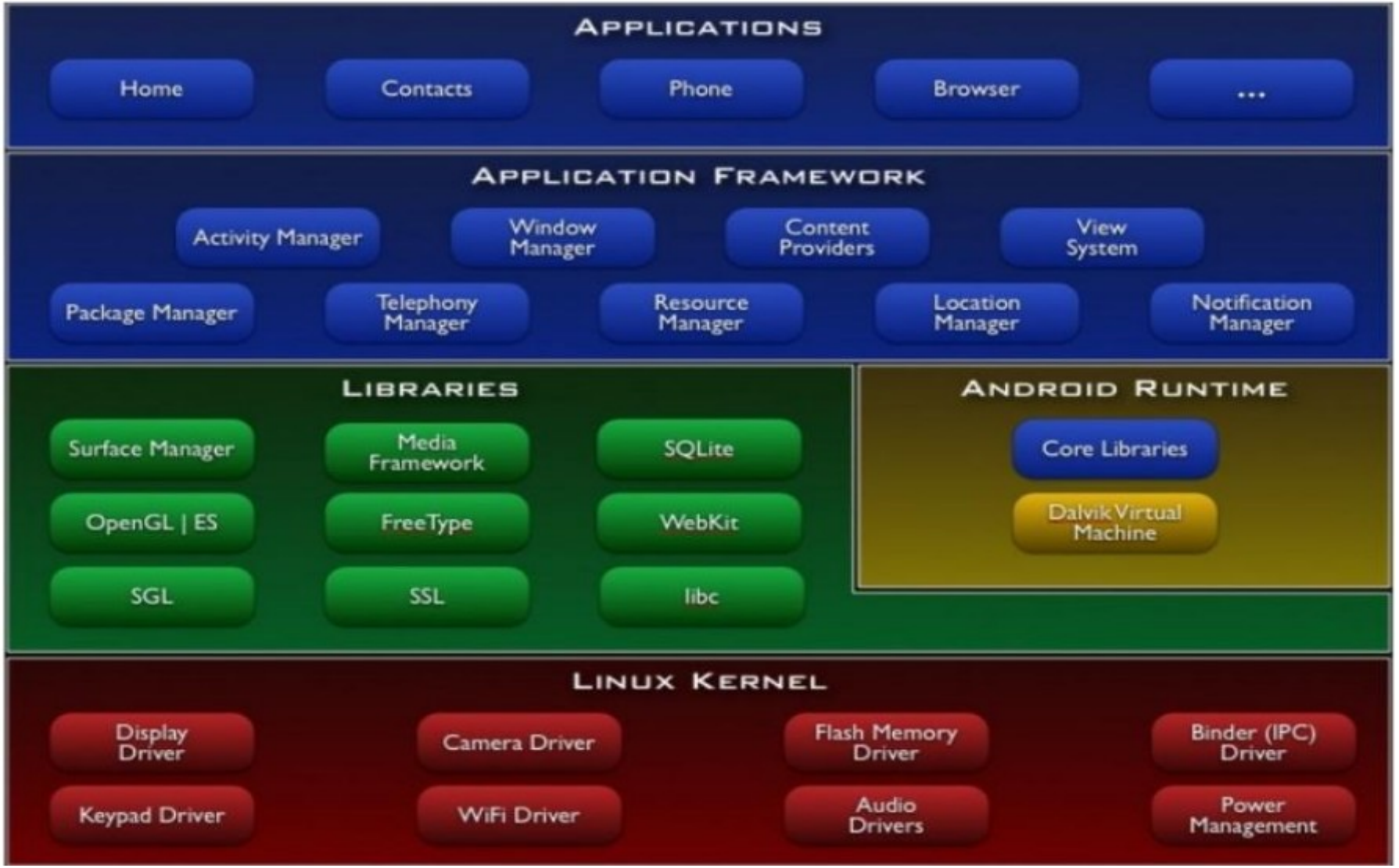
Android Framework and HAL

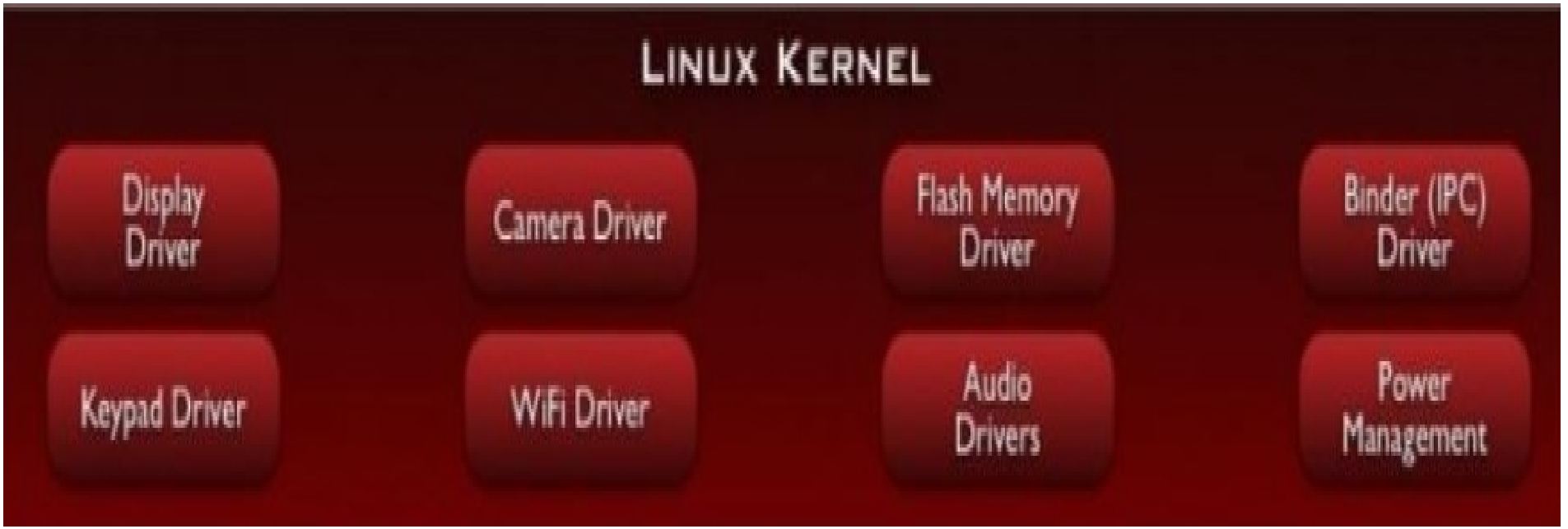
Agenda

- Android Stack
- Why HAL Layer is required?
- How to add System Service in Android?
- Flow of Android code from Application to Kernel
- Power Control Example in Android

