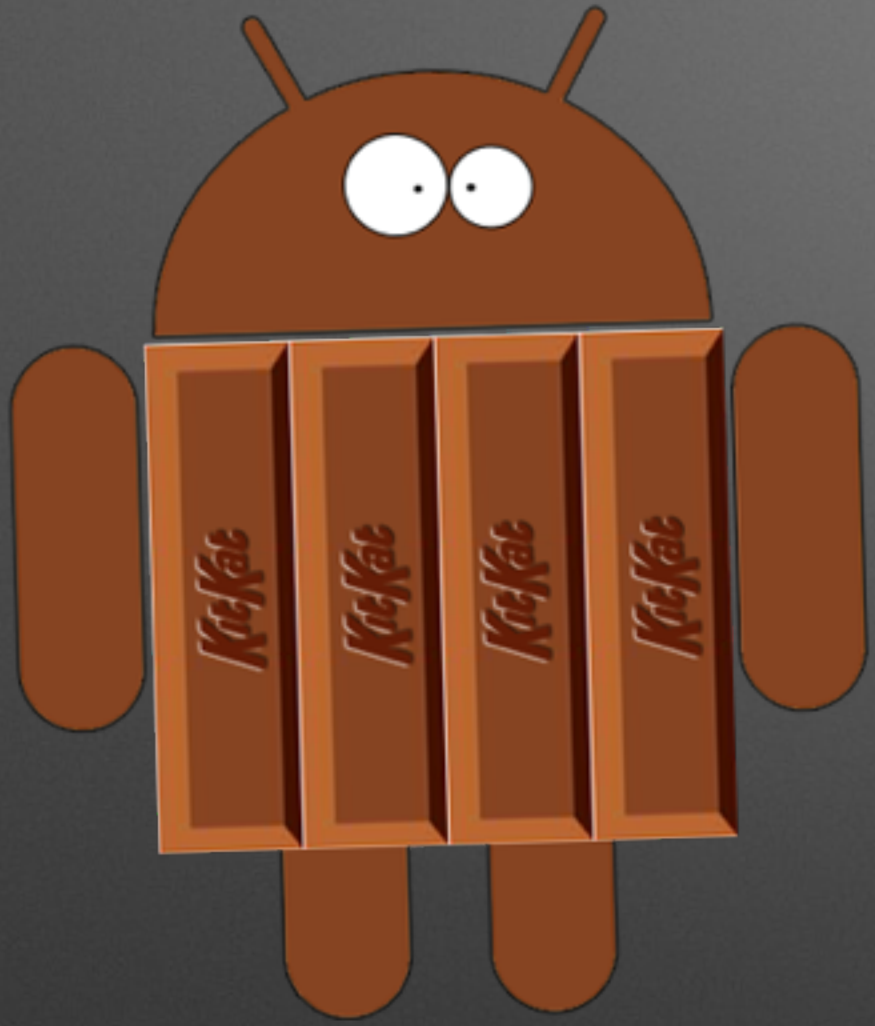


ANDROID Kit Kat INTERNALS

Benjamin Zores

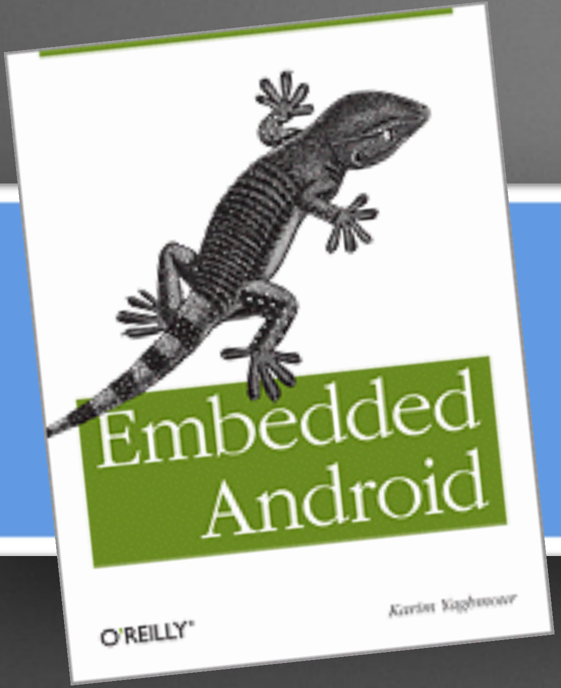
Android Builder Summit 2014
30th April 2014 - San Jose, USA

Benjamin Zores

benjaminzores

@gxben

#Benjamin Zores



Embedded Android
Karim Yaghmour, O'Reilly - Mar 2013

Android 4: Fondements Internes
Benjamin Zores, Ed. Diamond - Q3'2014

PREVIOUSLY ON

THE LINUX FOUNDATION
ANDROID™
BUILDERS
SUMMIT

Android Device Porting Walkthrough

Benjamin Zores
ABS 2012 – 13th February 2012 – Redwood Shores, SF Bay, CA, USA

Alcatel-Lucent

Jelly Bean Device Porting Walkthrough

Benjamin Zores
ABS 2013 – 18th February 2013 – San Francisco, CA

Alcatel-Lucent

Ice Cream Sandwich
Device Porting Walkthrough
ABS 2012 - Focus on ICS 4.0 to 4.0.4

http://goo.gl/1LD92r

Jelly Bean
Device Porting Walkthrough
ABS 2013 - Focus on JB 4.1

http://goo.gl/l2oSZd

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

Today Focus:

