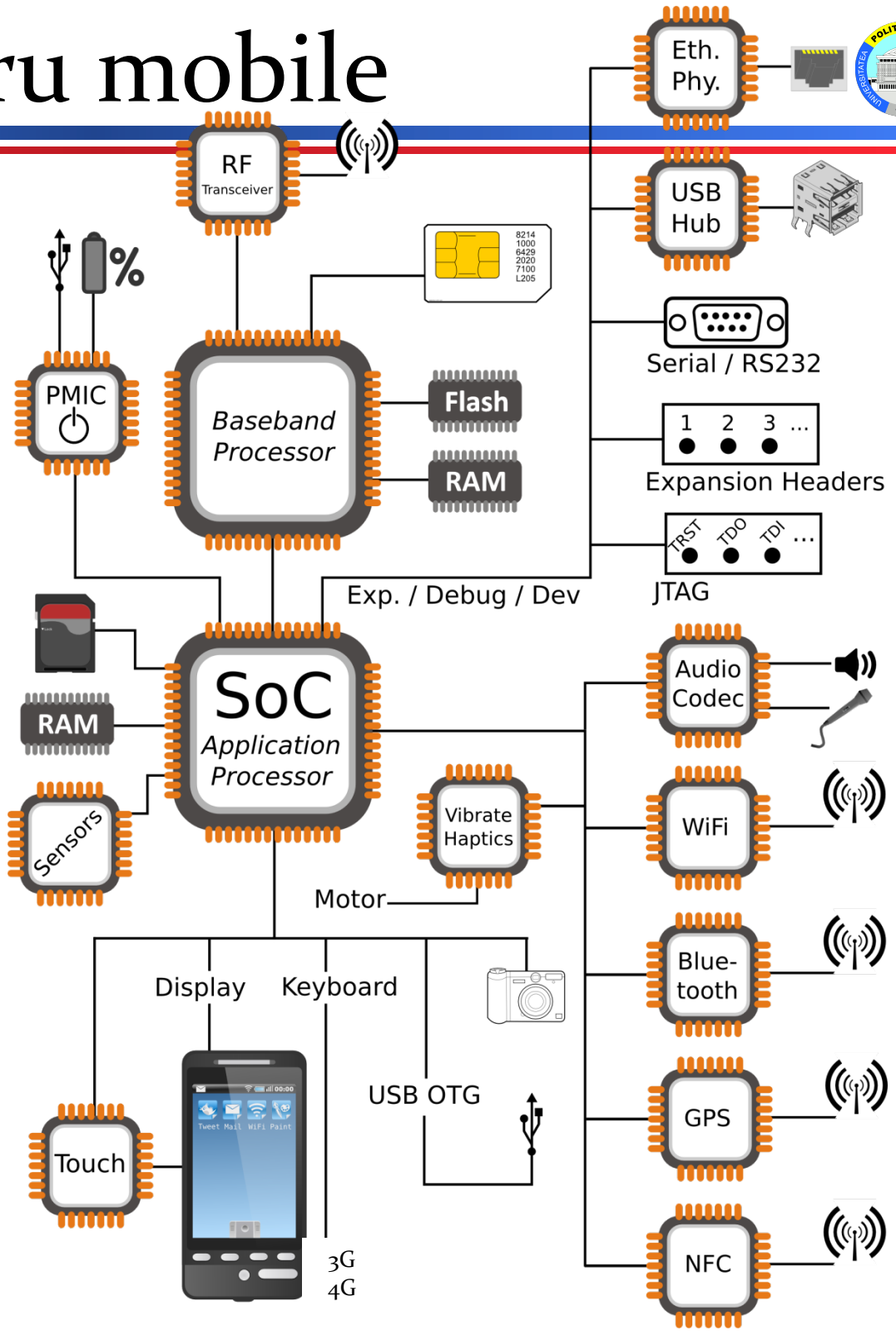
Android Internals

Hardware pentru mobile

- ARM, nu x86
- Memorie limitată
- Baterie mică
- Flash, nu disc
- GUI
- Radio – BT, WiFi, 4G
- GPS