Android Stack



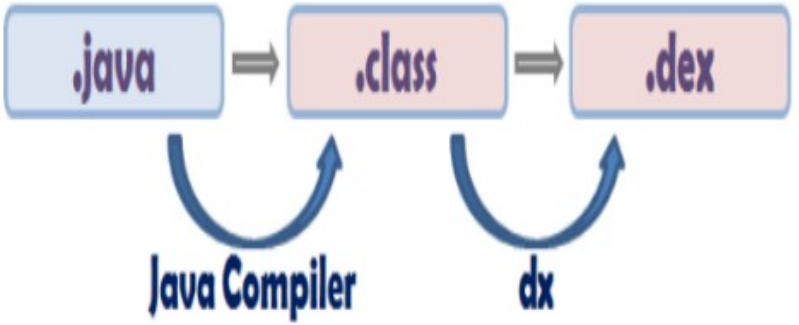
Linux Kernel



Libraries

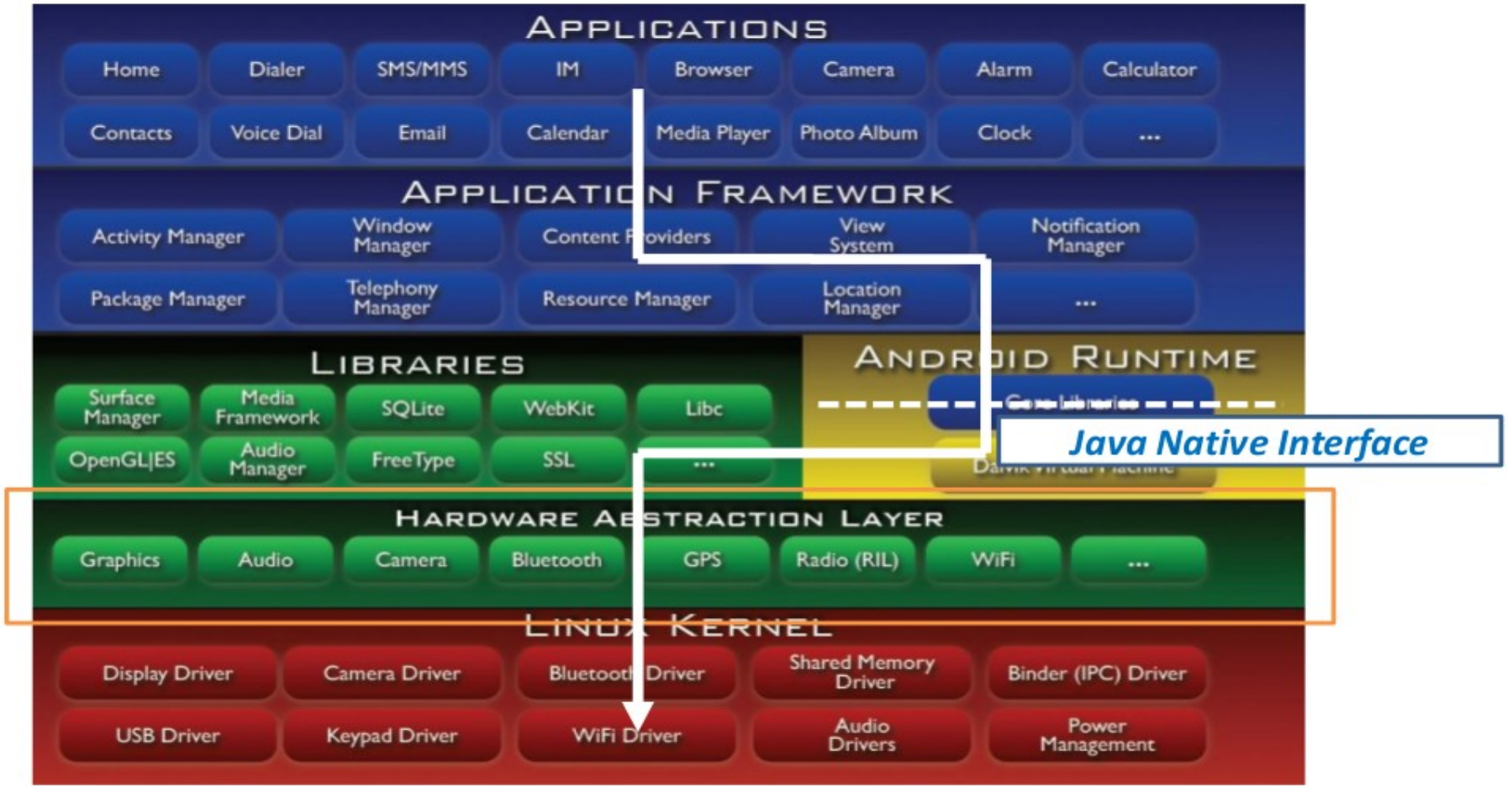

LIBRARIES
Surface Manager
Media Framework
SQLite
WebKit
Libc
OpenGL|ES
Audio Manager
FreeType
SSL
...

Android Runtime



Application Framework

