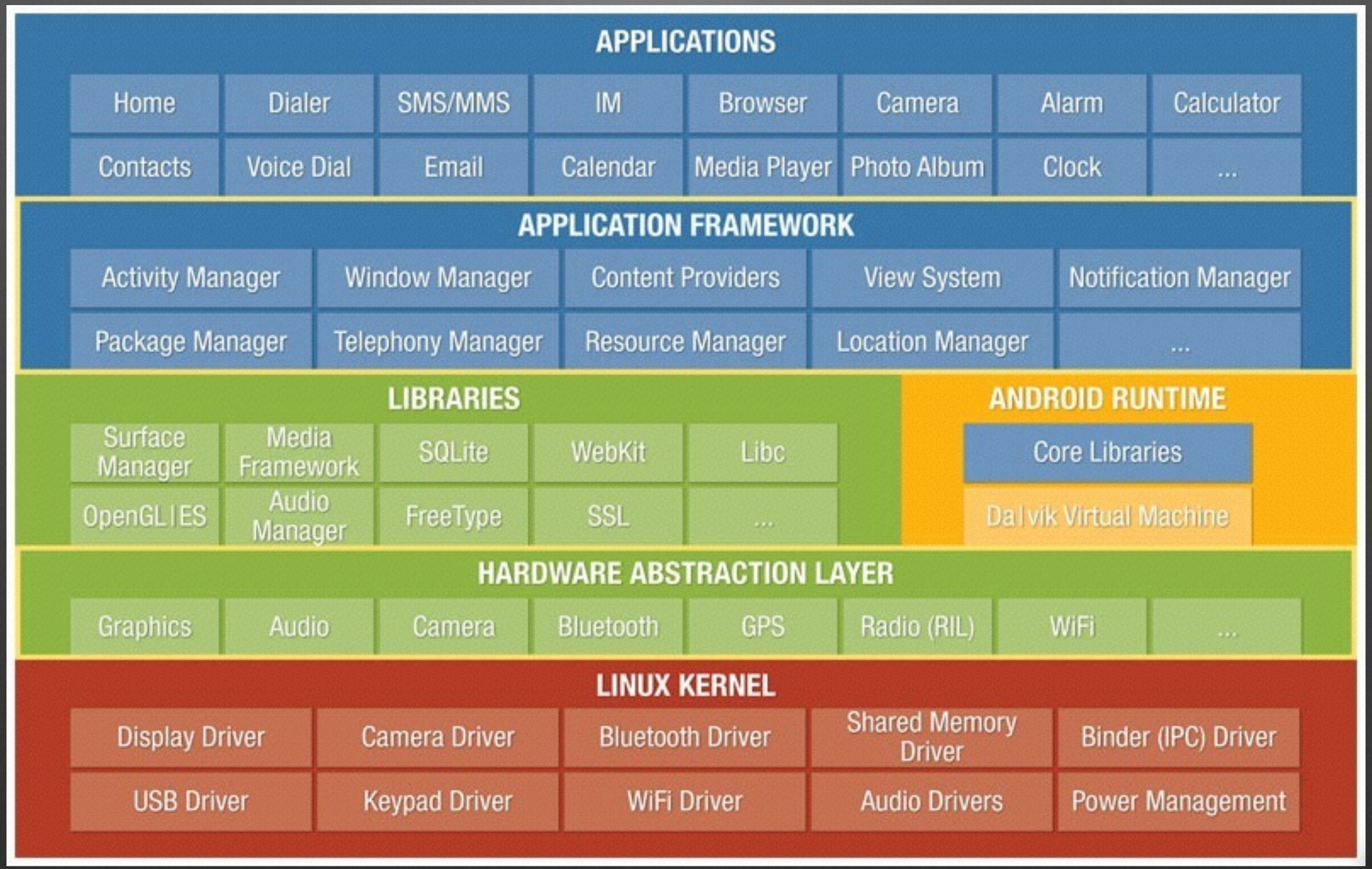
- Got rid of **Glibc**, **X.org** / **Wayland**, **Busybox**, **PulseAudio**, **GStreamer**, **GTK** / **Qt** …

“Proprietary” Development Model

- Often referred to as « clopen »
  (for closed/open)
- NOT developed in a community way.
- Sources drop depends on Google’s willingness to share.

- Google got rid of (L)GPL in favor of Apache/MIT/BSD licenses.
  - Safe solution for companies to build devices without fear of further legal complications.

A Life Without GNU



Mobile-Targeted Kernel

- Google introduced several « Androidisms » to vanilla Linux kernel.

- Agressive Power Management Policy
  - WakeLocks, Early Suspend …
  - Desktop follows the « always-on » policy.
  - Android does the opposite.

- Binder IPC Message Bus

Java Application Framework

- Java is quite unpopular with embedded developers.
  - Slow, resource consuming, hard to debug, heavy and complicated to deploy.

- Google introduced its own bytecode: Dalvik.