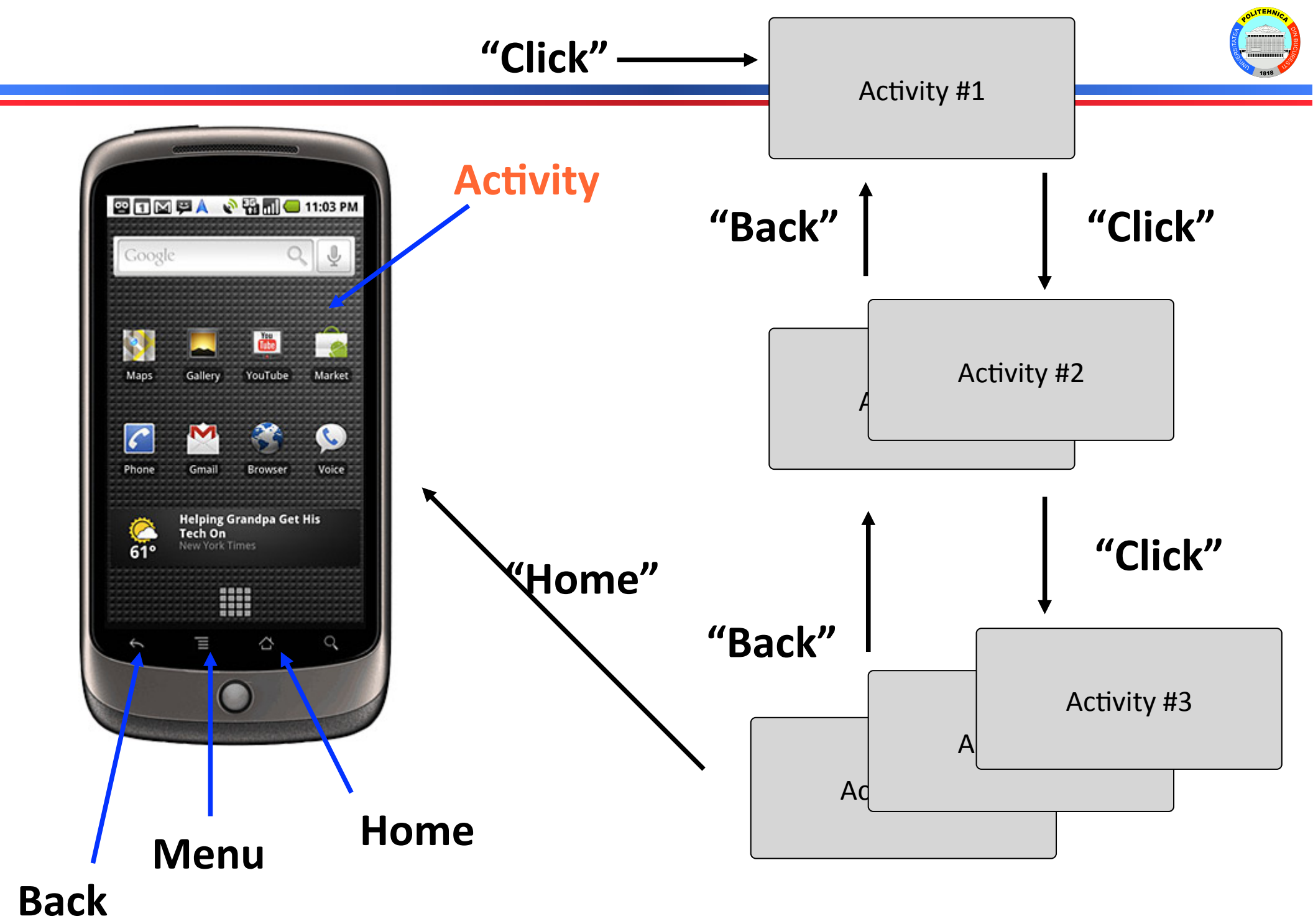
Android

- De ce bazat pe Linux?
  - Drivere
  - Rețea, servicii
  - Gestiune procese, memorie
- Imens
- Se schimbă repede
- Schimbări imprevizibile
  - Schimbări interne, dar API relativ stabile

Android

Au fost adăugate

- Biblioteci C proprii (bionic)
- Mediu de rulare JAVA (dalvik, ART)
- Framework aplicații
- Binder, ashmem
- Gestiune consum
  - wakelocks


"Click"
Activity #1
Activity
"Back"
"Click"
Activity #2
"Home"
"Back"
"Click"
Activity #3
Back
Menu
Home

Comportarea aplicațiilor

- filozofie diferită de aplicații desktop (Unix, Windows)
- fără punct de intrare unic… fără main() !?
- procesele și aplicațiile terminate aleator
  - developerul trebuie să țină cont
- UI deconectat de “creierul” aplicației
- aplicațiile sunt foarte izolate
- **comportarea controlată de**
  - **memorie redusă**
  - **baterie redusă**

Concepte Linux

- Processes (fork() și prietenii)
- Signals (kill() ... or be killed)
- Sockets / Pipes / Fifos / SysV IPC
- Hardware devices ca fisiere (/dev)
- Daemons
- Shell / scripts
- Useri (root vs. ceilalți -- # vs. $)
- ELF files
- GNU toolchain
- ... 40 de ani de Unix

Concepte Android

- Components
- Intents
- Manifest file
- Component lifecycle
- Processes and threads
- Remote procedure calls
- Permissions
- Storage
- Native development

Componente

- 1 Aplicație = N Componente
- aplicațiile pot partaja componente
- procesele aplicațiilor sunt pornite automat când o componentă e apelată
- **N puncte de intrare, !1, !main()**
- Componente:
  - Activities
  - Services
  - Broadcast Receivers
  - Content Providers