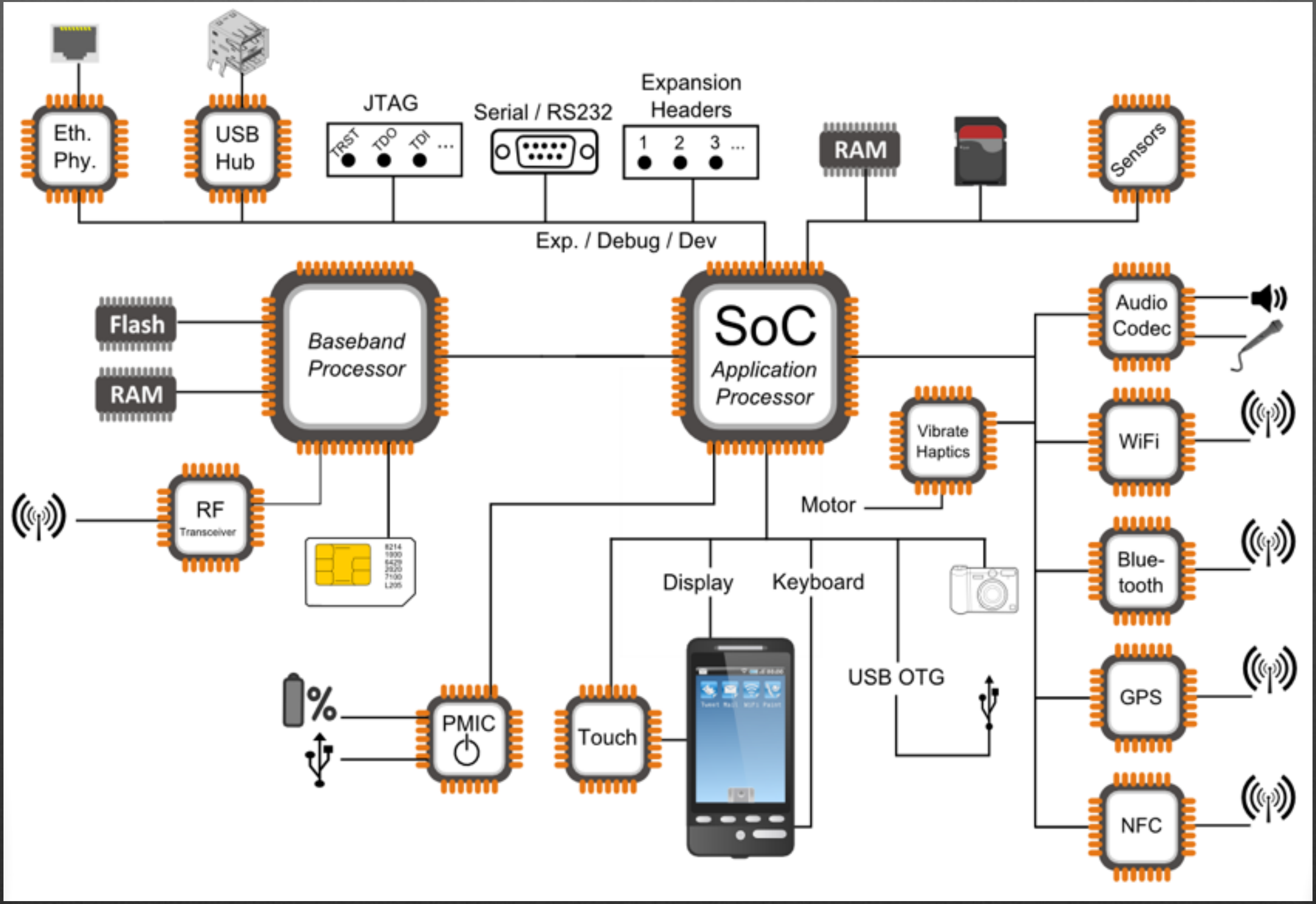
Jelly Bean 4.1 to Kit Kat 4.4

AOSP Source Tree Changes

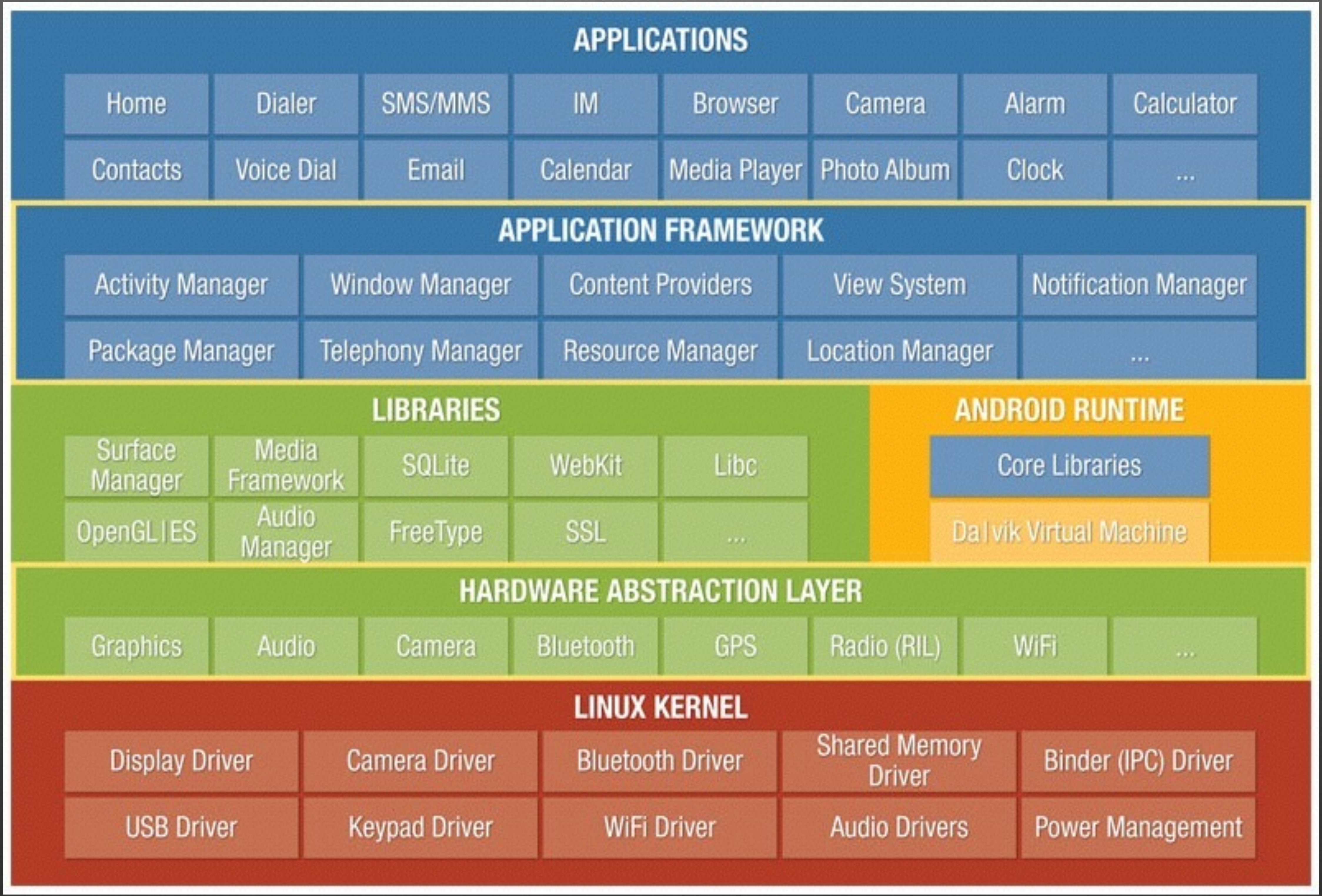
NAME	PROJECTS	SIZE
ICE CREAM SANDWICH (4.0)	230	5.5 GB
JELLY BEAN (4.1)	361 (+57%)	8 GB (+46%)
KIT KAT (4.4)	407 (+13%)	10 GB (+25%)

I’m not an app developer guy (really) !
So instead, we’ll focus on …

Modern Device Internals



Android Software Architecture



Project Butter

Android GFX Architecture

- Triple Buffering
    - Better coordination and sync. of animations between CPU, GPU and display.
- Multi-Threaded GFX Rendering Engine
    - GPU or multi-cores CPU
- OpenGL ES 3.0
    - For NDK and Java apps
    - Optimized HW 2D Rendering
    - ETC2/EAC Texture Compression support.

