intel® Look Inside™

Android™ Configurable Audio Policy

July 2015

Main Goals of Audio Policy

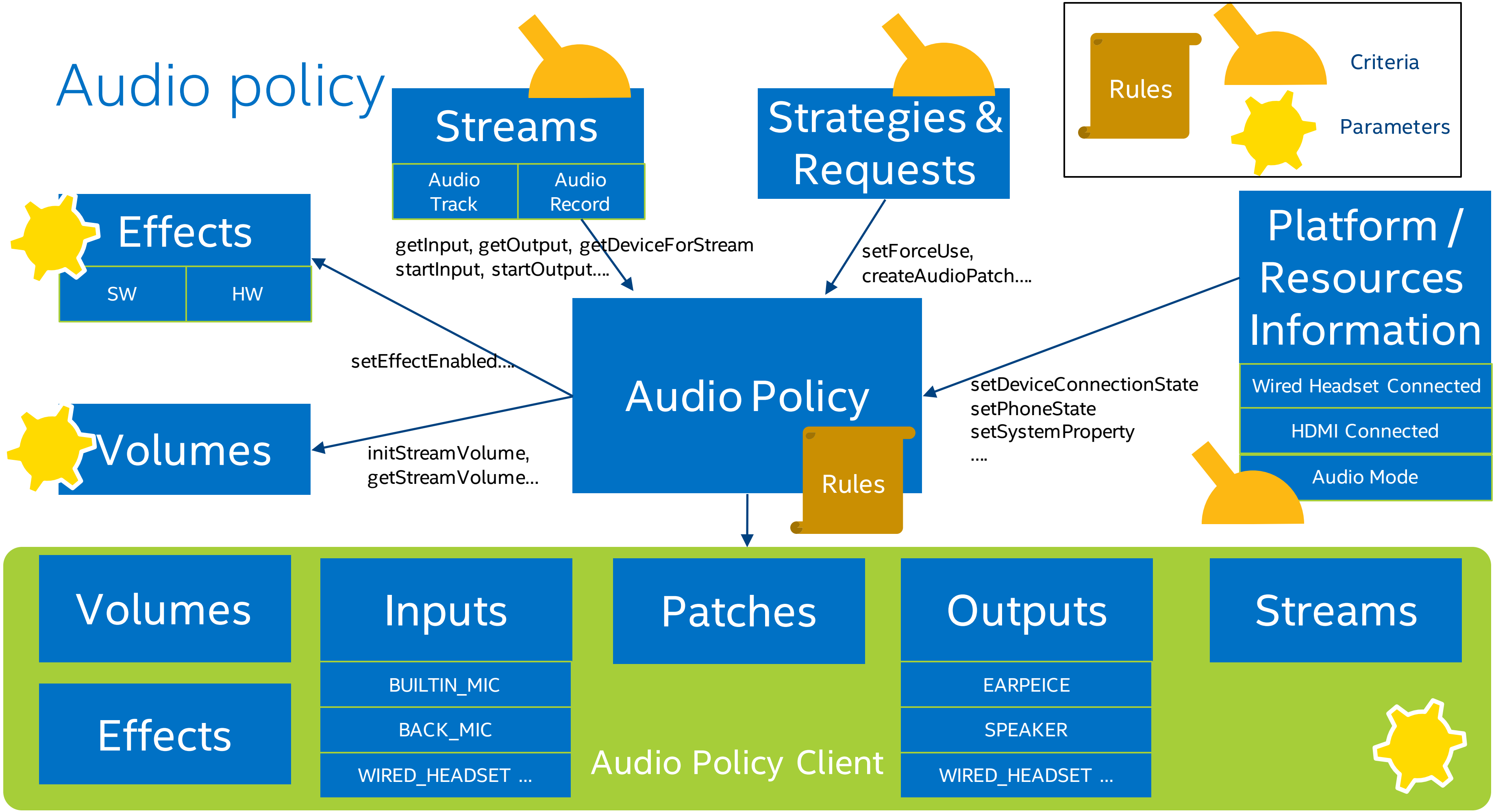
Manage

- selected output/input device for on going use case
- “abstracted” Audio routing (Audio Patch)
- Volume and Mute
- Audio Effects (pre & post), Decoders and Encoders applicability.

according to:

- Product expected behavior
- Application request & “known” System Event (Audio System API)
- Platform capabilities (List of Audio devices) & resources constraints
- Device availabilities (connected/not connected, hotplug,…)
- Streams states

Audio policy



Rules

Criteria

Parameters

Streams

Audio Track

Audio Record

Strategies & Requests

Effects

SW

HW

Volumes

Platform / Resources Information

Wired Headset Connected

HDMI Connected

Audio Mode

getInput, getOutput, getDeviceForStream
startInput, startOutput….

setForceUse,
createAudioPatch….

setEffectEnabled….

initStreamVolume,
getStreamVolume…

setDeviceConnectionState
setPhoneState
setSystemProperty
….