Intents

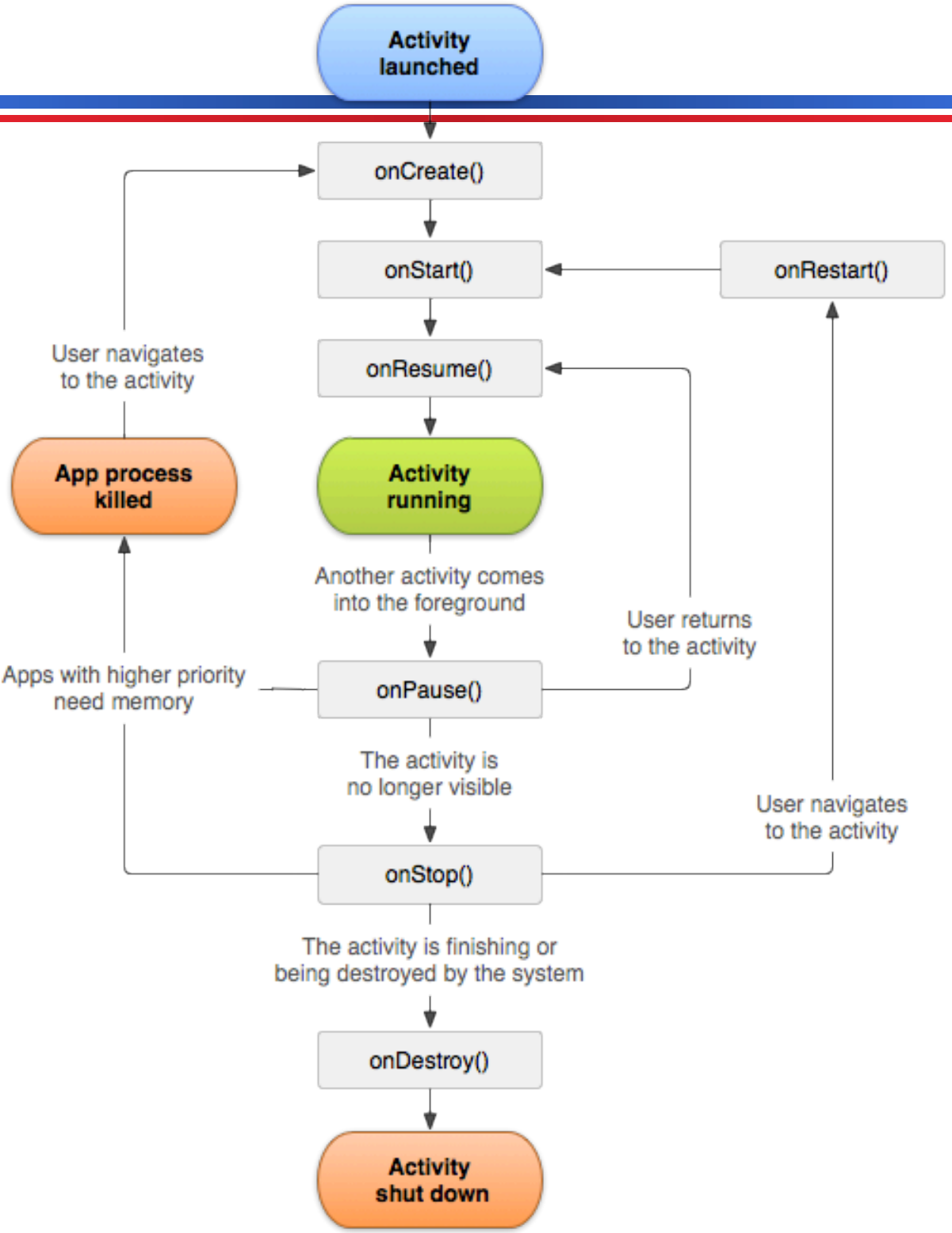
- Intent = mesaj asincron cu/fără target
- Ca un signal polimorfic de Unix signal, fără target
- Intent Filters specificate în Manifest file

Manifest file

- Informează sistemul despre componentele aplicației
- numit mereu AndroidManifest.xml
- Activity = <activity> ... static
- Service = <service> ... static
- Broadcast Receiver:
  - Static = <receiver>
  - Dynamic = Context.registerReceiver()
- Content Provider = <provider> ... static

Ciclul de viață al componentelor

- Sistem pornește/oprește/omoară procesele automat
- **Toată comportarea sistemului ține cont de**
  - **Memorie redusă**
  - **Baterie redusă**
- Sistemul
  - apelează callback-uri când e necesar
  - gestionează ciclul de viață
- unele comopnente sunt mai complex de gestionat


Activity launched
onCreate()
onStart()
onRestart()
onResume()
User navigates to the activity
App process killed
Activity running
Another activity comes into the foreground
User returns to the activity
Apps with higher priority need memory
onPause()
The activity is no longer visible
User navigates to the activity
onStop()
The activity is finishing or being destroyed by the system
onDestroy()
Activity shut down

Procese și threaduri

- **Procese**
  - default: ∀callback către ∀ componentă -> main thread
  - nu se fac operații blocante în thread-ul principal:
    - threaduri suplimentare
  - terminare/restartare proces – la discreția sistemului
  - **sistemul gestionează ciclul de viață**

