Applications of Android

What is Android?

- **Android** is a mobile operating system(OS) currently developed by Google, based on the Linux kernel and designed primarily for touchscreen mobile devices such as smartphones and tablets.
- Android's users interface is mainly based on direct manipulation, using touch gestures that loosely correspond to real-world actions, such as swiping, tapping and pinching, to manipulate on-screen objects, along with a virtual keyboard for text input.
- In addition to touchscreen devices, Google has further developed Android TV for televisions, Android Auto for cars, and Android Wear for wrist watches, each with a specialized user interface.
- Variants of Android are also used on notebooks, game consoles, digital cameras, and other electronics.

Android Applications

We can develop wide variety applications in Android that includes

- Business/Office Applications
- Communication Applications
- Multimedia Applications
- Travel Applications
- Fun Applications/Games
- Utility Applications
- Security Applications

Amongst the wide area of application of Android ,an example of Multimedia is explained.

Rio: A System Solution for Sharing I/O between Mobile Systems

Abstract:

- Mobile systems are equipped with a diverse collection of I/O devices, including cameras, microphones, sensors, and modems
- Rio, an I/O sharing solution that supports unmodified applications and exposes all the functionality of an I/O device for sharing.
- Rio's design is common to many classes of I/O devices, thus significantly reducing the engineering effort to support new I/O devices.
- The implementation of Rio on Android consists of about 7100 total lines of code and supports four I/O classes with fewer than 500 class-specific lines of code.
- Rio also supports I/O sharing between mobile systems of different form factors, including smartphones and tablets

Android Implementation

Architecture:

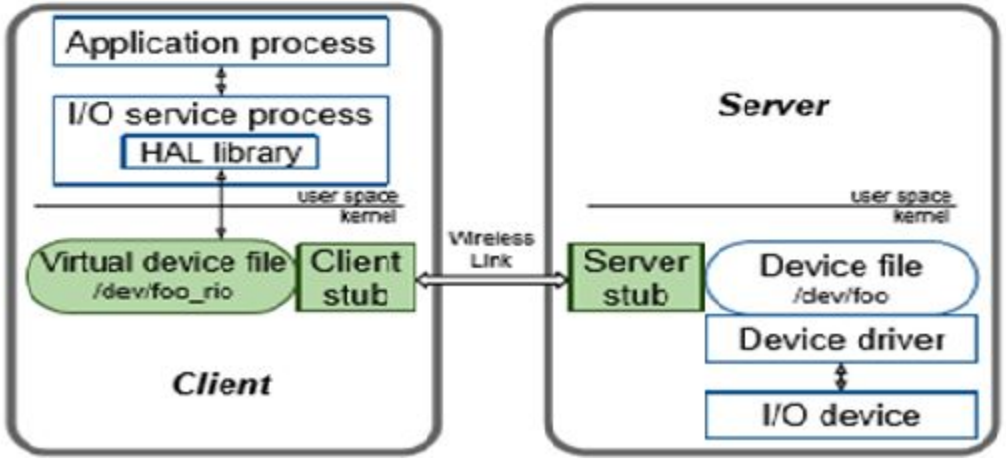
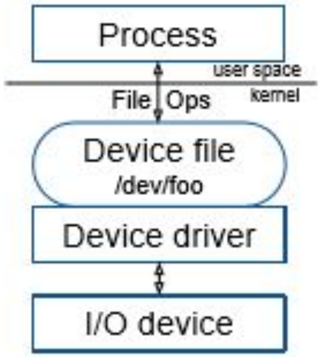
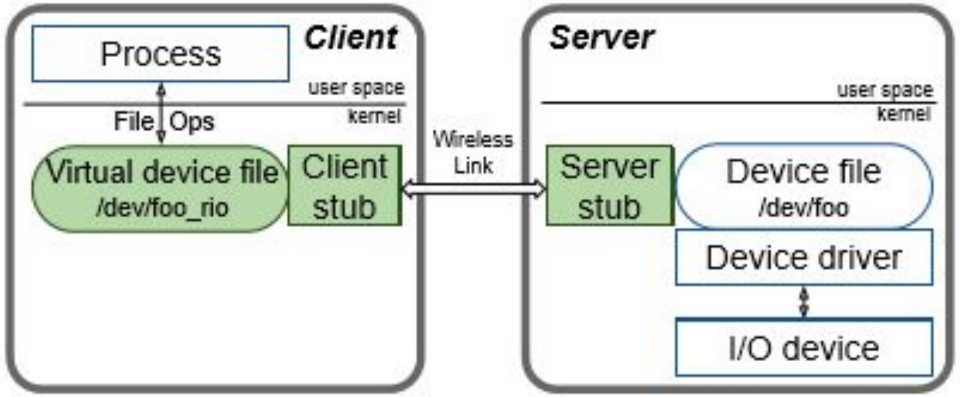