Audio Policy

Rules

Volumes

Effects

Inputs

BUILTIN_MIC

BACK_MIC

WIRED_HEADSET …

Patches

Audio Policy Client

Outputs

EARPEICE

SPEAKER

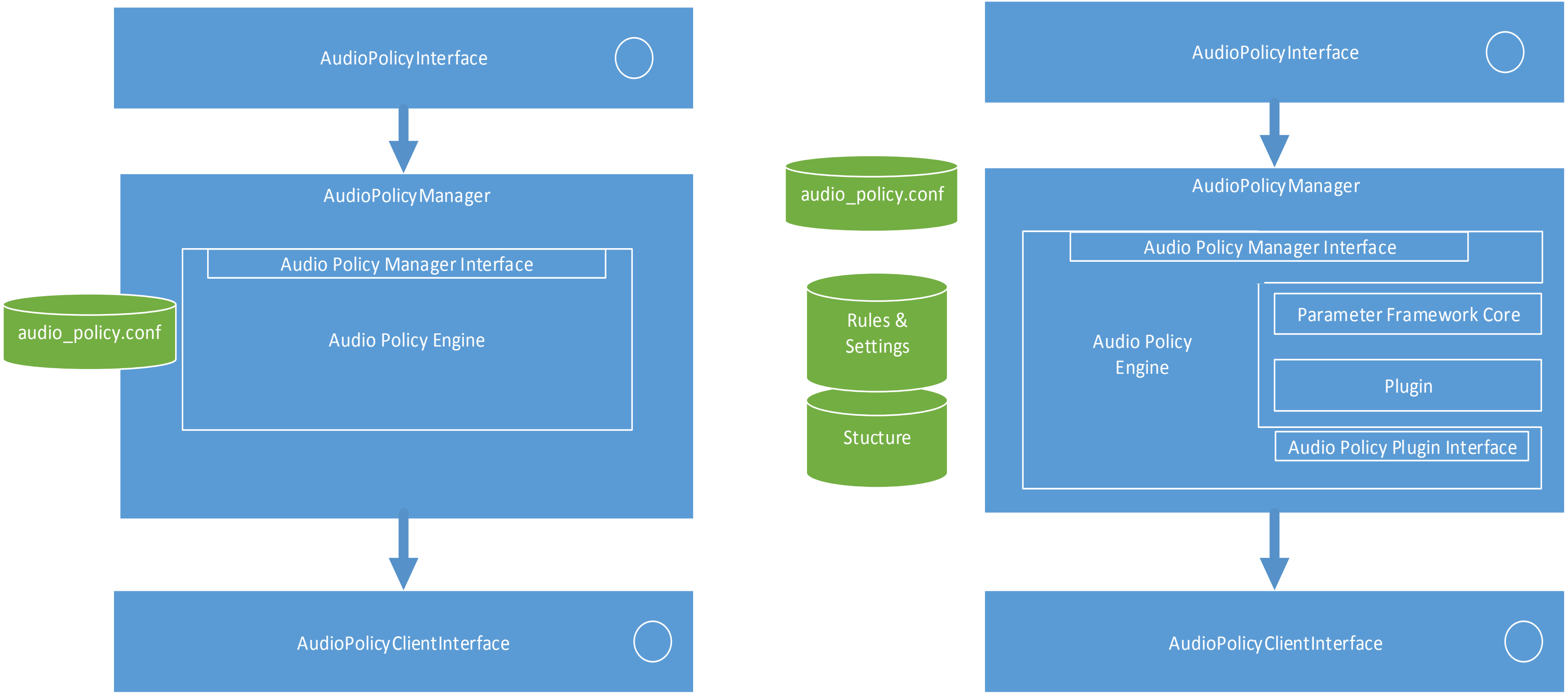
WIRED_HEADSET …

Streams

What is configurable in M release?