Vertical Synchronization

- Events are synchronized against display refresh cycles.
- Avoid tearing effect
- Display refresh rate updated from 30 to 60 fps.
- A 16ms timeslot is allocated to each frame to be displayed.
- Apps draw frame at start of VSYNC period.
- SurfaceFlinger starts composition at the start of next VSYNC period.

to be called at each VSYNC event reception.

- HW Composer HAL to be updated:

```
void (*vsync)(const struct hwc_procs* procs,
                    int disp, int64_t timestamp);
```

  - Max timestamp tolerance between caller and callee: 1ms

Vertical Synchronization

- Gralloc HAL:
  - Composition buffers to be acquired/released in sync with VSYNC events.
  - Producers and consumers to notify when they're done with a buffer.
- Explicit sync through:
  - kernel driver, used to sync HW with HW composer
  - HAL v1.1: sync mechanisms through set() and prepare()
  - libsync for user/kernel communication
    - system/core/include/sync/sync.h
    - system/core/libsync

Better Touch Reactivity

- Touch events are HW synchronised with VSYNC events.
- Predict upcoming actions on touchscreen based on finger position:
  - Where it's gonna be at next VSYNC period.
  - Instant CPU boost at wake-up, for better latency.

Multiple Displays Support

- True Dual-Head Support (ICS was limited to mirror/clone)

- HWComposer HAL v1.1:
  - Display detection routine:
    ```
    void (*hotplug)(const struct hwc_procs* procs,
                    int disp, int connected);
    ```
  - Check display capabilities:
    ```
    int (*getDisplayAttributes)(struct hwc_composer_device_1* dev,
        int disp, uint32_t config, const uint32_t* attributes, int32_t* values);
    ```
    - HWC_DISPLAY_VSYNC_PERIOD : VSYNC period (ns)
    - HWC_DISPLAY_{WIDTH,HEIGHT}: screen resolution (pixels)
    - HWC_DISPLAY_DPI_{X,Y}: screen density (pixels / 1000 inches)
  - Blanking:
    ```
    int (*blank)(struct hwc_composer_device_1* dev, int disp, int blank);
    ```

- New Display Manager Service: allow apps to talk to HAL and displays.

Project

Svelte

"We were kind of joking that, when I started, the first thing that I was working on was Project Butter to make the system smoother. The thing is, butter puts on weight. So then I did Project Svelte to lose weight. So now my contribution to Android is basically zero."

Dave Burke, head of engineering for Android at Google.

Goals

- Reduce system memory footprint to allow running on devices with 512 MB RAM.
- Reduce the footprint (memory usage) of the apps that run on a Google Experience (Nexus) device.
- Fix how apps react and crash during bad memory situations.
- Provide better measurement and instrumentation of how apps are running in Android so developers can see how memory-conscious their apps are.

Kernel Same-Page Merging (KSM)

- Introduced in Linux 2.6.32
- Allow processes to share memory pages.
- Kernel scan for identical pages (marked as MADV_MERGEABLE) and merges through COW operations.
- Great for memory but not for performances
    - Consumes CPU, hence battery too !

- Enable KSM through init.rc:

```
write /sys/kernel/mm/ksm/pages_to_scan 100
write /sys/kernel/mm/ksm/sleep_millisecs 500
write /sys/kernel/mm/ksm/run 1
```

Swap to ZRAM

- Uses compressed memory as SWAP.
- Allow more processes to be launched.
- Great for memory but not for performances
  - Consumes CPU, hence battery too !

- Enable Swap-to-ZRAM through fstab:

```
/dev/block/zram0 none swap defaults
zramsize=<size in bytes>,swapprio=<swap partition priority>
```

- and init.rc:

```
swapon_all /fstab.X
```

- ActivityManager makes low-priority threads more elligible to swap.

LowRamDevice

- New ActivityManager.isLowRamDevice() API.
- Allow apps to know if device has 512 MB RAM (or less):
  - Allow them to disable some features in that case.
  - System also kills heavy idle apps and services earlier.
  - System starts services sequentially.
- Device must declared themselves in BoardConfig.mk:

```
PRODUCT_PROPERTY_OVERRIDES +=
        ro.config.low_ram=true
```

System Memory Footprint

- Shrink of system_server and SystemUI provides a few MB here and there.
- DEX caches preload inside Dalvik VM.
- Java framework replaces ArrayMap/ArraySet by HashMap/HashSet for better efficiency.
- Reduction of fonts management cache.

- Option to disable Dalvik JIT: saves 200 kB memory per app
  - Overall system gain of 3 to 6 MB RAM.
  - Done through BoardConfig.mk:

```
PRODUCT_PROPERTY_OVERRIDES +=
    dalvik.vm.jit.codecachesize=0
```

MemTrack

- New HAL plugin to check memory usage.
    - See hardware/libhardware/include/hardware/memtrack.h
- Mostly used to track GFX surfaces allocation.
    - Must interface with HW (e.g. GPU).
    - A texture is allocated in GPU memory (even if dedicated system memory).
        - Invisible from process address space.
- GFX memory can be categorized:
    - Camera, GL, Graphics, Multimedia, Other.
- Stats can be retrieved through:

  getMemory(<pid>, MEMTRACK_TYPE_GL)

ProcStats

- Allows developers to track their applications memory consumption.
- Provides execution time metrics for apps and background services.
- Provides continuous metrics, not instant snapshot.
- Stats are automatically retrieved by system
    - No special compilation option is required.

- Started through:

```
adb shell dumpsys procstats –details
```

Now let’s consider an app with 2 services

(FirstService and SecondService)

ProcStats

```
* com.test.procstats / u0a51:
   * com.test.procstats / u0a51:
          TOTAL: 100% (4.4MB-5.0MB-6.1MB/3.0MB-3.1MB-3.1MB over 3)
            Top: 1.7% (6.1MB-6.1MB-6.1MB/3.1MB-3.1MB-3.1MB over 1)
        Service: 90% (4.4MB-4.4MB-4.4MB/3.0MB-3.0MB-3.0MB over 2)
     Service Rs: 8.1%
   * com.test.procstats.FirstService:
     Process: com.test.procstats
         Running count 2 / time 0.23%
         Started count 1 / time 0.23%
       Executing count 2 / time 0.01%
   * com.test.procstats.SecondService:
     Process: com.test.procstats
         Running count 1 / time 92%
         Started count 1 / time 92%
       Executing count 1 / time 0.01%
```

FirstService consumed 0.23% of app's global execution time.

SecondService consumed 92% of app's global execution time.