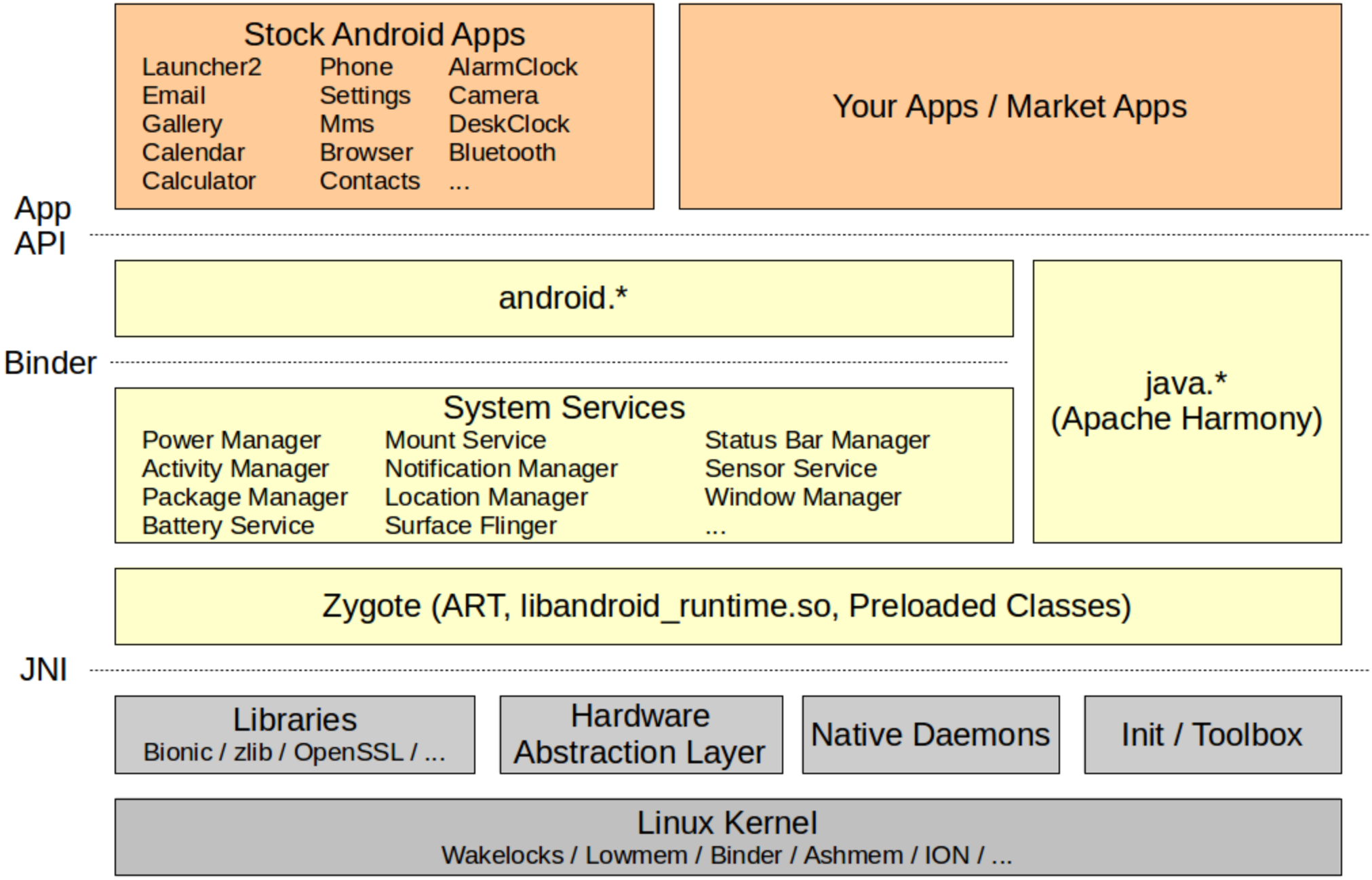
- **Threaduri**
  - create cu obiectul Java Thread
  - nu GUI în thread-ul de networking
  - nu networking în thread-ul de GUI

Remote procedure calls

- Nu există IPC de UNIX (semafoare , cozi de mesaje, pipe)
  - dificilă portarea programelor UNIX
- Android RPC = Binder
- Binder in kernelul de Linux 3.19 (2015)