- Strategy selection
- Output device selection
- Input device selection
- Volume/Mute management

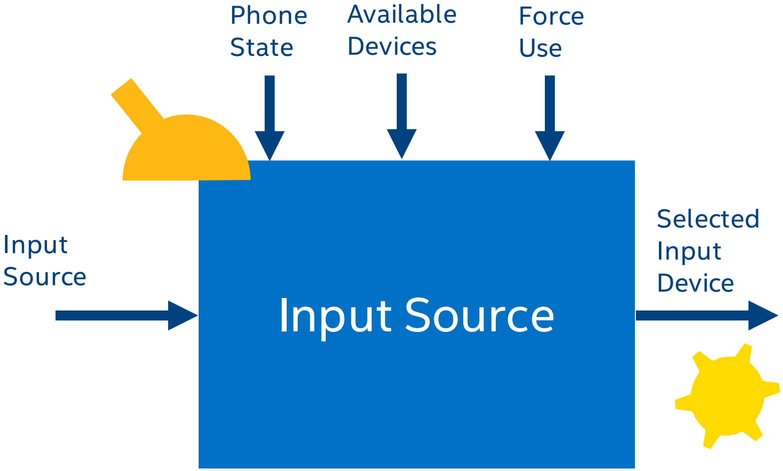
Audio Policy Refactor

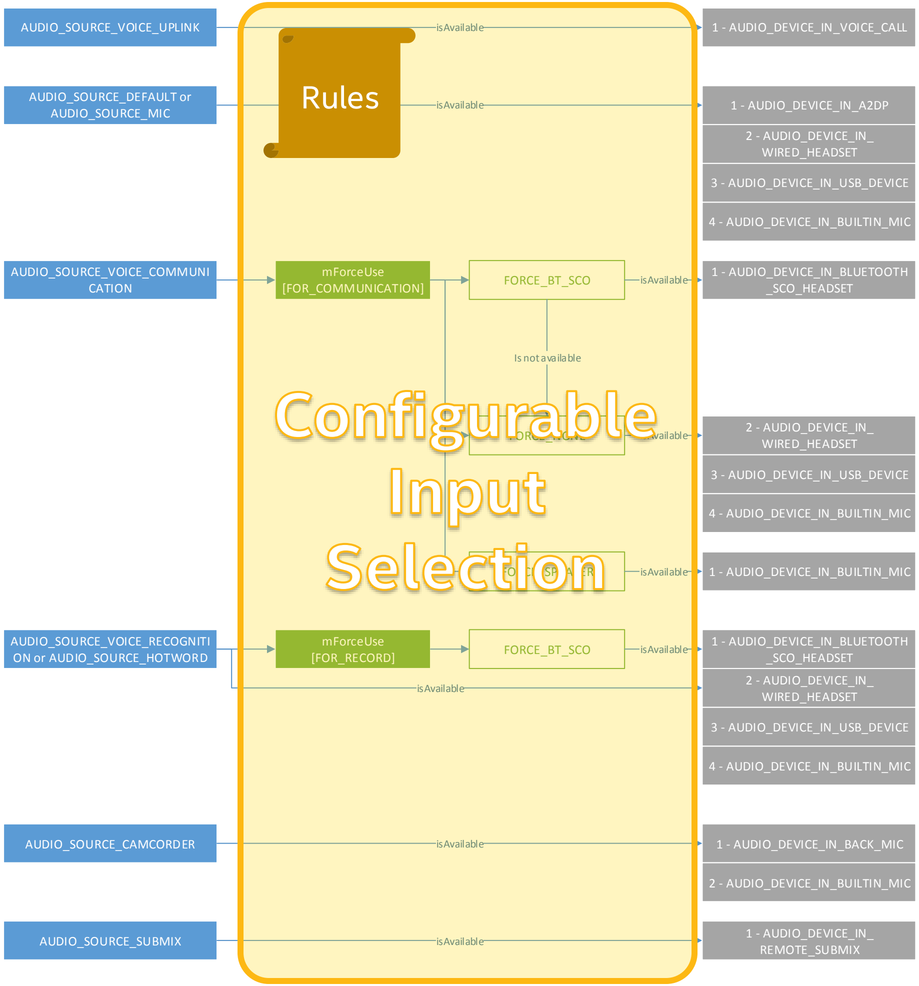


Configurable Audio Policy
Input device selection

Configurable
Input selection

- audio_devices_t AudioPolicyManager::**getDeviceForInputSource**(audio_source_t inputSource)