ProcStats

```
* com.test.procstats / u0a51:
   Process com.test.procstats (3 entries):
    Screen On  / Norm / Top       : +30s259ms
* com.test.procstats / u0a51:
   Process com.test.procstats (3 entries):
    Screen On  / Norm / Top       : +30s259ms
               Service   : +26m35s118ms (running)
               Service Rs: +2m23s130ms
               TOTAL     : +29m28s507ms
    # [...]
    mActive=true
    mNumActiveServices=1 mNumStartedServices=1
   Service com.test.procstats.FirstService:
    Process: com.test.procstats
    Running op count 2:
     Screen On  / Norm / +4s116ms
            TOTAL: +4s116ms
    # [...]
   Service com.test.procstats.SecondService:
    Process: com.test.procstats
    Running op count 1:
     Screen On  / Norm / +27m4s66ms (running)
            TOTAL: +27m4s66ms
```

Let's check for
real execution times:
adb shell dumpsys -a

mNumActiveServices=1
=> Service is still running !
Guess why battery is running out ?

FirstService
ran for 4s

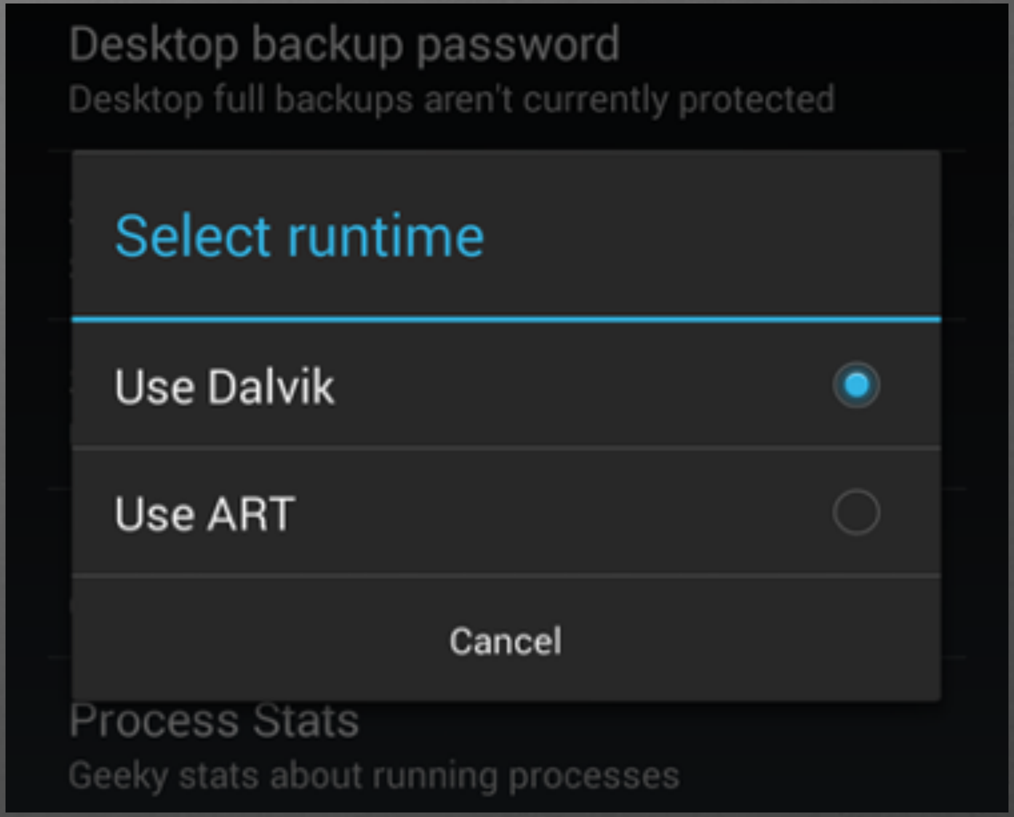
SecondService
ran for 27 min.

Android RunTime

- ART -

Android RunTime

- New experimental VM.
  - See libcore/libart.
- Aims at superseding Dalvik anytime soon.
- Though not yet 100% compatible with Dalvik (may prevent application from running correctly).


Desktop backup password
Desktop full backups aren't currently protected

Select runtime

Use Dalvik ●

Use ART ○

Cancel

Process Stats
Geeky stats about running processes

Can be enabled through **Settings > Developer Options**

Android RunTime

- ART is supposed to be more efficient than Dalvik.

- Today devices have more memory
  and better CPUs than once Dalvik was designed.

- Can be added/enabled at build in
  build/target/product/core_minimal.mk:

```
PRODUCT_RUNTIMES :=
        runtime_libdvm_default
PRODUCT_RUNTIMES += runtime_libart
```

Android RunTime

- Replaces JIT by AOT ("Ahead-of-Time") approach.
  - Native code is compiled at app's installation time.
  - Apps execute faster as code is already compiled.
  - Prevents lags as CPU is not compiling code in background anymore.
  - But consumes more storage space (not really an issue) and a bit more memory.
  - CPU is used less often and should save battery life (a bit).
- Developers may hate that apps take more time to install on a daily basis.
- Early benchmarks show a 10-20% performances bump.

Network Enhancements

Wireless Display

- JB 4.3 introduced Miracast support.
    - Transmits audio/video over HDMI.
- Requirements
    - HW radio chip must be P2P compliant.
    - HW radio chip must support multiple connections at a time.
    - AudioFlinger's policy must provide r_submix remote audio mixing capability.
    - Device must provide HDCP keys as to stream DRM protected content.
- Can be enabled through frameworks/base/core/res/res/values/config.xml:

```
<bool name="config_enableWifiDisplay">true</bool>
```

Bluetooth

- Provides BTLE (Bluetooth Low-Energy) / Bluetooth Smart Ready support.