Hardware Abstraction Layer



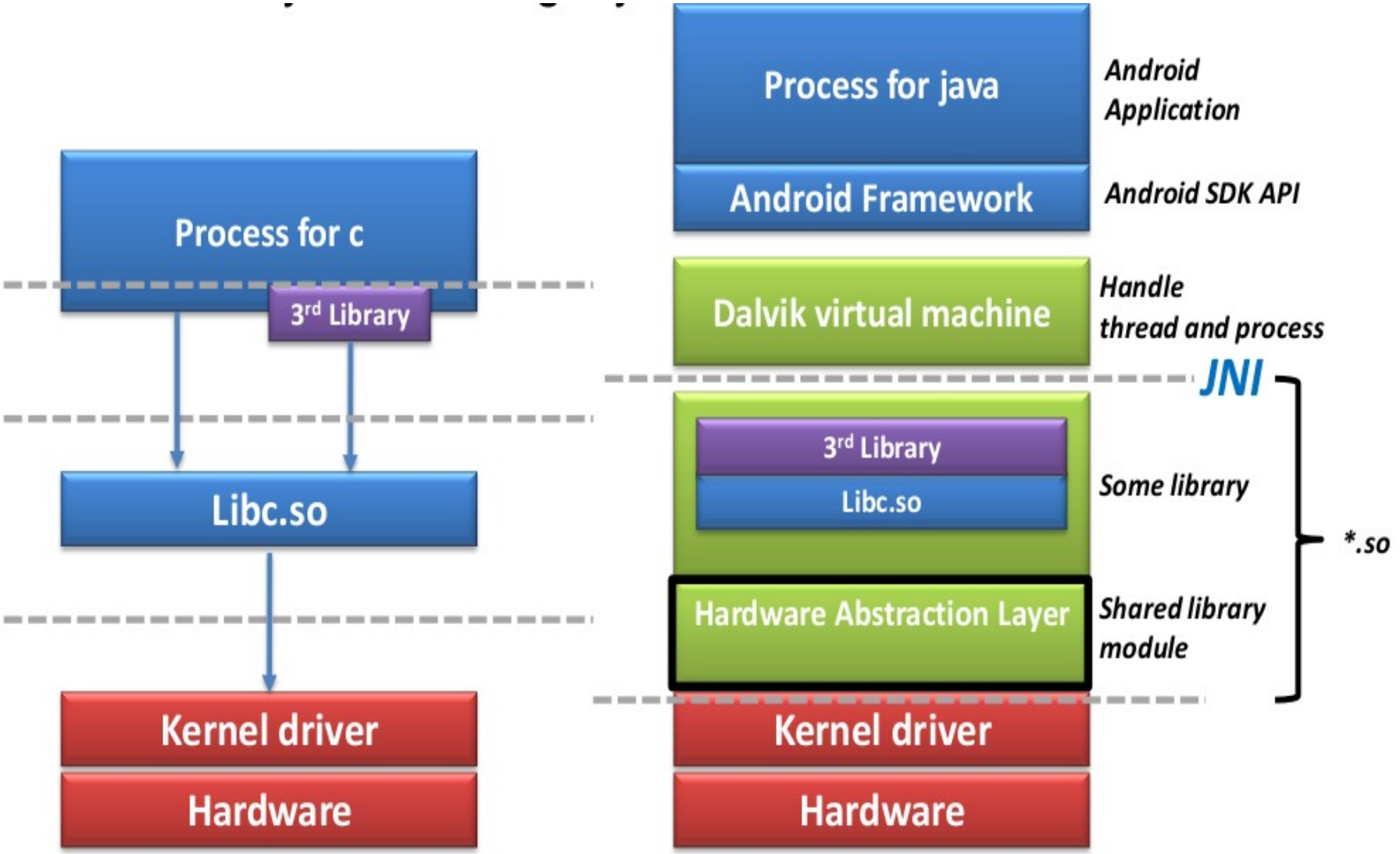
HAL

- Two general categories of HAL Module
- Explicitly Loaded modules(runtime using dlopen())

  <aosp>/hardware/libhardware/include/hardware
- Automatically loaded modules by dynamic linker