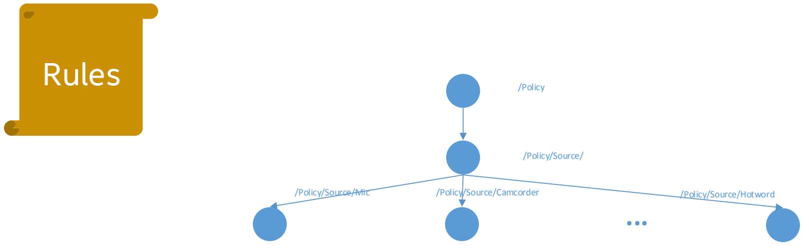
AUDIO_SOURCE_VOICE_UPLINK
AUDIO_SOURCE_DEFAULT or AUDIO_SOURCE_MIC
AUDIO_SOURCE_VOICE_COMMUNICATION
AUDIO_SOURCE_VOICE_RECOGNITION or AUDIO_SOURCE_HOTWORD
AUDIO_SOURCE_CAMCORDER
AUDIO_SOURCE_SUBMIX
Rules
isAvailable
isAvailable
mForceUse [FOR_COMMUNICATION]
FORCE_BT_SCO
isAvailable
Is not available
FORCE_NONE
isAvailable
FORCE_SPEAKER
isAvailable
mForceUse [FOR_RECORD]
FORCE_BT_SCO
isAvailable
isAvailable
isAvailable
isAvailable
Configurable Input Selection
1 - AUDIO_DEVICE_IN_VOICE_CALL
1 - AUDIO_DEVICE_IN_A2DP
2 - AUDIO_DEVICE_IN_WIRED_HEADSET
3 - AUDIO_DEVICE_IN_USB_DEVICE
4 - AUDIO_DEVICE_IN_BUILTIN_MIC
1 - AUDIO_DEVICE_IN_BLUETOOTH_SCO_HEADSET
2 - AUDIO_DEVICE_IN_WIRED_HEADSET
3 - AUDIO_DEVICE_IN_USB_DEVICE
4 - AUDIO_DEVICE_IN_BUILTIN_MIC
1 - AUDIO_DEVICE_IN_BUILTIN_MIC
1 - AUDIO_DEVICE_IN_BLUETOOTH_SCO_HEADSET
2 - AUDIO_DEVICE_IN_WIRED_HEADSET
3 - AUDIO_DEVICE_IN_USB_DEVICE
4 - AUDIO_DEVICE_IN_BUILTIN_MIC
1 - AUDIO_DEVICE_IN_BACK_MIC
2 - AUDIO_DEVICE_IN_BUILTIN_MIC
1 - AUDIO_DEVICE_IN_REMOTE_SUBMIX

Input selection Configurable functions

/frameworks/av/services/audiopolicy/engineconfigurable/parameter-framework/example/Settings/device_for_input_source.pfw

```
domain: Mic
        conf: A2dpSink
                AvailableInputDevices Includes A2dpSink
        /Policy/source/mic/selected_input = A2dpSink
        conf: WiredHeadset
                AvailableInputDevices Includes WiredHeadset
        /Policy/source/mic/selected_input = WiredHeadset
        conf: UsbDevice
                AvailableInputDevices Includes UsbDevice
        /Policy/source/mic/selected_input = UsbDevice
        conf: Default
        /Policy/source/mic/selected_input = BuiltinMic

domain: Camcorder
        conf: Default
        /Policy/source/camcorder/selected_input = BackMic
...
```



Rule				
A2dp Sink Connected	AUDIO_DEVICE_IN_A2DP	AUDIO_DEVICE_IN_BACK_MIC or AUDIO_DEVICE_IN_BUILTIN_MIC or AUDIO_DEVICE_NONE*	…	AUDIO_DEVICE_IN_VOICE_CALL or AUDIO_DEVICE_IN_BUILTIN_MIC or AUDIO_DEVICE_NONE*
A2dp Sink Not Connected and wired headset connected	AUDIO_DEVICE_IN_ WIRED_HEADSET			
A2dp Sink and wired headset not connected and USB headset connected	AUDIO_DEVICE_IN_USB_DEVICE			
Default	AUDIO_DEVICE_IN_BUILTIN_MIC Or AUDIO_DEVICE_NONE*			