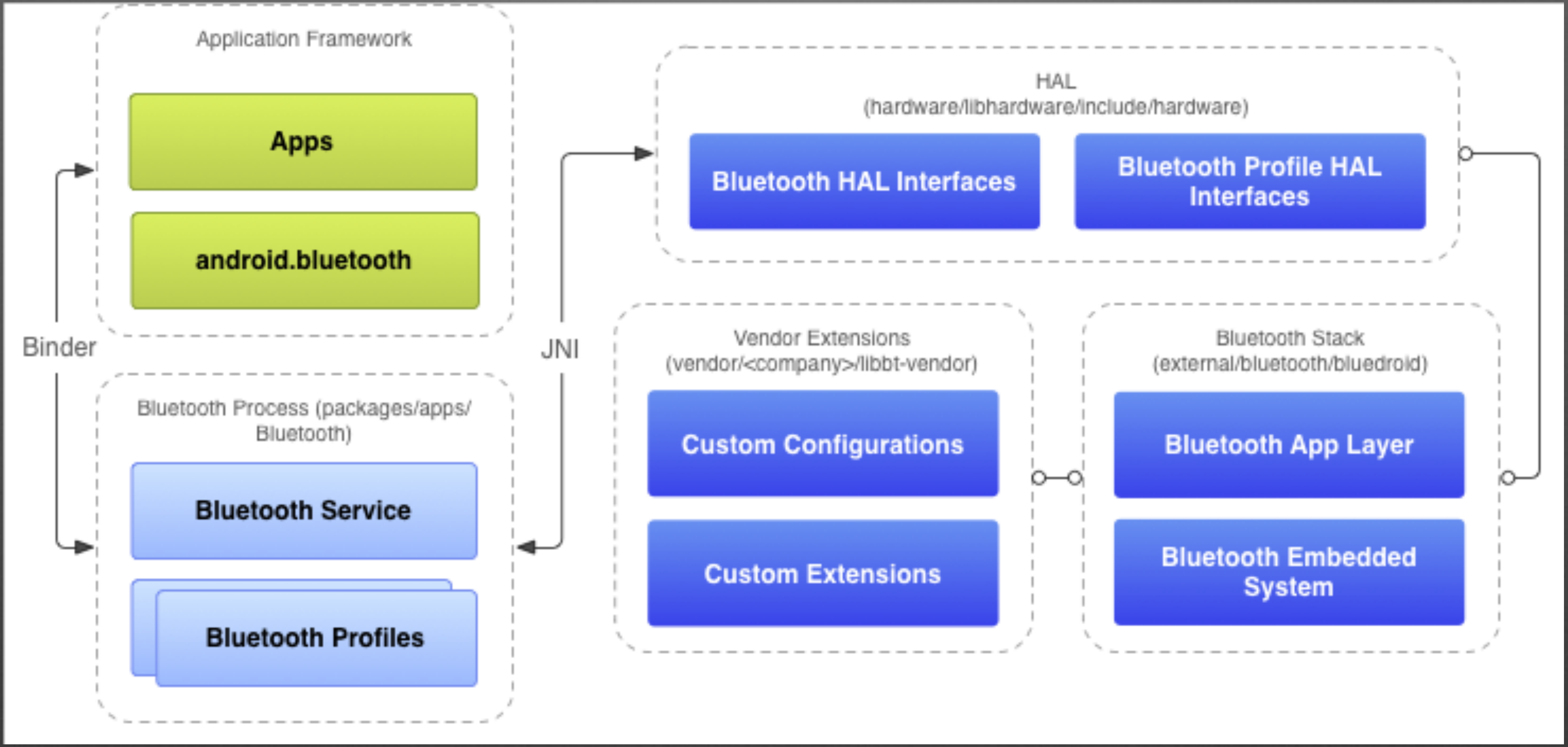
- JB 4.2 replaced BlueZ by Broadcom's BlueDroid (external/bluetooth/bluedroid)
  - Got rid of GPL dependencies (BlueZ and D-BUS).
  - Reference implementation can be customized by every vendor.

- Now features a HAL (like other sub-systems).
  - See hardware/libhardware/include/hardware/bluetooth.h
  - Vendor-specific HCI communication can be done through libbt-vendor plugin.

- BT profiles are implemented through HAL, BlueDroid and called by apps (JNI).
  - See hardware/libhardware/include/hardware/bt_{profile}.h
  - See packages/apps/Bluetooth/jni/com_android_bluetooth_{profile}.cpp

- Now supports the following profiles: A2DP, AVRCP, GATT, HDP, HFP, HID and PAN.

Bluetooth



Application Framework
Apps
android.bluetooth
Binder
Bluetooth Process (packages/apps/Bluetooth)
Bluetooth Service
Bluetooth Profiles
JNI
HAL (hardware/libhardware/include/hardware)
Bluetooth HAL Interfaces
Bluetooth Profile HAL Interfaces
Vendor Extensions (vendor/<company>/libbt-vendor)
Custom Configurations
Custom Extensions
Bluetooth Stack (external/bluetooth/bluedroid)
Bluetooth App Layer
Bluetooth Embedded System

Bluetooth Embedded System (BTE) implements system layer.

Bluetooth Application Layer (BTA) talks with application framework.

Near Field Communication (NFC)

- HAL Update:
  - From NXP's PN544-centric HCI HAL
  - To Broadcom's more generic NFC-NCI HAL.
  - Both continue to exist while new developments should use NFC-NCI one.
  - See hardware/libhardware/include/hardware/nfc.h

- New Broadcom's external/libnfc-nci NFC Controller Interface user stack.

Audio Subsystem

New Features

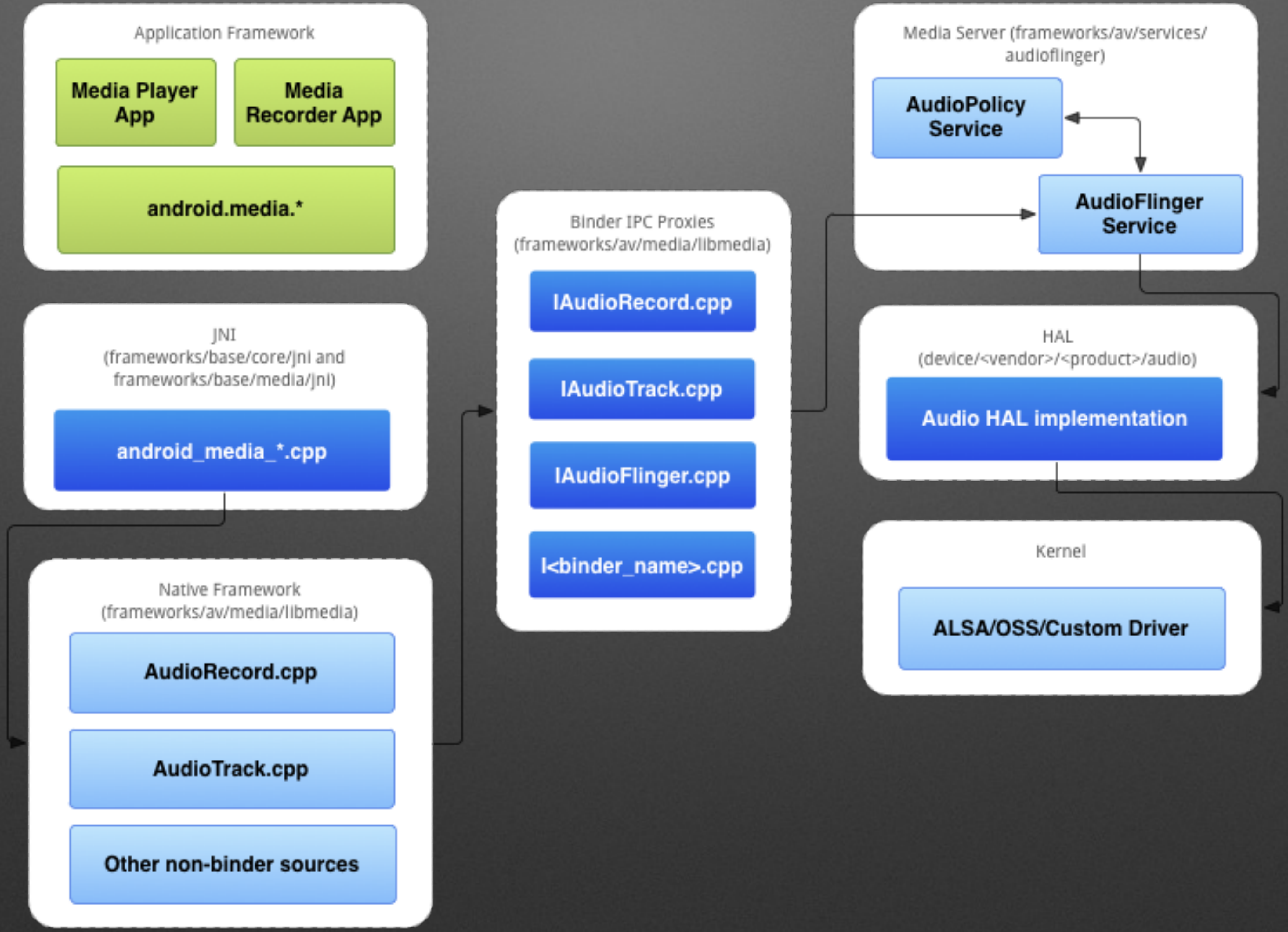
- Multi-channels (e.g. 5.1 surround) support.
- USB devices output.
- Audio streams pre-processing FX.
- Remote audio devices streaming.
- Low Latency enhancements through OpenSL ES APIs.