  <aosp>/hardware/libhardware_legacy/include/hardware_legacy

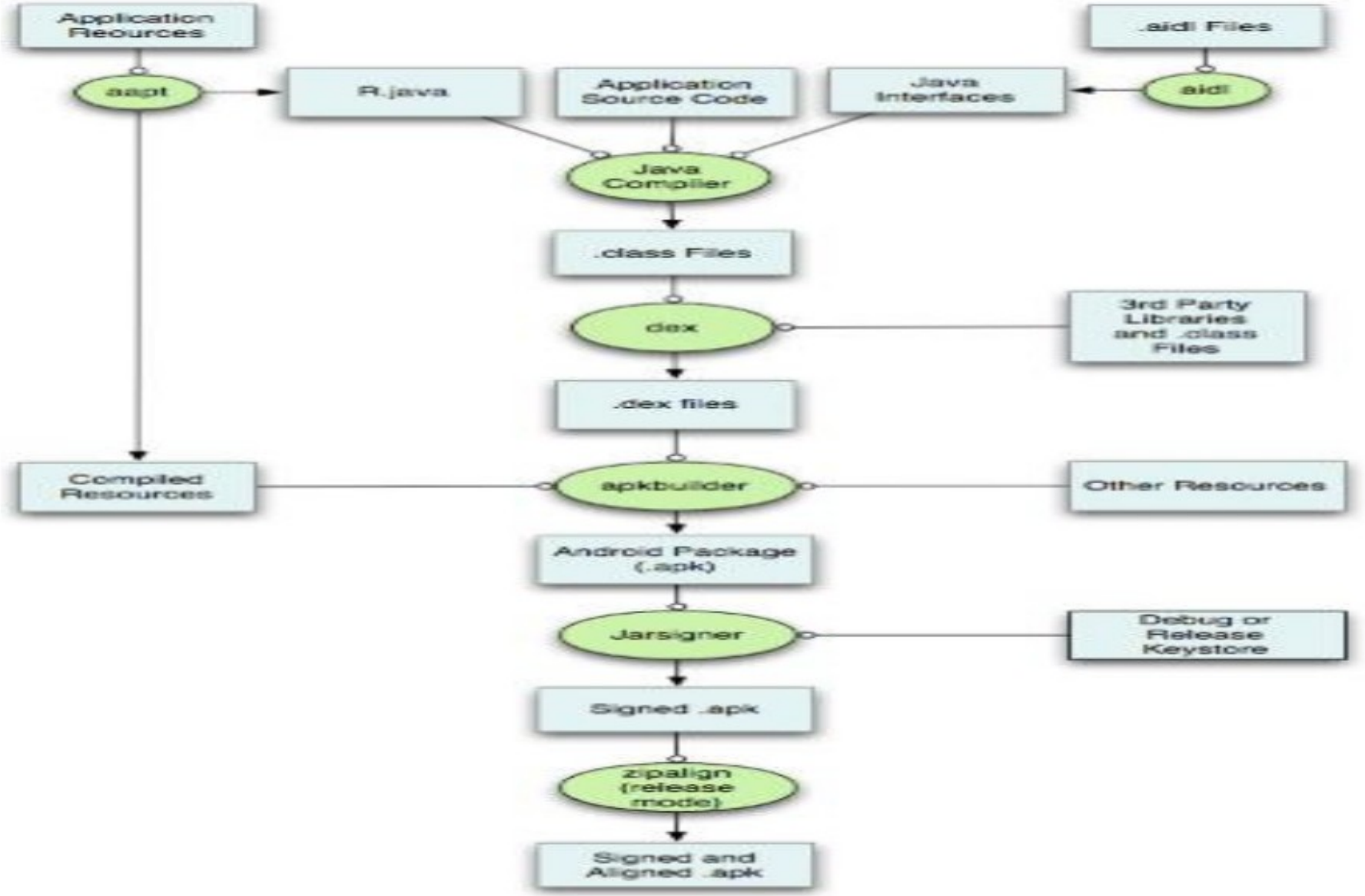
General Linux Vs Android System for legacy



Android Layer Analysis

Layer	Language	Form	Ship
Application	JAVA	*.apk	System.img
Framework	JAVA	*.jar	System.img
Libraries	C/C++	*.so	System.img
HAL	C/C++	*.so	System.img
Kernel	C/asm	*.ko	Boot.img

Building an Android Application



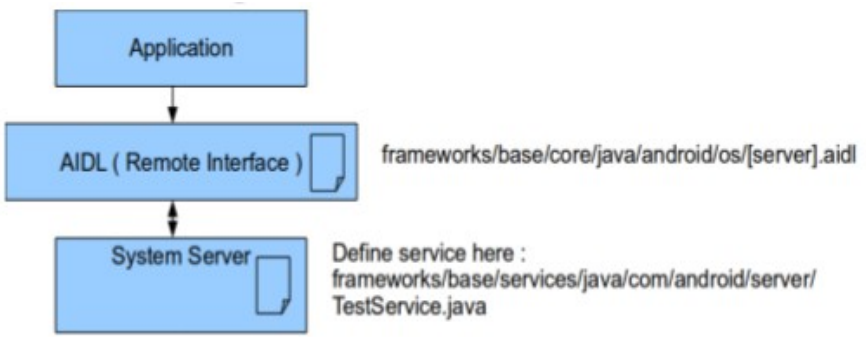
Example: Power Control in Application

```
MButton.setOnClickListner(new Button.OnClickListener()
{
public void onClick(View v)
    {
        mPowerManager=(PowerManager)getSystemService(Context.POWER_SERVICE);
        mPowerManager.goToSleep(1000);
    }
});
```