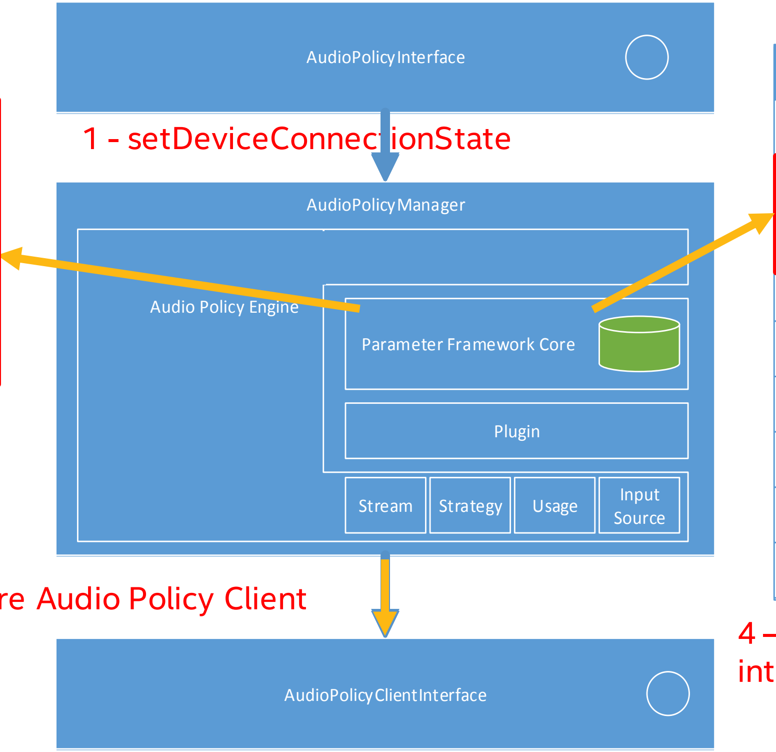
Configure Input Selection According to your needs

Policy interfaces (1/2)

status_t **setDeviceConnectionState**(audio_devices_t device, audio_policy_dev_state_t state, const char *device_address)

3 – apply configurations

InputSource Component	Input Device Parameter Value
AUDIO_SOURCE_DEFAULT	AUDIO_DEVICE_IN_WIRED_HEADSET
AUDIO_SOURCE_MIC	AUDIO_DEVICE_IN_WIRED_HEADSET
AUDIO_SOURCE_VOICE_COMMUNICATION	AUDIO_DEVICE_IN_WIRED_HEADSET
AUDIO_SOURCE_VOICE_RECOGNITION	AUDIO_DEVICE_IN_WIRED_HEADSET
AUDIO_SOURCE_HOTWORD	AUDIO_DEVICE_IN_WIRED_HEADSET
AUDIO_SOURCE_CAMCORDER	AUDIO_DEVICE_IN_BACK_MIC
AUDIO_SOURCE_SUBMIX	AUDIO_DEVICE_IN_REMOTE_SUBMIX
AUDIO_SOURCE_VOICE_UPLINK	AUDIO_DEVICE_IN_VOICE_CALL



