VRPN - Android

Design Document

Eric Boren

Duncan Lewis

Ted Driggs

Kristen Janick

ARCHITECTURE

The System Architecture is displayed in Figure 1.1 below. It shows how the Android Server Application is connected to the VRPN Client Application through the internet. Each of the modules depicted in Figure 1.1 are explained in further detail below.

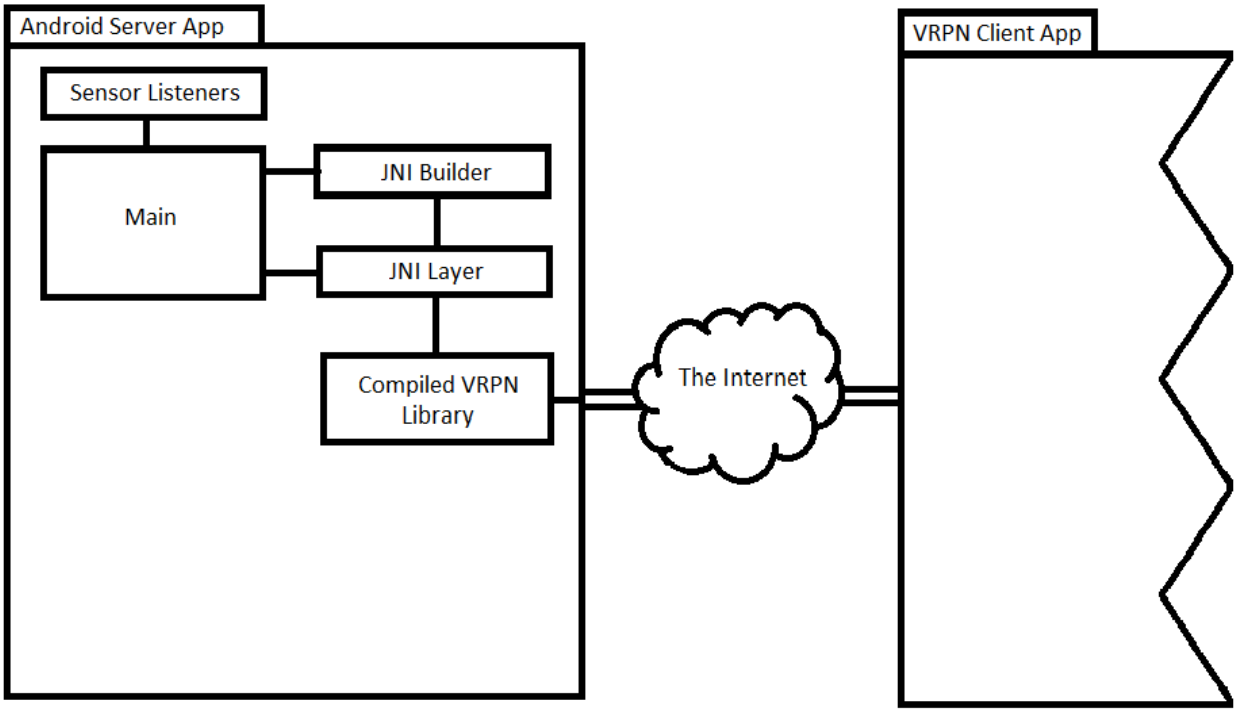

Figure 1.1: System Architecture

Android Server App:

The Android App is the server in the VRPN Client-Server model. The app collects input from the user and serves it to a listening VRPN Client.

- **Main Activity**: This is the main android activity. It contains the interface and processes information received from the **Sensor Listeners**.
- **Sensor Listeners**: These are the hardware sensors which collection sensory information such as multi-touch presses and orientation information.
- **JNI Builder**: This module constructs the **JNI Layer**. It dynamically constructs the **JNI Layer** with the appropriate number of buttons and analog servers.
- **JNI Layer:** This module provides an interface between the Java code of the Android App and the native C/C++ code of the VRPN libraries.
- **Complied VRPN Library:** This is the complete and compiled code of the VRPN Library. It 'automagically' handles the creation and management of the individual servers corresponding to button and analog input sources.

"The Internet"
The Internet is a series of tubes.

VRPN Client App
The VRPN Client App is an arbitrary VRPN compliant application, acting as the client in the VRPN Client-Server model. The client app listens for incoming data sent to it by a server application.