Power Control in Android framework
Android/frameworks/base/core/java/android/os/PowerManager.java

```
public void goToSleep(long time)
    {
        try {
            mService.goToSleep(time);
        } catch (RemoteException e) {
        }
    }
```

Adding a System service



- Create service
- Register service
- Expose service
- Add [service].aidl for build : open frameworks/base/Android.mk and add line for .aidl file
- Rebuild the framework/base or android system. Service is now ready to use by other application/process
- To use service

  first get service handle using "ServiceManager.getService()" api

  use service handle to call set of functions exposed by service

Power Control In android runtime service
Android/frameworks/base/services/java/com/android/server/PowerManagerService.java
Android/frameworks/base/core/java/android/os/Power.java

Trace the code

- **goToSleep ()**-> goToSleepWithReason() -> goToSleepLocked() -> setPowerState() ->

  updateNativePowerStateLocked();

  setScreenStateLocked(boolean on)->setScreenState(boolean on);

- private void updateNativePowerStateLocked() {

  nativeSetPowerState(

  (mPowerState & SCREEN_ON_BIT) != 0,

  (mPowerState & SCREEN_BRIGHT) == SCREEN_BRIGHT);

  }

- **private native void nativeSetPowerState(boolean screenOn, boolean screenBright);**
- **public static native int setScreenState(boolean on);**

Register Service
Android/frameworks/base/services/java/com/android/server/SystemServer.java

```
Try {
        Slog.i(TAG, "Power Manager");
        power = new PowerManagerService();
        ServiceManager.addService(Context.POWER_SERVICE, power);
}Catch(Exception e){
}
```

Expose Service
Android/frameworks/base/core/java/android/os/IPowerManager.aidl

```
/** @hide */
interface IPowerManager
{
        void goToSleep(long time);
        void goToSleepWithReason(long time, int reason);
}
```

Add .aidl for build :Android/frameworks/base/Android.mk

LOCAL_SRC_FILES += \

**core/java/android/os/IPowerManager.aidl **

core/java/android/os/IMusicAppService.aidl \

core/java/android/os/IRemoteCallback.aidl \

core/java/android/os/IVibratorService.aidl

Power Control In JNI
Android/frameworks/base/services/services/jni/com_android_server_PowerManagerService.cpp

- static JNINativeMethod **gPowerManagerServiceMethods[]** = {

  /* name, signature, funcPtr */

  { "nativeInit", "()V", (void*) android_server_PowerManagerService_nativeInit },

  { "nativeSetPowerState", "(ZZ)V",(void*) android_server_PowerManagerService_nativeSetPowerState },

  };
- int register_android_server_PowerManagerService(JNIEnv* env) {

  int res = jniRegisterNativeMethods(env, "**com/android/server/PowerManagerService**",

  gPowerManagerServiceMethods, NELEM(gPowerManagerServiceMethods));

  }
- static void android_server_PowerManagerService_nativeSetPowerState(JNIEnv* env,

  jobject serviceObj, jboolean screenOn, jboolean screenBright) {

  }

PowerControl IN JNI
Android/frameworks/base/services/jni/onload.cpp

- int register_android_server_PowerManagerService(JNIEnv* env);
- extern "C" jint **JNI_OnLoad**(JavaVM* vm, void* reserved)

  {

  register_android_server_PowerManagerService(env);

  register_android_server_AlarmManagerService(env);

  }

Power Control In JNI
Android/frameworks/base/core/jni/android_os_Power.cpp

- static JNINativeMethod **method_table[]** = {

  { "setScreenState", "(Z)I", (void*)setScreenState },

  };
- int register_android_os_Power(JNIEnv *env)

  {

  return AndroidRuntime::registerNativeMethods(env, "android/os/Power",

  method_table, NELEM(method_table));

  }
- static int setScreenState(JNIEnv *env, jobject clazz, jboolean on)

  {

  return **set_screen_state(on);**

  }

PowerControl in JNI
Android/frameworks/base/core/jni/AndroidRuntime.cpp

- extern int register_android_os_Power(JNIEnv *env);
- static const RegJNIRec **gRegJNI[]** = {

  REG_JNI(register_android_os_Power)

  };
- int AndroidRuntime::startReg(JNIEnv* env)

  {

  register_jni_procs(gRegJNI, NELEM(gRegJNI), env);

  }

JNI Table Datatype and Signature

JAVA Datatype	Signature
Boolean	z
Int	I
Void	V
long	J
String	Ljava/lang/String
Int Array	[I
Boolean Array	[Z

Power Control In Legacy_HAL
Android/hardware/libhardware_legacy/power/power.c

- const char * const NEW_PATHS[] = {

  "/sys/power/wake_lock",

  "/sys/power/wake_unlock",

  "/sys/power/state"

  };
- **Int set_screen_state(int on)**

  {

  if(on)

  len = snprintf(buf, sizeof(buf), "%s", on_state);