1 - setDeviceConnectionState

5 – configure Audio Policy Client

2 - updateCriteria

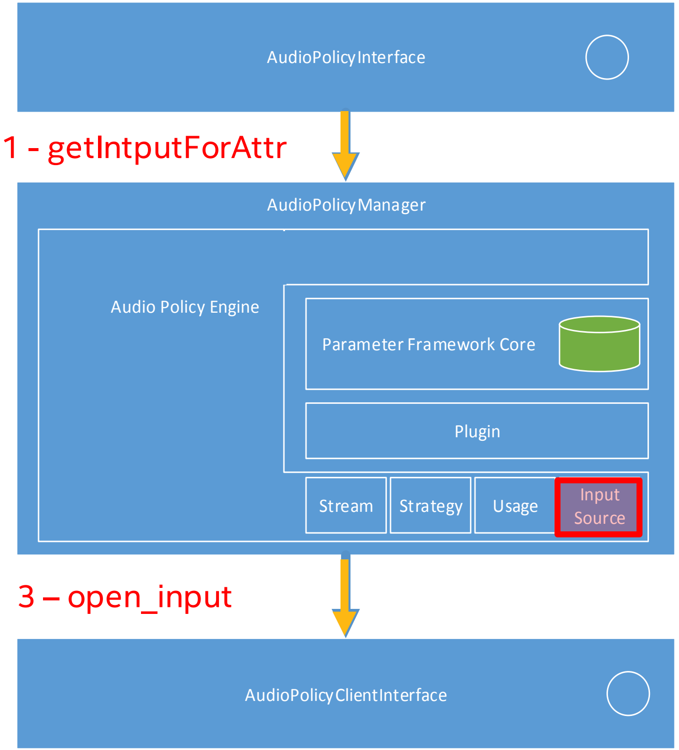
Criteria	Value
TelephonyMode	InCommunication
AvailableInputDevices	BuiltinMic | BackMic |RemoteSubmix | WiredHeaset
AvailableOutputDevices	Earpeice |Speaker |RemoteSubmix | WiredHeadset
ForceUseForCommunication	ForceNone
ForceUseForMedia	ForceNone
ForceUseForRecord	ForeceNone
ForceUseForDock	ForceNone
ForceUseForSystem	ForceNone
ForceUseForHdmi SystemAudio	ForceNone

4 – Audio Policy Manager checks if intput should change

Policy interfaces (2/2)

status_t **getInputForAttr**(const audio_attributes_t *attr, …)

InputSource Component	Input Device Parameter Value
AUDIO_SOURCE_DEFAULT	AUDIO_DEVICE_IN_BUILTIN_MIC
AUDIO_SOURCE_MIC	AUDIO_DEVICE_IN_BUILTIN_MIC
AUDIO_SOURCE_VOICE_COMMUNICATION	AUDIO_DEVICE_IN_BUILTIN_MIC
AUDIO_SOURCE_VOICE_RECOGNITION	AUDIO_DEVICE_IN_BUILTIN_MIC
AUDIO_SOURCE_HOTWORD	AUDIO_DEVICE_IN_BUILTIN_MIC
AUDIO_SOURCE_CAMCORDER	AUDIO_DEVICE_IN_BACK_MIC
AUDIO_SOURCE_SUBMIX	AUDIO_DEVICE_IN_REMOTE_SUBMIX
AUDIO_SOURCE_VOICE_UPLINK	AUDIO_DEVICE_IN_VOICE_CALL



1 - getIntputForAttr

3 – open_input

Criteria	Value
TelephonyMode	InCommunication
AvailableInputDevices	BuiltinMic | BackMic |RemoteSubmix
AvailableOutputDevices	Earpeice |Speaker|RemoteSubmix
ForceUseForCommunication	ForceNone
ForceUseForMedia	ForceNone
ForceUseForRecord	ForeceNone
ForceUseForDock	ForceNone
ForceUseForSystem	ForceNone
ForceUseForHdmi SystemAudio	ForceNone

2 – AudioPolicyManager gets device from input source

Configurable Audio Policy
Volume management

Default Audio Policy Engine

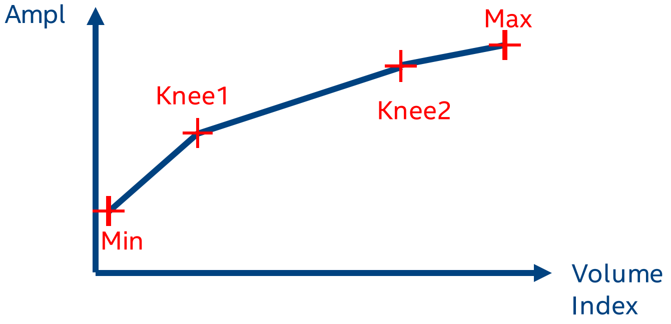
frameworks/av/services/audiopolicy/enginedefault/src/Gains.cpp