- Amazing Zygote app server:
  - Framework (2000+ classes) is loaded once and for all in memory.
  - Apps are spawned by Zygote with copy-on-write methods, optimizing resources usage.

Dealing With

Embedded Linux OS

Dealing with Embedded Linux OS

- Embedded Linux available customizations can come in handy.

- Diversity of commercial providers
  - Windriver, Montavista, Mentor Graphics …
- DIY OpenSource Embedded Frameworks
  - The Yocto Project, OpenEmbedded, Buildroot, LTIB …
- SoC vendors specific BSPs

- Mature solutions, allows you to suits your exact needs
  - -> But where to start from ??!
  - -> To which price ??
  - R&D efforts usually are spent on maintaining system instead of bringing values.

Dealing with Embedded Linux OS

Android (while being forked by various groups) is unique.

Device manufacturers surely customize it, but there's only one project you want to be compatible with, and it's actively maintained for you the Google way.

Reasons For

Android's Attraction

Rich Application Framework

- GNU/Linux brings you choice to do things at your convenience.
- Android comes with a single stable long-term API and excellent SDK.
- Standardized Ecosystem for app-developers and 3rd-party partners.
- Build apps once for multiple targets to drastically save costs and efforts.

Aggressive Time-to-Market

- Stick to HW reference design, adapt the platform and release your new device in a few months !
  - Though far from being easy
  - Requires Android-specific expertise and knowledge of the OS internals !

Focus on “What Really Matters”

- You don’t have to care about the platform and framework anymore.
  - Board bring-up is time consuming and no one wants to waste more time re-inventing yet another embedded distribution.
- Developers actually only focus on areas that add commercial value (i.e. apps)

Open Source

- Android remains 100% tunable
  - Though not developed in a community way.

- Provides companies a feeling of safety regarding potential legal threats and licensing.
  - Thanks to Apache/BSD/MIT licensing.

Under-The-Hood

Culprits

Standardization & Economy

- SoC development costs have grown in complexity and difficulty of integration
    - HW manufacturers only invest in volume-driven apps and customers.
- Vendors now feature Linux BSP only as an internal sandbox.
    - Android drives market hence engineering resources allocation.
    - HW vendors don’t invest in Linux as much as they once did.

Android HAL

- Hardware Abstraction Layer
  - Allows device manufacturers to map Android framework API.
  - Specific to each Android release and platform API.

- Proprietary binary blobs prevents easy upgrade and/or ROM customization.
  - -> Customer often are forced to move to next-generation devices instead of upgrading SW :-(

Design Flaws

- Android uses many Open Source software but also reinvented the wheel !
  - -> Mostly for licensing and convenience purpose.

- NOT Real-Time Capable
  - Best Effort approach is 1000Hz low-latency.
  - -> No PREEMPT_RT (proprietary user-space drivers makes it impossible).
  - -> Dalvik VM garbage collector pauses execution context.

Design Flaws

- Terrible Audio and Multimedia Architecture
  - Lots of Java and JNI indirection calls makes it slooooooow …
    - -> Latency issues
  - Ages away from FFmpeg or GStreamer in terms of framework performances, hardware portability and codecs support.

- Castrated Network and Connectivity Layers
  - Can’t handle more than one input network connection at a time (one driver, one type, one interface).
  - Adding things like Bluetooth, WiFi or basic Ethernet support is a nightmare for device manufacturers.

A Trade-off between Performance and Portability

- Appealing Java « write once, run everywhere » framework's philosophy.

- Any serious performance-critical or multimedia app relies on native C/C++ code being done through NDK, cutting down portability.

The limits of “embedded”

- Originally designed for low-power and low-resource devices.
  - Current smartphones feature 4-core Cortex-A9/15, 32 GB eMMC, 2 GB RAM.

- Starting with ICS, it becomes challenging running Android with less than 512 MB RAM and without OpenGLES-compatible GPU.
  - Kit Kat highly improves this behaviour.
  - But hasn’t Android raised the hardware requirements just a bit too high ?

In a Few

Words ...

Conclusion

- Android has brought the Linux kernel to an incredible number of devices.
  - => More than a million devices being activated each day.

- Many manufacturers want Android on their device
  - Sometimes just to follow the trend and be sure not to be left behind the market.
  - Everybody surprisingly wants an app store (why ???)

- Paradoxically, Google has somehow slow down innovation:
  - All devices look and do almost the same.
  - To the extend of MMI and HW assembly quality.

Conclusion

- Embedded Linux remains the OS of choice for :
  - Headless devices
  - SOHO network equipments (routers, AP, servers …)
  - Companies where engineers master Linux development for years.
  - Devices where maximum performances are expected.

- Android makes perfect sense on devices :
  - Featuring an LCD screen with touch-capable display.
  - Intended to be apps-driven.

Conclusion

Android has brought to the market what GNU/Linux misses the most :

One single application framework that allows developers to deal with every single part of the system.

That's All Folks ...

Benjamin Zores


benjaminzores


@gxben


#Benjamin Zores