SDV: Integrating Simulink C++ Code in Android Automotive Environment

Rémy Brugnon, Renault Group
Agenda

- “Software Defined Vehicle” context: HPC, SOA, Top-Down development
- Proof of Concept: Connecting Models to Android Inter-Process Communication
- Demo: Climate Control example running on an Android Virtual Machine
- Conclusion and next steps
Context: Renault overall vision for Electric Electronic Platform

- Centralized EE Architecture
- Service Oriented Architecture (System + SW)
- Scalable and Upgradable Platform
Context: Renault upstream project “FACE” (Future Architecture for Automotive Computing Environment)

- Top-Down development flow from Software Architecture to Implementation (Body, ADAS, and Chassis domains)
- Adaptive AUTOSAR running on High Performance Computer
- Service Oriented Architecture (Request/Response methods, events)
Context: Renault SDV Project preparation

- Renault strategic collaboration with Google: Android Automotive OS replaces Adaptive AUTOSAR
- New Interface Definition Language: Android IDL (used for IPC generation)
- Service Oriented Architecture maintained (Request/Response methods => RPC, events => RPC + Callbacks)

RPC: Remote Procedure Call
IPC: Inter-Process Communication
Introduction to Android AIDL binders with a simple example

- Inter-Process Communication / Remote Procedure Call using AIDL Binder:
- MATLAB Simulink modeling
- Embedded Coder C++ code generation: UML Class Diagram representation
- AIDL definition and C++ Binder generation
- C++ Glue code (application main program)
Introduction to Android AIDL binders with a simple example

A simple interface example with one method (no argument)
Introduction to Android AIDL binders with a simple example

Android world

Client side

Simulink world

Server side

« Glue code » world
Introduction to Android AIDL binders with a simple example
Introduction to Android AIDL binders with a simple example

C++ Binder generation from AIDL

Client side

Server side

Service finding

Android service

Service interface implementation

Binder implementation
Climate Control Proof Of Concept

- A control loop (25ms) driving a compressor (cold air)
- A compressor and an evaporator as plant model
- An HMI to activate the climate control and to select the temperature setpoint
- A Service Oriented Architecture using a Request/Response method and Events
- Console output every 250ms for the demo
Climate Control PoC

HMI

ColdSourceTempNeed

CompValveCurrent

EvaporatorTemperature

swcCompCommand model

ON/OFF: bool
[ON : OFF]

swcCompControl model

TEMPERATURE: float

swcRefrigerantLoop model

TEMPERATURE: float

ACTUATOR: float [0:1A]

SENSOR: float [-5:+45°C]
Climate Control PoC Software Architecture (UML Diagram)
Climate Control PoC Software Architecture (UML Diagram)

AIDL translation

FIDL extraction
Climate Control PoC Simulink models
Climate Control PoC Simulink models
Demo!

- Running on Google Cloud Workstation
- Launching Android Cuttlefish VM (Android Open-Source Platform: AOSP)
- Launching Android debug environment (adb)
- Launching 3 applications (Software Components), communicating together:
  - swcCompCommand
  - swcCompControl
  - swcRefrigerantLoop
Conclusion

- MATLAB Simulink is able to model SWC in Service-Oriented Architecture.
- Embedded Coder C++ code generation is easy to connect to an object-oriented RPC inter-process communication like Android offers with AIDL Binders.
- Next technical steps to complete the demonstration:
  - Write a MATLAB script to import AIDLs interfaces in System Composer to create SWC model.
  - Automate main program (glue code) generation from AIDLs.
Thank you for your attention!

Any Questions?