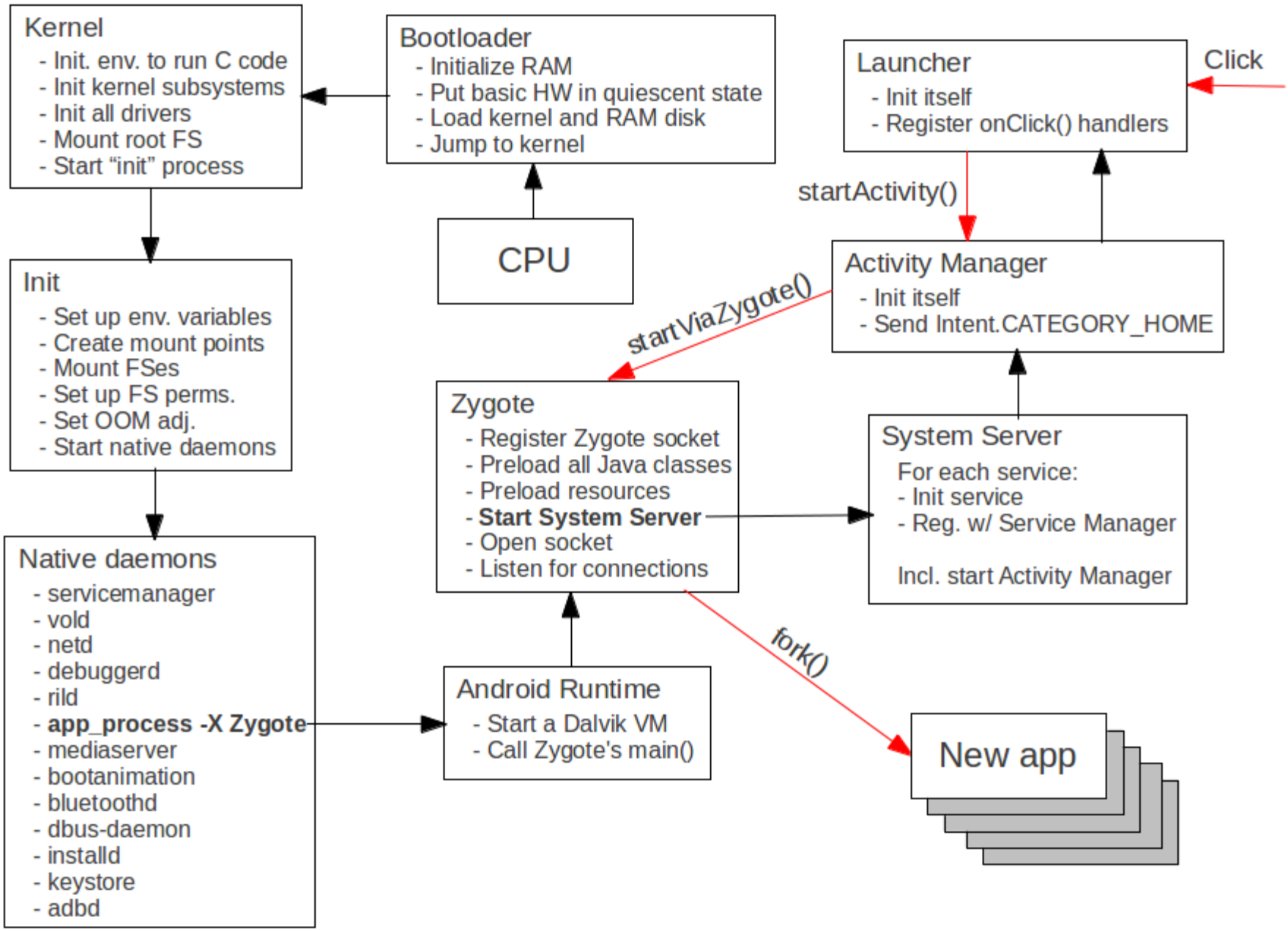
Securitate/Permisiuni

- Securitate la nivel de proces: UID, GID
- Permisiunile implementează restricții la:
  - Operații per-process
  - Acces per-URI
- Aplicațiile rulate în sandbox
- Permisiuni speciale pentru a ieși din sandbox
- Se dă accesul bazat pe:
  - Certificate
  - User prompt
- Toate permisiunile declarate static


Stock Android Apps
Launcher2
Email
Gallery
Calendar
Calculator
Phone
Settings
Mms
Browser
Contacts
AlarmClock
Camera
DeskClock
Bluetooth
...
Your Apps / Market Apps
App API
android.*
java.* (Apache Harmony)
Binder
System Services
Power Manager
Activity Manager
Package Manager
Battery Service
Mount Service
Notification Manager
Location Manager
Surface Flinger
Status Bar Manager
Sensor Service
Window Manager
...
Zygote (ART, libandroid_runtime.so, Preloaded Classes)
JNI
Libraries
Bionic / zlib / OpenSSL / ...
Hardware Abstraction Layer
Native Daemons
Init / Toolbox
Linux Kernel
Wakelocks / Lowmem / Binder / Ashmem / ION / ...

Secvența de startup

- Bootloader
- Kernel
- Init
- Zygote
- System Server
- Activity Manager
- Launcher (Home)


Kernel
- Init. env. to run C code
- Init kernel subsystems
- Init all drivers
- Mount root FS
- Start “init” process
Bootloader
- Initialize RAM
- Put basic HW in quiescent state
- Load kernel and RAM disk
- Jump to kernel
Launcher
- Init itself
- Register onClick() handlers
Click
CPU
startActivity()
Activity Manager
- Init itself
- Send Intent.CATEGORY_HOME
Init
- Set up env. variables
- Create mount points
- Mount FSes
- Set up FS perms.
- Set OOM adj.
- Start native daemons
startViaZygote()
Zygote
- Register Zygote socket
- Preload all Java classes
- Preload resources
- Start System Server
- Open socket
- Listen for connections
System Server
For each service:
- Init service
- Reg. w/ Service Manager
Incl. start Activity Manager
Native daemons
- servicemanager
- vold
- netd
- debuggerd
- rild
- app_process -X Zygote
- mediaserver
- bootanimation
- bluetoothd
- dbus-daemon
- installd
- keystore
- adbd
Android Runtime
- Start a Dalvik VM
- Call Zygote's main()
fork()
New app