Supported Devices

- Audio HAL v2.0 now supports multiples input/output types:

- primary: usually SoC's internal sound card.
- a2dp: optional Bluetooth device
  - e.g. headset or speakers.
- usb: optional USB external device
  - e.g. DAC or audio dock.
- r_submix: optional remote interface
  - e.g. HDMI TV over Miracast.
- codec_offload: optional HW audio DSP.

Audio Architecture



Audio Policy - Device Configuration

See /system/etc/audio_policy.conf:

```
global_configuration {
        attached_output_devices
                AUDIO_DEVICE_OUT_EARPIECE |
                AUDIO_DEVICE_OUT_SPEAKER
        default_output_device AUDIO_DEVICE_OUT_SPEAKER
        attached_input_devices
                AUDIO_DEVICE_IN_BUILTIN_MIC |
                AUDIO_DEVICE_IN_BACK_MIC
}
```

Audio Policy - Module Configuration

See /system/etc/audio_policy.conf:

```
audio_hw_modules {
  primary {
    outputs {
      ...
      [X]
        ...
    }
    inputs {
          [Y]
      }
}
```

```
primary {
  sampling_rates 44100|48000
  channel_masks AUDIO_CHANNEL_OUT_STEREO
  formats AUDIO_FORMAT_PCM_16_BIT
  devices AUDIO_DEVICE_OUT_EARPIECE|
             AUDIO_DEVICE_OUT_SPEAKER|
AUDIO_DEVICE_
             OUT_WIRED_HEADSET|
```

```
primary {
     sampling_rates 8000|11025|12000|16000|22050|24000|
32000|44100|48000
     channel_masks AUDIO_CHANNEL_IN_MONO|
AUDIO_CHANNEL_IN_STEREO
     formats AUDIO_FORMAT_PCM_16_BIT
     devices AUDIO_DEVICE_IN_BUILTIN_MIC|
                 AUDIO_DEVICE_IN_WIRED_HEADSET|
```

Audio Policy - HDMI Configuration

- Original raw audio streaming, bypassing AudioFlinger.

- See /system/etc/audio_policy.conf:

```
hdmi {
  sampling_rates 44100|48000
  channel_masks dynamic
  formats AUDIO_FORMAT_PCM_16_BIT
  devices AUDIO_DEVICE_OUT_AUX_DIGITAL
  flags AUDIO_OUTPUT_FLAG_DIRECT
}
```

Automatic HW-detected channels downsizing through "dynamic" option.

Audio Policy - HDMI Miracast Config

```
r_submix {
  outputs {
    submix {
      sampling_rates 44100|48000
      channel_masks AUDIO_CHANNEL_OUT_STEREO
      formats AUDIO_FORMAT_PCM_16_BIT
      devices AUDIO_DEVICE_OUT_REMOTE_SUBMIX
    }
  }
  inputs {
    submix {
      sampling_rates 44100|48000
      channel_masks AUDIO_CHANNEL_IN_STEREO
      formats AUDIO_FORMAT_PCM_16_BIT
      devices AUDIO_DEVICE_IN_REMOTE_SUBMIX
    }
  }
```

Audio Policy - USB Configuration

```
usb {
 outputs {
  usb_accessory {
   sampling_rates 44100
   channel_masks AUDIO_CHANNEL_OUT_STEREO
   formats AUDIO_FORMAT_PCM_16_BIT
   devices AUDIO_DEVICE_OUT_USB_ACCESSORY
  }

  usb_device {
   sampling_rates 44100
   channel_masks AUDIO_CHANNEL_OUT_STEREO
   formats AUDIO_FORMAT_PCM_16_BIT
   devices AUDIO_DEVICE_OUT_USB_DEVICE
  }
 }
}
```

Audio FX

- See /system/etc/audio_effects.conf:

```
pre_processing {
  voice_communication {
     aec {}
     ns {}
  }
  camcorder {
     agc {}
  }
}
```

Enables Accoustic Echo Cancellation (AEC) and Noise Suppression (NS) on

Enables Automatic Gain Control (AGC) on line input.

- Supported inputs: mic, camcorder, voice_recognition and voice_communication.
- Supported Effects: bassboost, virtualizer, equalizer, volume, reverb_env_aux, reverb_env_ins, reverb_pre_aux, reverb_pre_ins, visualizer, downmix, aec, ns, and agc.

Multimedia Subsystem

Features Enhancements

- Low-Level NDK Media Codec Access
    - Provides codec capabilities query (availability, HW/SW implementation ...).
    - GStreamer-like pipeline design capability.
- Media Routing
    - New MediaRouter, MediaRouteActionProvider and MediaRouteButton APIs.
    - Allow streaming to newly-supported remote output devices.
- Media Muxing
    - StageFright now offers API to create TS/PS streams from ES ones.