Figure 4: Rio's architecture inside an Android system. Rio forwards to the server the file operations issued by the I/O service process through HAL. Rio supports unmodified applications but requires small changes to the class-specific I/O service process and/or HAL.



(a) I/O stack in Unix-like systems



(b) Rio splits the I/O stack at the device file boundary

Rio splits the I/O stack at the device file boundary. The process that remotely uses the I/O device resides in the client system and interacts with a virtual device file. The actual device file, device driver, and I/O device all reside in the server system. Rio forwards file operations between the client and server. The wireless link can either be through an AP or a device-to-device connection.

On the previous slide Figure 4 shows the architecture of Rio inside an Android system.

- Rio adopts a split-stack model for I/O sharing between mobile systems.
- It intercepts communications at the device file boundary in the I/O stack on one mobile system and forwards them to the other system to be executed by the rest of the I/O stack
- In Android, the application processes do not directly use the device files to interact with the driver.
- Instead, they communicate to a class-specific I/O service process through class-specific APIs.
- The I/O service process loads a Hardware Abstraction Layer (HAL) library in order to use the device file to interact with the device driver.
- Rio's device file boundary lies below the I/O service processes, forwarding its file operations to the server

Implementation Details:

Type	Total LoC	Component	LoC
Generic	6618	Server stub	2801
		Client stub	1651
		Shared between stubs	647
		DSM	1192
		Supporting Linux kernel code	327
Class-specific	498	Camera:	
		- HAL	36
		- DMA	134
		Audio device	64
		Sensor	128
		Cellular modem	136

Table 1: Rio code breakdown.

- Rio is implemented for Android OS and ARM architecture.
- The implementation currently supports four classes of I/O devices:
- sensors (e.g., accelerometer), audio devices (e.g., microphone and speaker), camera, and modem (for phone calls and SMS).
- It consists of about 7100 LoC, fewer than 500 of which are I/O class-specific as shown in Table 1.
- The implementation is tested on Galaxy Nexus smartphones running CyanogenMod 10.1 (Android 4.2.2) atop Linux kernel 3.0, and on a Samsung Galaxy Tab 10.1 tablet running CyanogenMod 10.1 (Android 4.2.2) with Linux kernel 3.1.
- The implementation can share I/O between systems of different form factors like for sharing sensors between a smartphone and a tablet.

UseCases Demonstrated with Rio :

- *Multi-system photography*: With Rio, one can use a camera application on one mobile system to take a photo with the camera on another system. This capability can be handy in various scenarios, especially when taking self-portraits, as it decouples the camera hardware from the camera viewfinder, capture button, and settings. Several existing applications try to assist the user in taking self-portraits using voice recognition, audio guidance, or face detection . However, Rio has the advantage in that the user can see the camera viewfinder up close, comfortably configure the camera settings, and press the capture button whenever ready, even if the physical camera is dozens of feet away. Alternatively, one can use the front camera, which typically has lower quality than the rear-facing one.

- *Multi-system gaming:* Many mobile games require the user to physically maneuver the mobile system for control. Tablets provide a large screen for gaming but are bulky to physically maneuver. Moreover, maneuvers like tilting make it hard for the user to concentrate on the content of the display. With Rio, a second mobile system, e.g., a smartphone, can be used for physical maneuvers while the tablet running the game remains stationary.

- *Music sharing:* A user might want to allow a friend to listen to some music via a music subscription application on her smartphone. With Rio, the user can simply play the music on her friend’s smartphone speaker.

Thank you…