Zygote

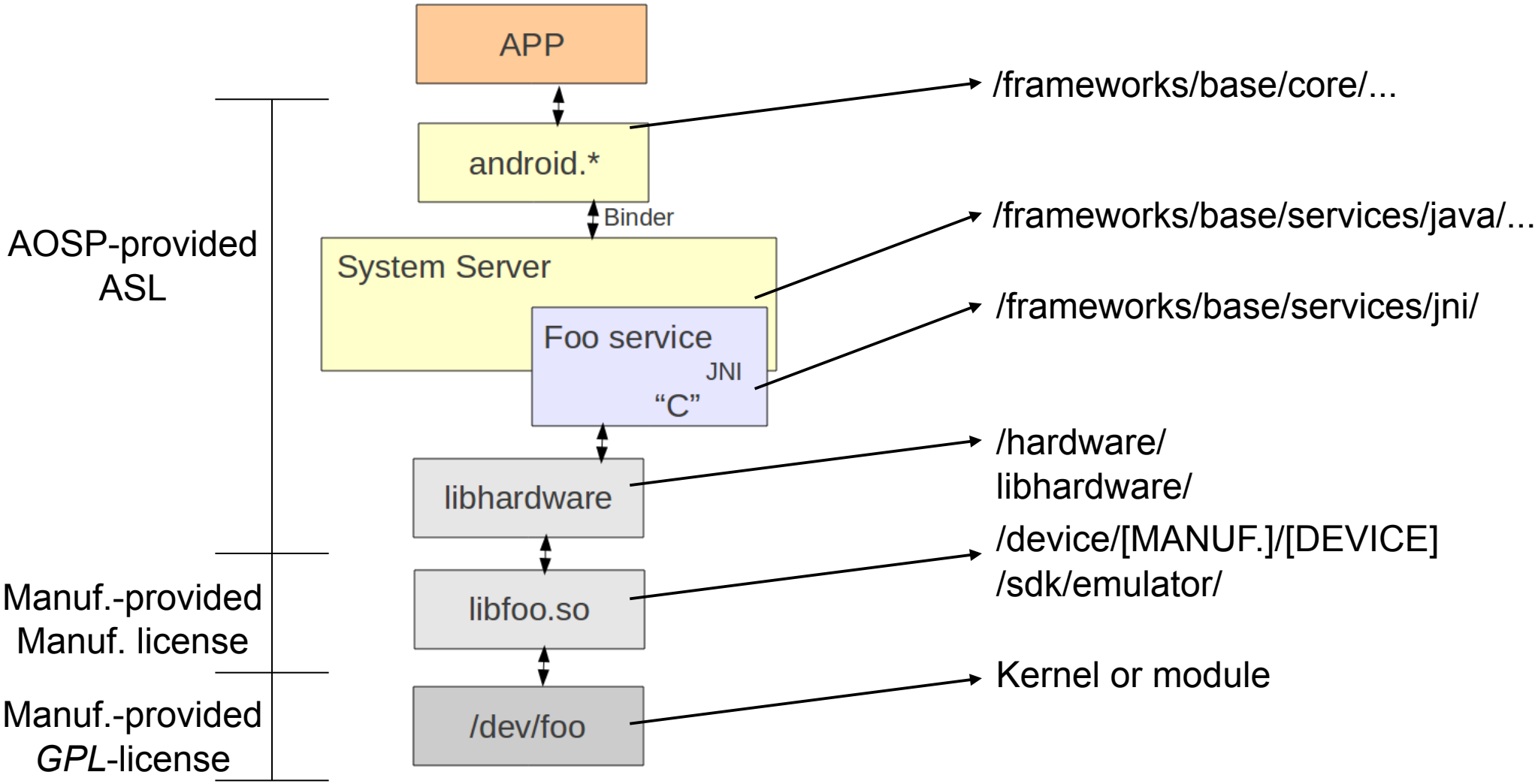
- Pornește după startup-ul tipic de Linux
- Are toate bibliotecile încărcate
- Conține masina virtuala Java
  - Dalvik < 4
  - ART (Android RunTime) >= 4
- fiecare proces nou este fork de zygote

Hardware Support

- hardware not accessed directly
- HAL = hardware abstraction layer
- HAL “modules” are .so files

- Activity Recognition
- Audio
- Bluetooth
- Camera
- ConsumerIr
- Framebuffer
- Fingerprint
- Fused Location
- GPS
- Gralloc
- HWcomposer
- Keymaster
- Lights
- NFS
- Power
- Sensors

Hardware Abstraction Layer



User-Space Nativ

- Principale:
  - /data => User data
  - /system => System components
  - /cache => Cache (& OTA update)
- Sistem Linux
  - /dev
  - /proc
  - /sys
  - /sbin
  - /mnt
- Multe directoare Linux lipsesc ...