Features Enhancements

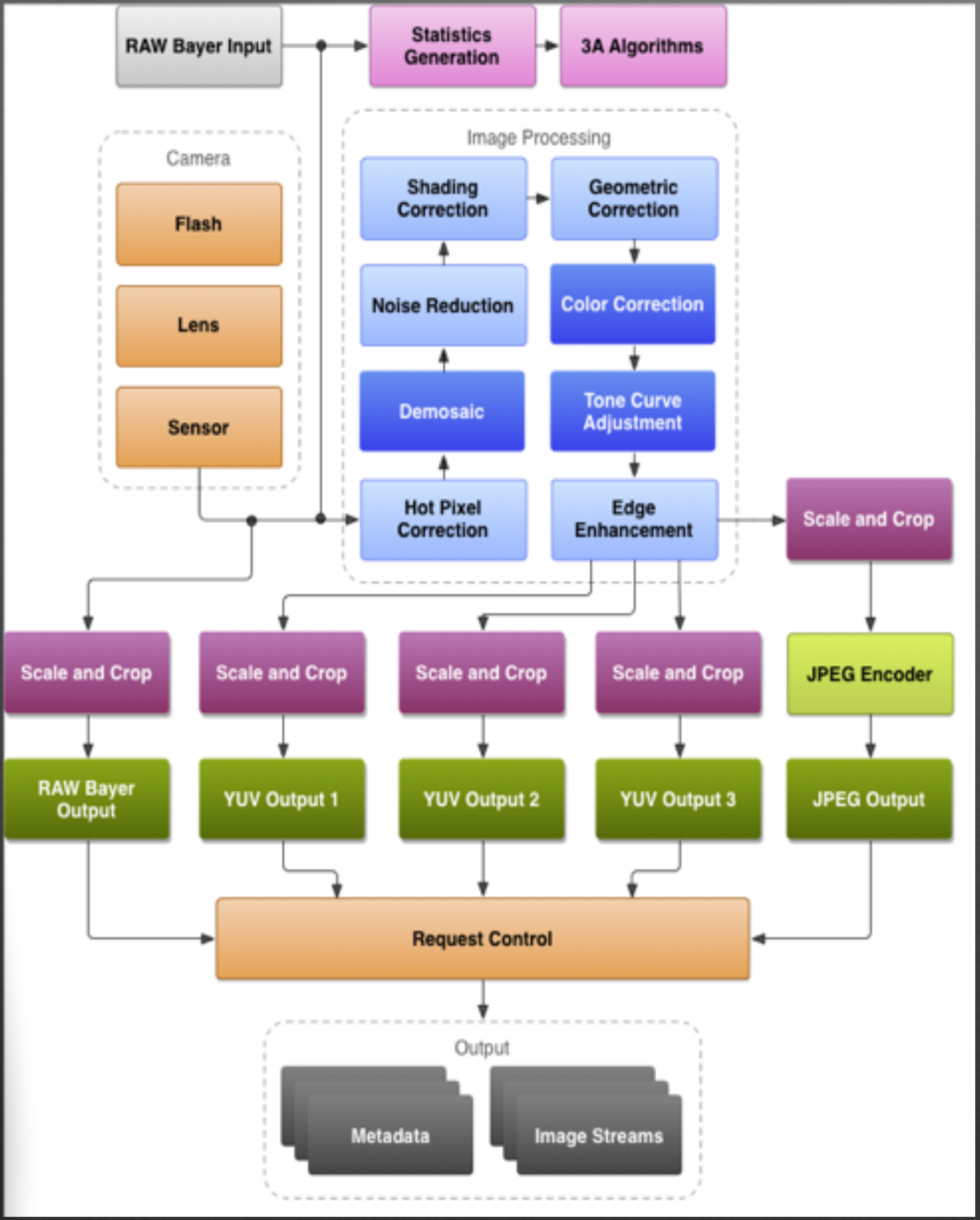
- Media Rights Management
  - Allows apps to add DRM to MPEG DASH over HTTP.
- HW VP8 Encoder
  - Profils/Levels can be configured through NDK's OpenMAX 1.1.2 APIs.
- Video Surface Encoding
  - StageFright now offers 0-copy OpenGL ES video surface encoding (e.g. screen recording).

Camera New Features

- Support for Computational Photography
  - High-Dynamic-Range (HDR) pictures
  - Panoramic pictures
  - Post-Processing FX
    - Blur, noise, image enhancements …

- Requires new camera sensor's metadata to process raw images and deliver the expected final picture.

Image Processing Pipeline



- HAL has to implement image processing pipeline through 3A ISP algorithms:
  - AF: Auto Focus
  - AWB: Automatic White Balance
  - AE: Auto Exposure
- Consumes a RAW YV12 sensor image, applies filters and produces picture.
- Pipeline construction usually is a secret recipe
  - See Qualcomm's SnapDragon 800 Nexus 5 HAL v3 implementation.

Camera API vs. HAL API

ANDROID VERSION	CAMERA_MODULE VERSION	CAMERA_DEVICE (HAL) VERSION	SDK VERSION	FEATURES
4.0	1.0	1.0	14	Face Recognition
4.0.4	1.0	1.0	15	Video Stabilisation
4.1	1.0	1.0	16	AutoFocus Control
4.2	2.0	2.0	17	HDR, Shutter Sound
4.3	2.1	3.0	18	N/A
4.4	2.2	3.1	19	N/A

Camera API v2 Compatibility

To be compliant with Camera Module API v2, device must implement Camera HAL v2 or v3+.

- Multiple HALs
- 1.0 (camera.h)
  - Introduced with ICS as a 1:1 mapping of old C++ CameraHardwareInterface.
  - Perfect compatibility with SDK android.hardware.camera API v1.
- 2.0 (camera2.h)
  - Introduced with JB 4.2 but considered as unstable/deprecated.
  - Extends HAL v1 with manual control capabilities and Zero Shutter Lag support.
  - Requires specific HW sensors.
  - Perfect compatibility with SDK android.hardware.camera API v1 and v2.
- 3.0 (camera3.h)
  - Introduced with JB 4.3 and considered as stable.
  - Complete HAL ABI breakage but offer same hardware requirements than HAL v2.
  - Complete rework of synchronization mechanisms and actions handlers.
  - Perfect compatibility with SDK android.hardware.camera API v1 and v2.
- 3.1 (camera3.h)
  - Introduced with KK 4.4 and considered as stable.
  - Add flush() command to cancel all queued requests (and associated buffers).
  - Perfect compatibility with SDK android.hardware.camera API v1 and v2.

Camera API v2

- Camera HAL v3 is stable but Camera SDK v2 IS NOT !
- Packaged as android.hardware.camera2 (a.k.a "CameraPro").

- Not available to application developers (yet), unless:
  - You extract Kit Kat Java framework:

    ```
    adb pull /system/framework/core.jar .
    adb pull /system/framework/framework.jar .
    ```

  - And convert DEX to JAR:

    ```
    dex2jar core.jar
    dex2jar framework.jar
    ```

  - Then import JARs in Eclipse

    Project > Properties > Java Path >
        Libraries > Add external JARs

External Devices

Multi-Users Support

- Each user now has it own jailed virtual data partition
    - Though large OBB files can be shared in Android/obb directory.
- SD/eMMC user data partition is FAT32 formatted
    - => Try to managed user rights there ;-)

    Dynamic per-user /sdcard mount point

- New "sdcard" daemon (see externals/core/sdcard)
    - FUSE-based FAT emulator that manages files/directories permissions.