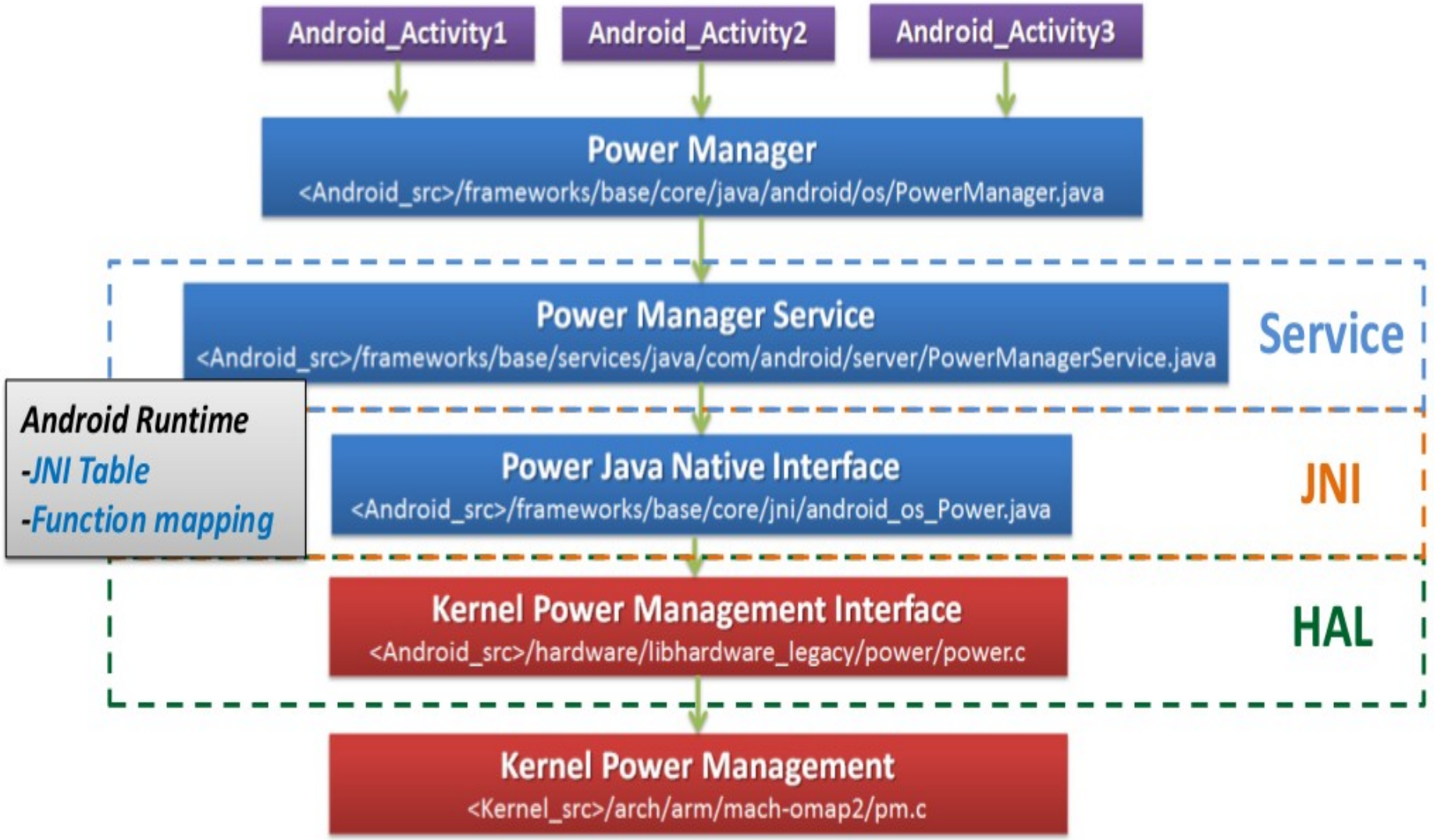
  else

  len = snprintf(buf, sizeof(buf), "%s", off_state);

  len = write(g_fds[REQUEST_STATE], buf, len);

  }

Power Control In android



HAL

- Each hardware must implement the interface in stub format for using in android service
- Stub format

  defined in android/hardware/libhardware/include/hardware/hardware.h

  hw_module_t

  hw_module_method_t

  hw_device_t

  defined in android/hardware/libhardware/hardware.c

  Load()

  Use dlopen() to load *.so

  hw_get_module()

  Get module information and call load() function

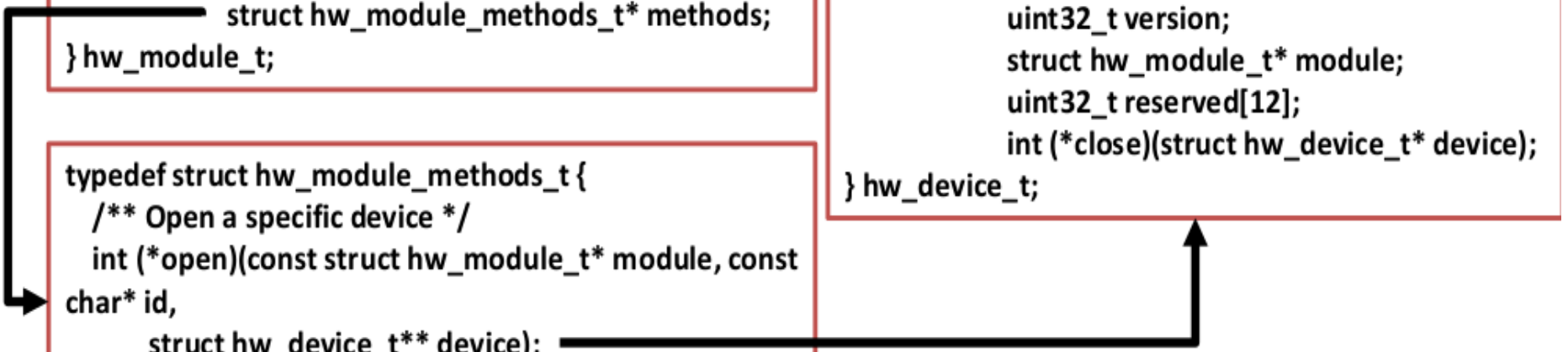
HAL

```
typedef struct hw_module_t {
        uint32_t tag;
        uint16_t version_major;
        uint16_t version_minor;
        const char *id;
        const char *name;
        const char *author;
        /** Modules methods */
        struct hw_module_methods_t* methods;
} hw_module_t;
```