Java din Android

- Oracle (Sun)
  - Java = Java language + JVM + JDK libs
- Android < 5 (Lollipop)
  - Java = Java language + Dalvik
- Android < 7 (Nougat)
  - Java = Java language + ART + Apache Harmony
- Android >= 7
  - Java = Java language + ART + OpenJDK

Mașina virtuală java

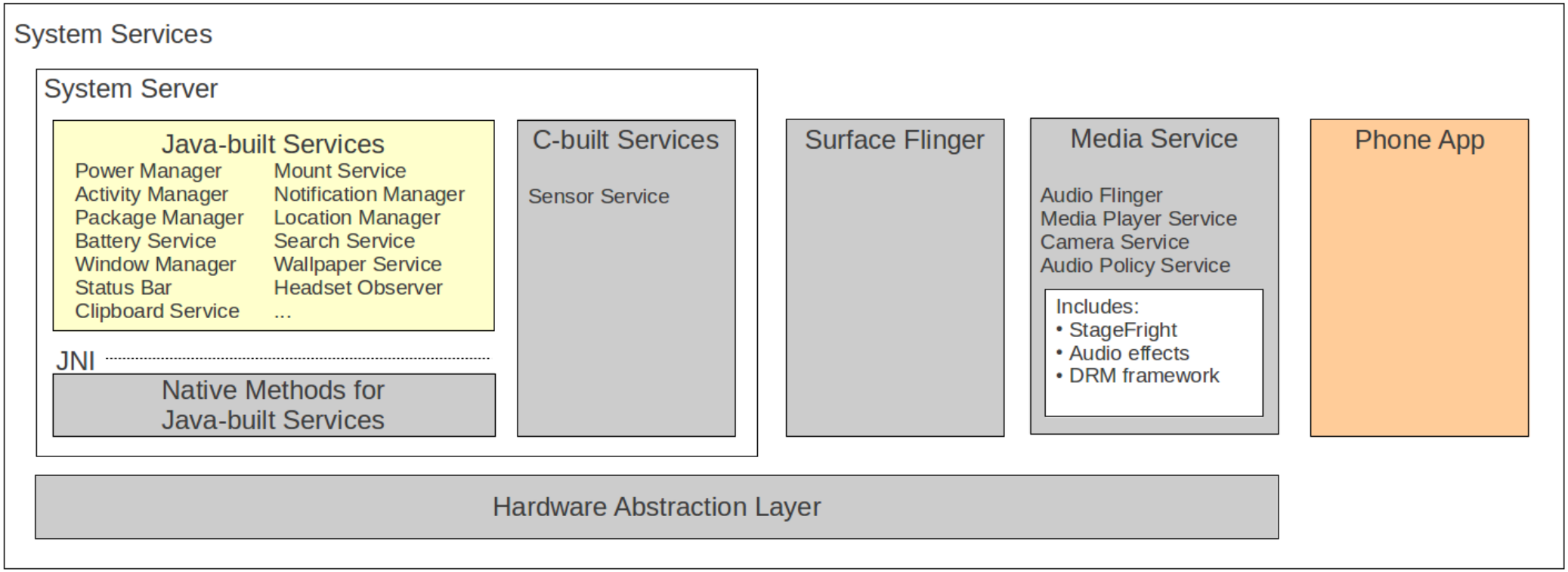
- Aplicațiile Android rulează propriul proces
  - Mașina virtuală java

- ART = Android RunTime
  - 64 bit
  - Multi-core
  - AOT instead of JIT

- Dalvik

Servicii de sistem

- Sistemul Android
- ~100 servicii (Lollipop)
- 5 - 6 servicii noi la fiecare release
- adb shell “service list” pentru listare
  - WifiService
  - ActivityManagerService
  - PowerService
  - PackageManagerService
  - LocationManagerService


System Services
System Server
Java-built Services
Power Manager
Activity Manager
Package Manager
Battery Service
Window Manager
Status Bar
Clipboard Service
Mount Service
Notification Manager
Location Manager
Search Service
Wallpaper Service
Headset Observer
...
JNI
Native Methods for Java-built Services
C-built Services
Sensor Service
Surface Flinger
Media Service
Audio Flinger
Media Player Service
Camera Service
Audio Policy Service
Includes:
• StageFright
• Audio effects
• DRM framework
Phone App
Hardware Abstraction Layer