Hardcoded tables

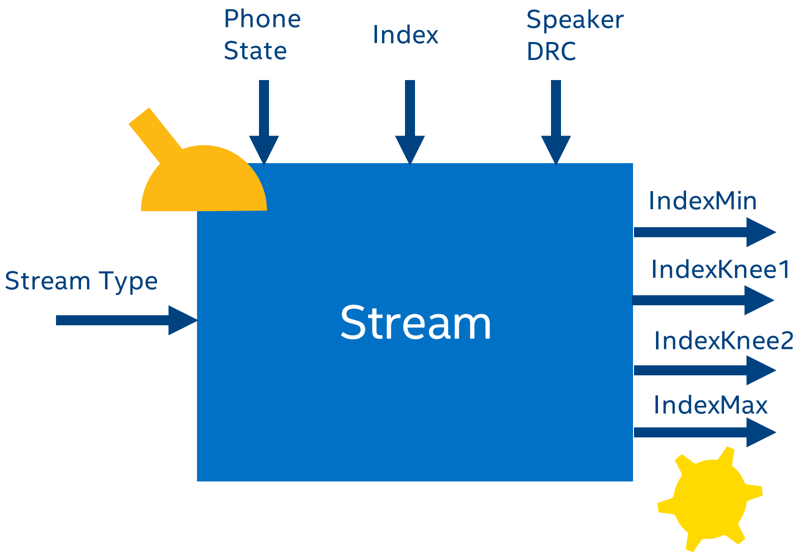
```
const VolumeCurvePoint
Gains::sDefaultVolumeCurve[Volume::VOLCNT] = {
   {1, -49.5f}, {33, -33.5f}, {66, -17.0f}, {100, 0.0f}
};
const VolumeCurvePoint
Gains::sDefaultMediaVolumeCurve[Volume::VOLCNT]  = {
   {1, -58.0f}, {20, -40.0f}, {60, -17.0f}, {100, 0.0f}
};
const VolumeCurvePoint
Gains::sExtMediaSystemVolumeCurve[Volume::VOLCNT]  = {
   {1, -58.0f}, {20, -40.0f}, {60, -21.0f}, {100, -10.0f}
};
const VolumeCurvePoint
Gains::sSpeakerMediaVolumeCurve[Volume::VOLCNT]  = {
   {1, -56.0f}, {20, -34.0f}, {60, -11.0f}, {100, 0.0f}
};
…
```

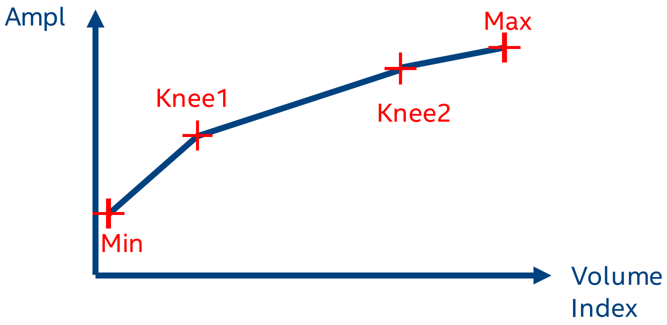
Computing function

```
float Gains::volIndexToDb(Volume::device_category deviceCategory,
const StreamDescriptor&  streamDesc, int indexInUi)
```



How to manage volume tables





Min, Knee1, Knee2, and Max parameters for each devices categories then plugin will compute volume.

Configurable Audio Policy Engine

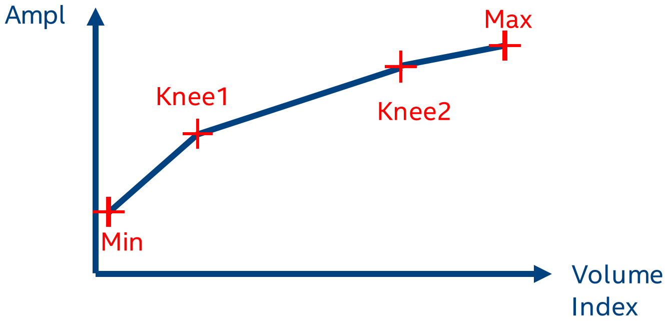
frameworks/av/services/audiopolicy/engineconfigurable/parameter-framework/example/Settings/volumes.pfw

Configurable Volumes

```
domain: Calibration
  conf: Calibration
    component: music/volume_profiles
      component: speaker_device_category/curve_points
          0/index = 1
          0/db_attenuation = -56.0
          1/index = 33
          1/db_attenuation = -34.0
          2/index = 66
          2/db_attenuation = -11.0
          3/index = 100
          3/db_attenuation = 0.0
…
```

Computing function

```
float Gains::volIndexToDb(Volume::device_category deviceCategory,
const StreamDescriptor& streamDesc, int indexInUi)
```



Configure Volume Tables According to your needs intel®