Android Automotive SIG - Audio Hardware Abstraction Layer (HAL)

Android in a Car Audio System

Wassim Filali  May 2020
Strategies for Android & System Level Audio

Topics Relevant for All Strategies

Topics requiring assignment

Audio HAL – Proof Of Concept
Strategies for Android & System Level Audio
Android Usage Strategies

Providing sources and sinks

Controlling the complete System

green and blue boxes can be partitions / virtual machines or different HW

[wiki page: Android & System Level Audio]
Strategies details

• Providing Sources and Sinks
  - Android apps are regarded as Audio sources
  - Devices managed by Android such as Headphones are regarded as Sinks
  - Strategy relies on an external mixer

• Controlling the Complete system
  - All external sources have to be injected into Android
  - The speaker ready stream is output
  - No external mixer required
Feasibility and Functions Partitioning

• Both strategies have advantages and disadvantage
  ▪ Relying on Android only does not fulfill some Safety requirements
  ▪ Considering Android as sources and sinks only, does not take enough advantage of it

• Some Topics have to be handled in both strategies
• Other Topics have to be analysed and assigned: inside or outside Android
Topics Relevant for All Strategies
Extracting Raw Streams

- Streams have properties (Compression, Bit depth,…) \textit{data formats}
- Internal mixer bypass (application or AudioPolicyManager) \textit{Policy config}
- Fixed Volume for external HW volume control \textit{configure volume} (also Fade, Balance,…)
- Available meta-data
  - Usage (Communication, Alarm, Notification,…)
  - Content Type (Unknown, Movie, Music, Sonification, Speech)
Injecting input streams

• external streams can be input and mixed by Android
• new in Android 10 HwAudioSource Player [link](#)
  - type can be configured in audio_policy_configuration.xml
  - low latency routing can bypass AudioFlinger with createAudioPatch()
Topics requiring assignment
Topics requiring assignment

- External Audio Mixing
  - Can be multi stages but last stage can be highly dependent of the custom Amplifier HW

- External Audio Signals
  - some warning signals are safety related
  - some signals require a very early availability
  - some signals require a very low response time
  - some signals require time synchronisation

- Internal User Settings and Control
  - close to the user interface
  - takes advantage of Android User management

- More Topics to be discussed…
Audio HAL – Proof Of Concept
Proof of Concept Overview

- Reference design from Google side by side with the Genivi Proof Of Concept Proposal
  - Android design relies on an External System (Mixer, Amplifier, Safety signals, …)
  - Genivi Proof Of Concept tries to provide an concrete instantiation of the Audio Control Split
Proof of Concept Details

- audio HAL (in the PoC Demo) is derived from goldfish emulator implementation

- limitation:
  - doesn’t differentiate between audio streams,
  - tested with emulator only,
  - no meta information sent,
  - no High-Performance Audio (AAudio, OpenSL ES),

- possible improvements:
  - add info about current volume
  - split audio streams for separate transport
  - add latency measurement,
  - add support for AAudio,

- lessons learned:
  - audio hal is forbidden to use network socket (public SELinux rules),
Proof of Concept Details

Data Flow

- Android Audio framework
- Audio HAL
- AudioRelay
- PCM Writer

- HIDL
- named socket
- TCP
Thank you!

Visit GENIVI:
http://www.genivi.org
http://projects.genivi.org

Contact us:
help@genivi.org