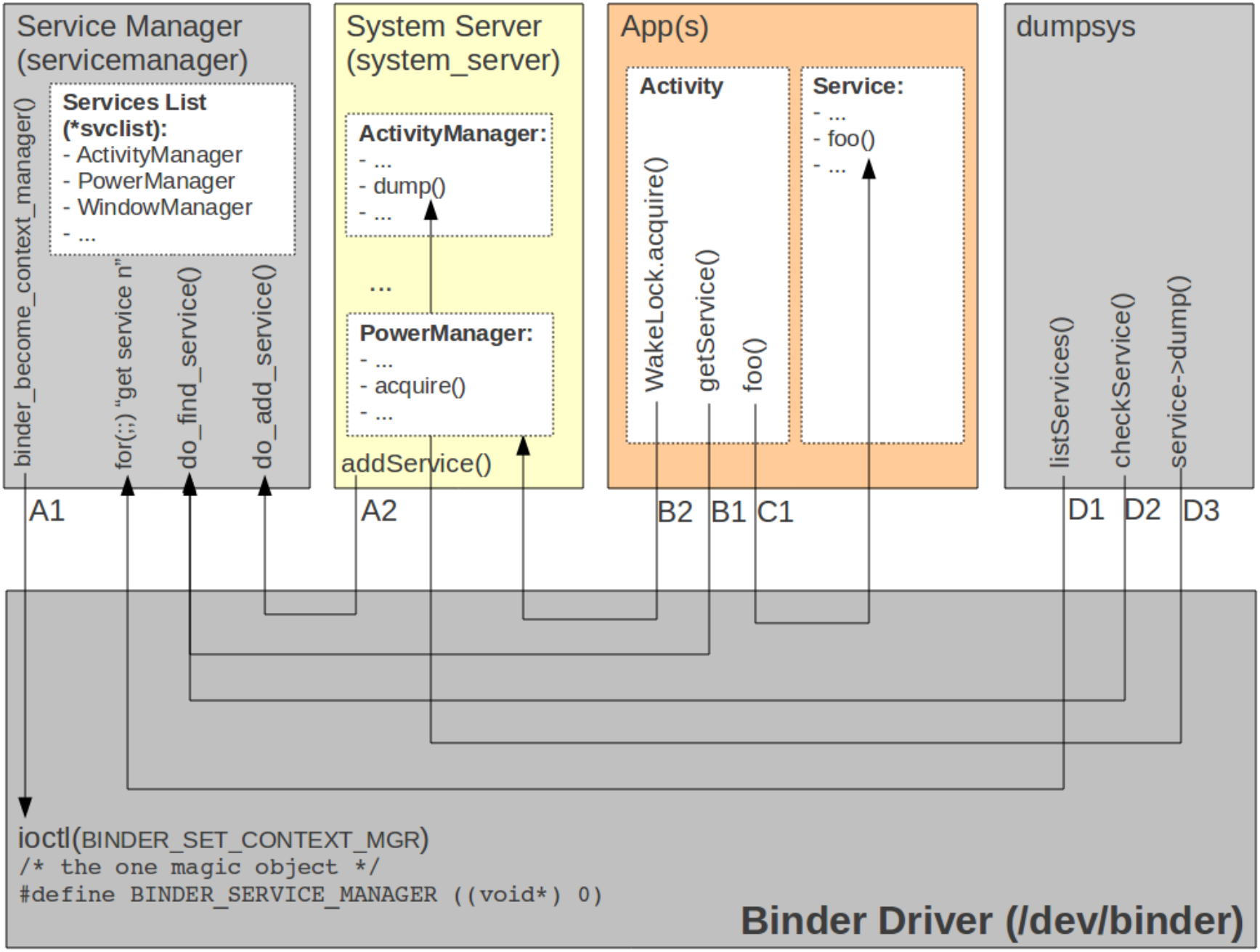
ActivityManager

- Pornește Activities, Services
- “găsește” Content Providers
- Intent broadcasting
- Application Not Responding
- Permisiuni
- Task management
- Lifecycle management

- **Exemplu:**
- Startăm app din Launcher:
- onClick(Launcher)
- startActivity(Activity.java)
- *<Binder>*
- ActivityManagerService
- startViaZygote(Process.java)
- *<Socket>*
- Zygote

Binder

- Datele trimise prin “parcels”, folosind “transactions”
- implementat în kernel
- /dev/binder
- /sys/kerne/debug/binder/*
- android.* API se conectează la System Server prin binder


Service Manager
(servicemanager)
Services List
(*svclist):
- ActivityManager
- PowerManager
- WindowManager
- ...
binder_become_context_manager()
for(;;) "get service n"
do_find_service()
do_add_service()
System Server
(system_server)
ActivityManager:
- ...
- dump()
- ...
...
PowerManager:
- ...
- acquire()
- ...
addService()
App(s)
Activity
WakeLock.acquire()
getService()
foo()
Service:
- ...
- foo()
- ...
dumpsys
listServices()
checkService()
service->dump()
A1
A2
B2
B1
C1
D1
D2
D3
ioctl(BINDER_SET_CONTEXT_MGR)
/* the one magic object */
#define BINDER_SERVICE_MANAGER ((void*) 0)
Binder Driver (/dev/binder)

Gestiunea bateriei

- Bazat pe “Suspend to RAM” din Linux
- module PowerManager
  - Display, backlight, keyboard
  - Monitorizează bateria
  - Coordonează circuitul de încarcare
  - Necesită permisiunea WAKE_LOCK
- Wakelocks

Flag	CPU	Screen	Keyboard
PARTIAL_WAKE_LOCK	On	Off	Off
SCREEN_DIM_WAKE_LOCK	On	Dim	Off
SCREEN_BRIGHT_WAKE_LOCK	On	Bright	Off
FULL_WAKE_LOCK	On	Bright	Bright