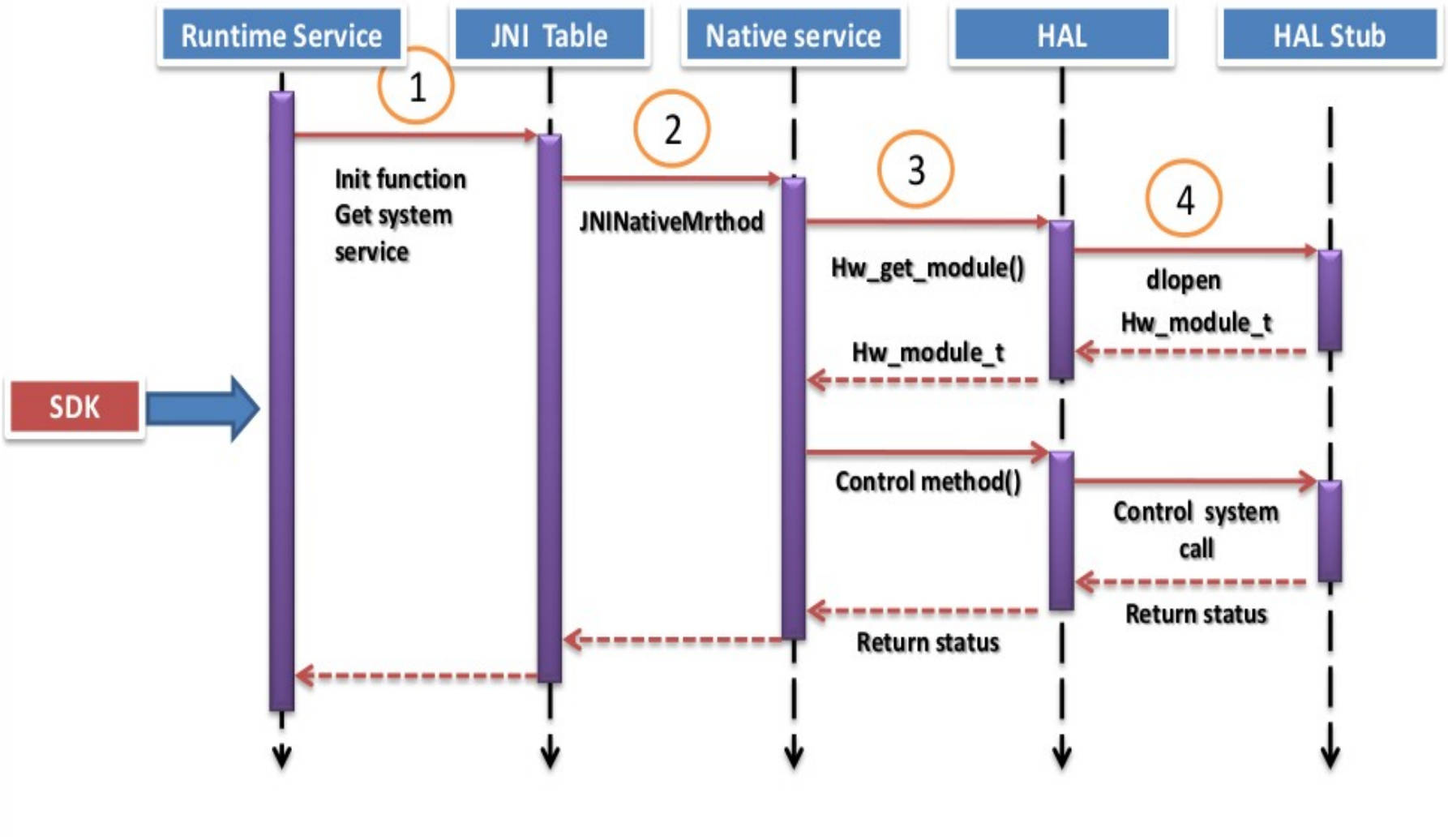
```
typedef struct hw_module_methods_t {
   /** Open a specific device */
   int (*open)(const struct hw_module_t* module, const
char* id,
         struct hw_device_t** device);

} hw_module_methods_t;
```

```
typedef struct hw_device_t {
        uint32_t tag;
        uint32_t version;
        struct hw_module_t* module;
        uint32_t reserved[12];
        int (*close)(struct hw_device_t* device);
} hw_device_t;
```