DECOMPOSITION:

The System Architecture displayed in Figure 1.1 has been decomposed into three groups of interaction modules. Each of the modules are explained in further detail below and their interactions are depicted in Figure 2.1, Figure 3.1, and Figure 4.1.

MODULES:

Main Module
The Main module is the Android Activity (http://developer.android.com/reference/android/app/Activity.html) that acts as the 'hub' for all of the other modules. It is responsible for displaying the interface on the Android screen, handling and meeting all Android-related specifications, and acting as the middle ground for the **Sensor Listener**, **JNI Builder**, and **JNI Layer** modules.

Sensor Listener Module
The Sensor Listener module consists of the Accelerometer Listener, the Button Listener, and the Seek Bar Listener sub-modules. Each of these modules handle interaction with Android's Widget API, registering to either a Button Widget (http://developer.android.com/reference/android/widget/Button.html), a SeekBar Widget (http://developer.android.com/reference/android/widget/SeekBar.html), or a hardware Accelerometer (http://developer.android.com/reference/android/hardware/SensorManager.html). The sub-modules handle the reporting of the input values from the widgets and hardware sensors.

JNI Builder Module
The JNI Builder module handles the construction of the **JNI Layer** object. The JNI Builder is seeded with information about the number of **Sensor Listeners** the **Main** module is tracking, and then creates a corresponding **JNI Layer** object.

JNI Layer Module

The JNI Layer module is a two part module consisting of a Java side and an equivalent C/C++ side. The module handles all of the application's interaction between Java and C/C++. Its main function is to send values from the **Sensor Listeners** that have been passed along by the **Main** module. The module also handles constructing and the deconstructing of the **VRPN Android Server** module as is necessary according to the state of the **Main** module and the Android Application.

VRPN Android Server Module

The Android Server module handles the 'automagic' instantiation of the VRPN Servers for Buttons and Analogs in the VRPN Library. These VRPN Library classes in turn handle packing and reporting information received from the **JNI Layer** across the VRPN Connection built during the 'automagic' construction of the VPRN Server.

MODULE INTERACTIONS:

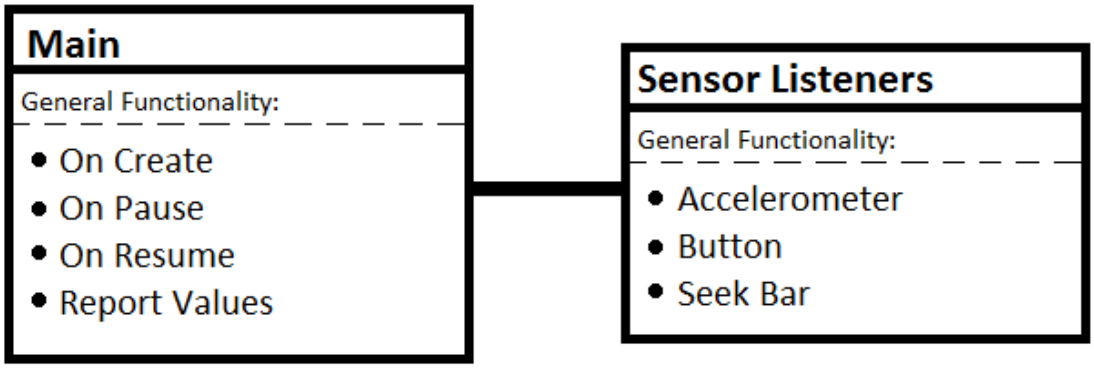

Figure 2.1: Module Interaction between the Main class and the various Sensor Listeners.

Main – Sensor Listener Relation

During the *'On Create'* process, the **Main** module registers event listeners for the **Accelerometer**, **Button**, and **Seek Bar** sub-modules in the **Sensor Listener** Module. Via the event listeners, the **Sensor Listener** sub-modules pass input values along to the **Main** module, where the **Main** module can report them.