- Enabled as a service through init.rc:

```
# virtual sdcard daemon running as media_rw (1023)
service sdcard /system/bin/sdcard /data/media
        /mnt/shell/emulated 1023 1023
class late_start
```

- Vold also supported user data encryption (see init.rc):

```
on fs
        setprop ro.crypto.fuse_sdcard true
```

Multi-Users Support

```
on init
    mkdir /mnt/shell/emulated 0700 shell shell
    mkdir /storage/emulated 0555 root root

    export EXTERNAL_STORAGE /storage/emulated/legacy
    export EMULATED_STORAGE_SOURCE /mnt/shell/emulated
    export EMULATED_STORAGE_TARGET /storage/emulated

    # Support legacy paths
    symlink /storage/emulated/legacy /sdcard
    symlink /storage/emulated/legacy /mnt/sdcard
    symlink /storage/emulated/legacy /storage/sdcard0
    symlink /mnt/shell/emulated/0 /storage/emulated/legacy

on post-fs-data
    mkdir /data/media 0770 media_rw media_rw
```

Sdcard daemon mounts EMULATED_STORAGE_TARGET directory when user connects to Android

Batch Sensors

- Sensors can now deliver events in batches.
  - SoC stays idle instead of being woke up at each sensor's IRQ.
  - Saves battery

- Events can be retrieved in 3 ways:
  - Explicit request at any time.
  - Postponed request at end of batch cycle.
  - Postponed request through cycle's delivery frequency control.
  -

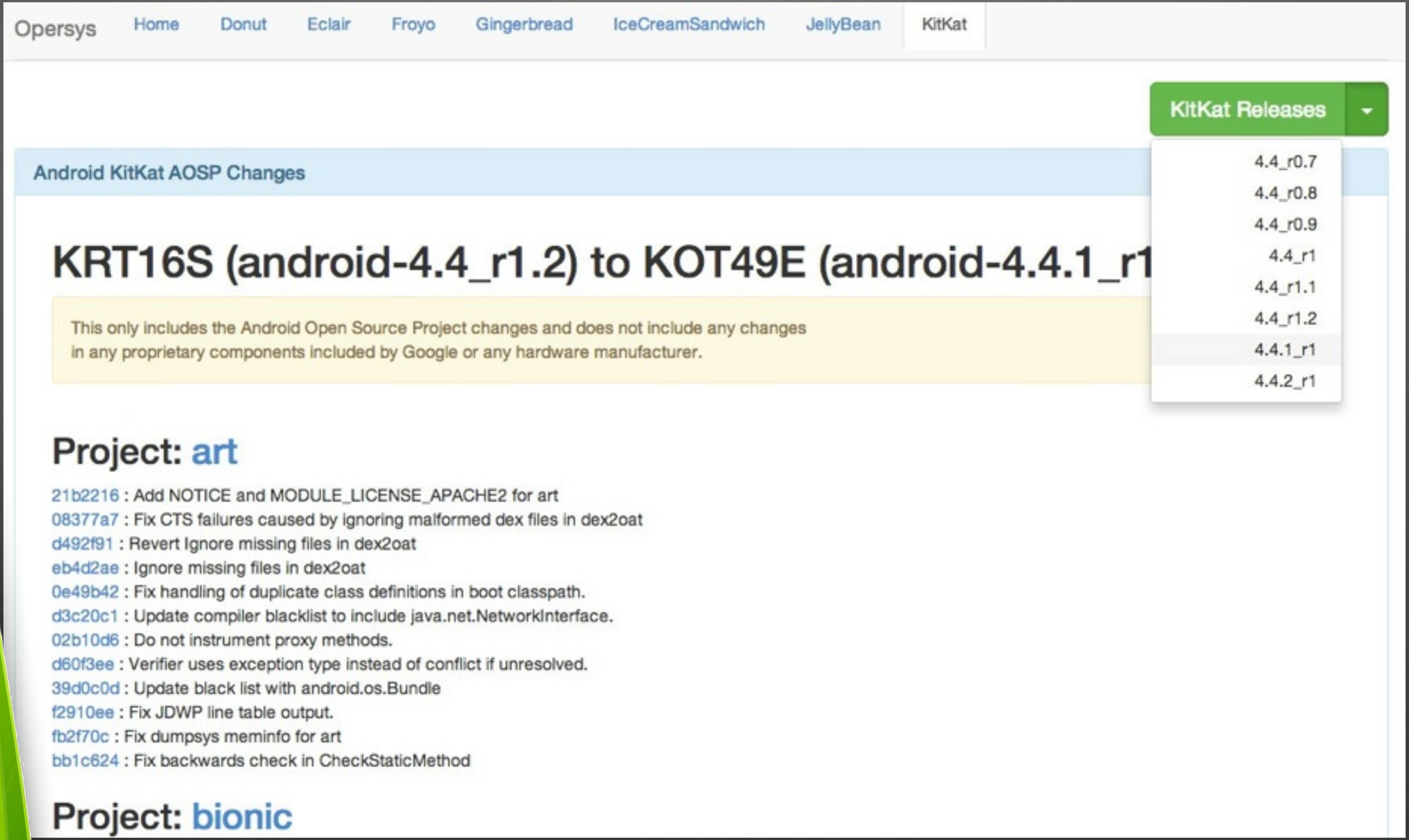
- Sensors HAL has been extended:
  - See hardware/libhardware/include/hardware/sensor.h
  - New batch() feature to be implemented by each driver.

Batch Sensors

- While in batch mode:
  - Reported events are stored.
  - Events are provided all together once an event reaches timeout.
  - Each event features a timestamp allowing apps
    to process all batch-delivered events.

- Batch processing:
  - When SoC's awake, all events are delivered at each period's end
    (when timeout has been reached).
    No event can be lost.
  - When SoC's idle, sensors MUST NOT wake up the CPU.
    Events are then stored in sensor's internal FIFO buffer.
    Events can be lost.
    Only the latest available ones will be delivered to CPU at
    wakeup.

Bonus Time !



Android KitKat AOSP Changes

KRT16S (android-4.4_r1.2) to KOT49E (android-4.4.1_r1

This only includes the Android Open Source Project changes and does not include any changes in any proprietary components included by Google or any hardware manufacturer.

Project: art

21b2216 : Add NOTICE and MODULE_LICENSE_APACHE2 for art
08377a7 : Fix CTS failures caused by ignoring malformed dex files in dex2oat
d492f91 : Revert Ignore missing files in dex2oat
eb4d2ae : Ignore missing files in dex2oat
0e49b42 : Fix handling of duplicate class definitions in boot classpath.
d3c20c1 : Update compiler blacklist to include java.net.NetworkInterface.
02b10d6 : Do not instrument proxy methods.
d60f3ee : Verifier uses exception type instead of conflict if unresolved.
39d0c0d : Update black list with android.os.Bundle
f2910ee : Fix JDWP line table output.
fb2f70c : Fix dumpsys meminfo for art
bb1c624 : Fix backwards check in CheckStaticMethod

Project: bionic

http://aosp.opersys.com/changelog

That's All Folks ...

Benjamin Zores

benjaminzores

@gxben

#Benjamin Zores