HAL Stub android Hardware Abstraction Layer



Runtime service

<android_source>/framework/base/services/java/com/android/server

- **private native void nativeInit();**
- private native void nativeSetPowerState(boolean screenOn, boolean screenBright);
- private native void nativeStartSurfaceFlingerAnimation(int mode);

JNI Table
<android_source>/framework/base/services/jni
<android_source>/framework/base/core/jni

static JNINativeMethod gPowerManagerServiceMethods[] = {

/* name, signature, funcPtr */

{ **"nativeInit", "()V“, (void*) android_server_PowerManagerService_nativeInit** },

{ "nativeSetPowerState", "(ZZ)V“, (void*)
android_server_PowerManagerService_nativeSetPowerState },

{ "nativeStartSurfaceFlingerAnimation", "(I)V“, (void*)

android_server_PowerManagerService_nativeStartSurfaceFlingerAnimation },

};

Native service
<android_source>/framework/base/services/jni
<android_source>/framework/base/core/jni

- static android_server_PowerManagerService_nativeInit(JNIEnv *env, jclass clazz)

  {

  module_t const * module;

  hw_get_module(HARDWARE_MODULE_ID, (const hw_module_t)&module);**

  return 0;

  }

Hal
<android_source>/hardware/libhardware/hardwar.c

- int **hw_get_module**(const char *id, const struct hw_module_t **module)

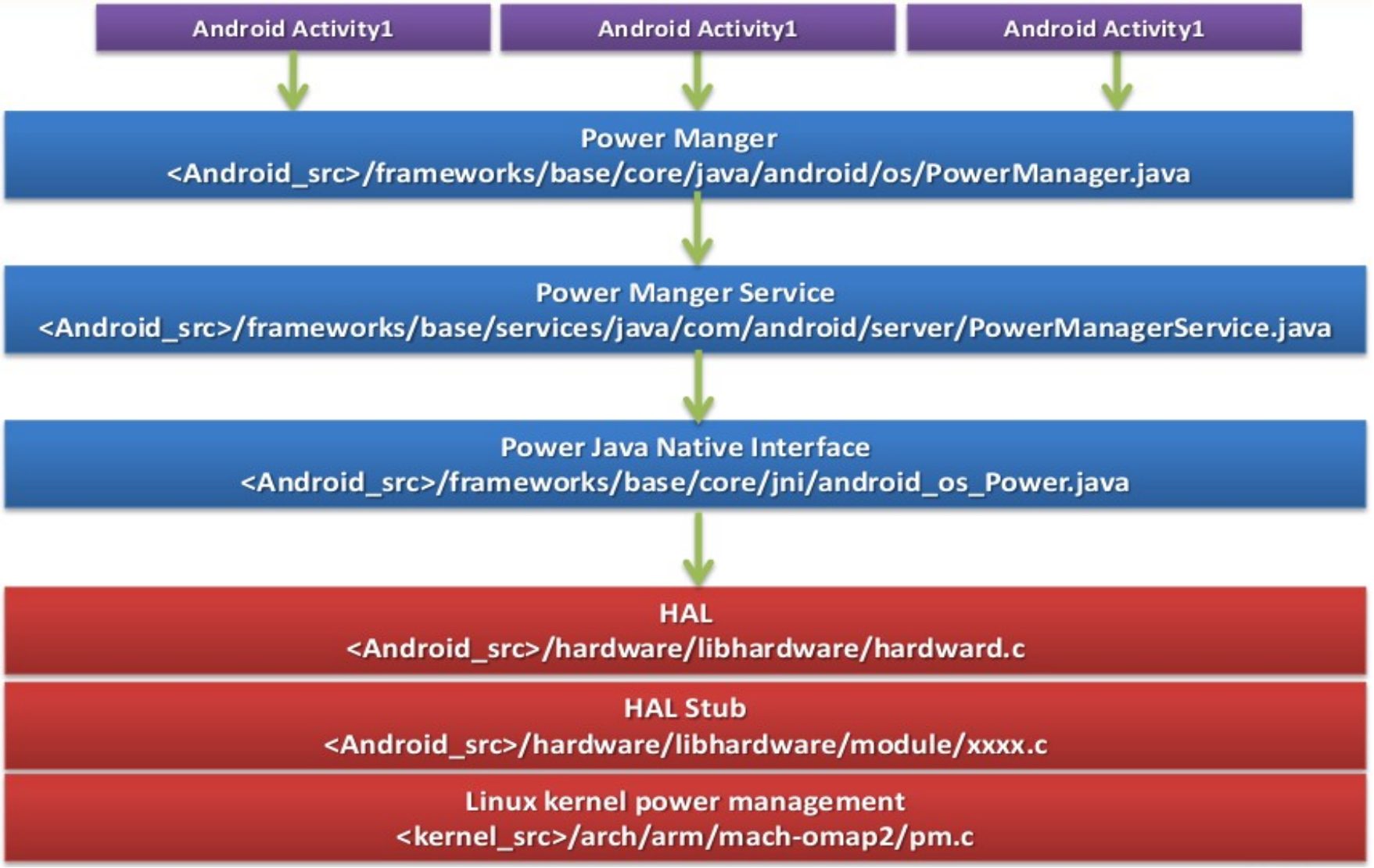
  {

  status = load(id, path, module);

  return status;

  }

Power Control in Android



Q & A

Thank You