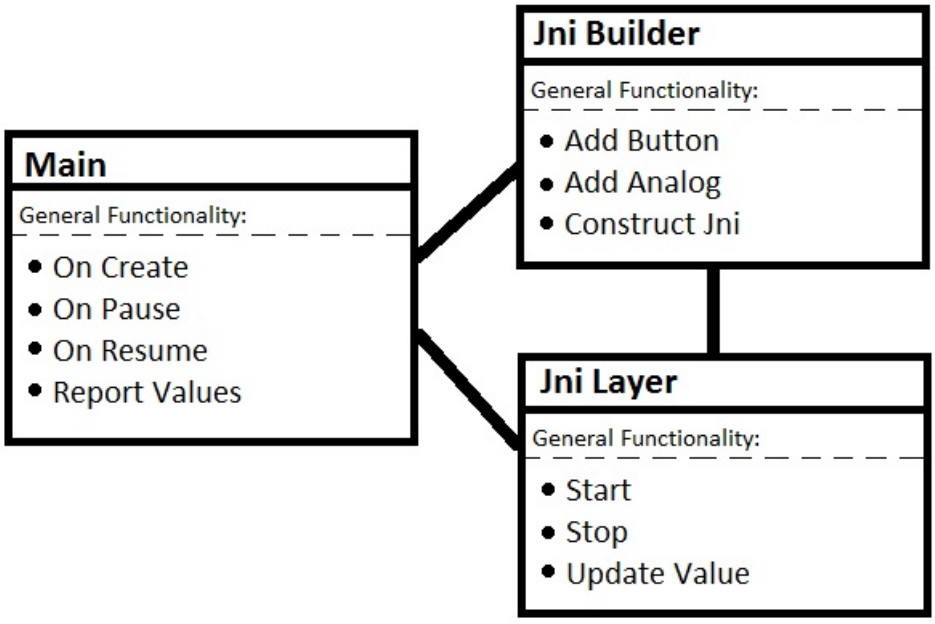

Figure 2.1: Module Interaction between the Main class, the Jni Builder and the Jni Layer.

Main – JNI Builder – JNI Layer Relation

Due to complications arising from the Android Specification, the **Main** module must have both an '*On Pause*' process, and an '*On Resume*' process. The '*On Resume*' process calls the **JNI Builder** module to run the '*Construct JNI*' process, which creates a **JNI Layer** with the inputs specified through the **JNI Builder**'s '*Add Button*' and '*Add Analog*' processes. The '*On Pause*' process destroys and stops the **JNI Layer**, for the purpose of removing dangling references introduced when the Android OS stops and starts an Activity (see the Android Activity Lifecycle: http://developer.android.com/reference/android/app/Activity.html#ActivityLifecycle). The **Main** module also reports the input values collected from **Sensor Listeners** to the **JNI Layer**.

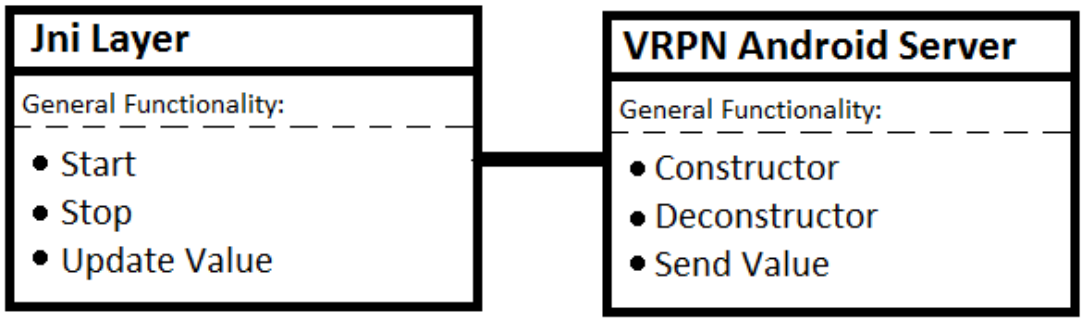

Figure 2.3: Module Interaction between the Jni Layer and the VRPN Android Server.

JNI Layer – VRPN Android Server

When the **JNI Layer's** *'Start'* and *'Stop'* processes are run, the **VRPN Android Server** is constructed and deconstructed, respectively. The construction and deconstruction processes are the **VRPN Library's** 'automagic' instantiating servers and connections. The **JNI Layer** sends along input values by posting them to the **VRPN Android Server** via the *'Update Value'* process, where the **VRPN Android Server's** *'Send Value'* process will inform the VRPN 'automagic' that it has unsent values ready to be shipped.

Summary

Once the 'automagic' decides to and successfully ships the values across all instantiated connections, the chain of processes starting in the **Main** module and trickling down through the **JNI Layer** to the **VRPN Android Server** module is complete, and input values from the Android Server Application have been served to all connected client applications.