MathWorks
AUTOMOTIVE CONFERENCE 2023
Europe

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

Rémy Brugnon, Renault Group

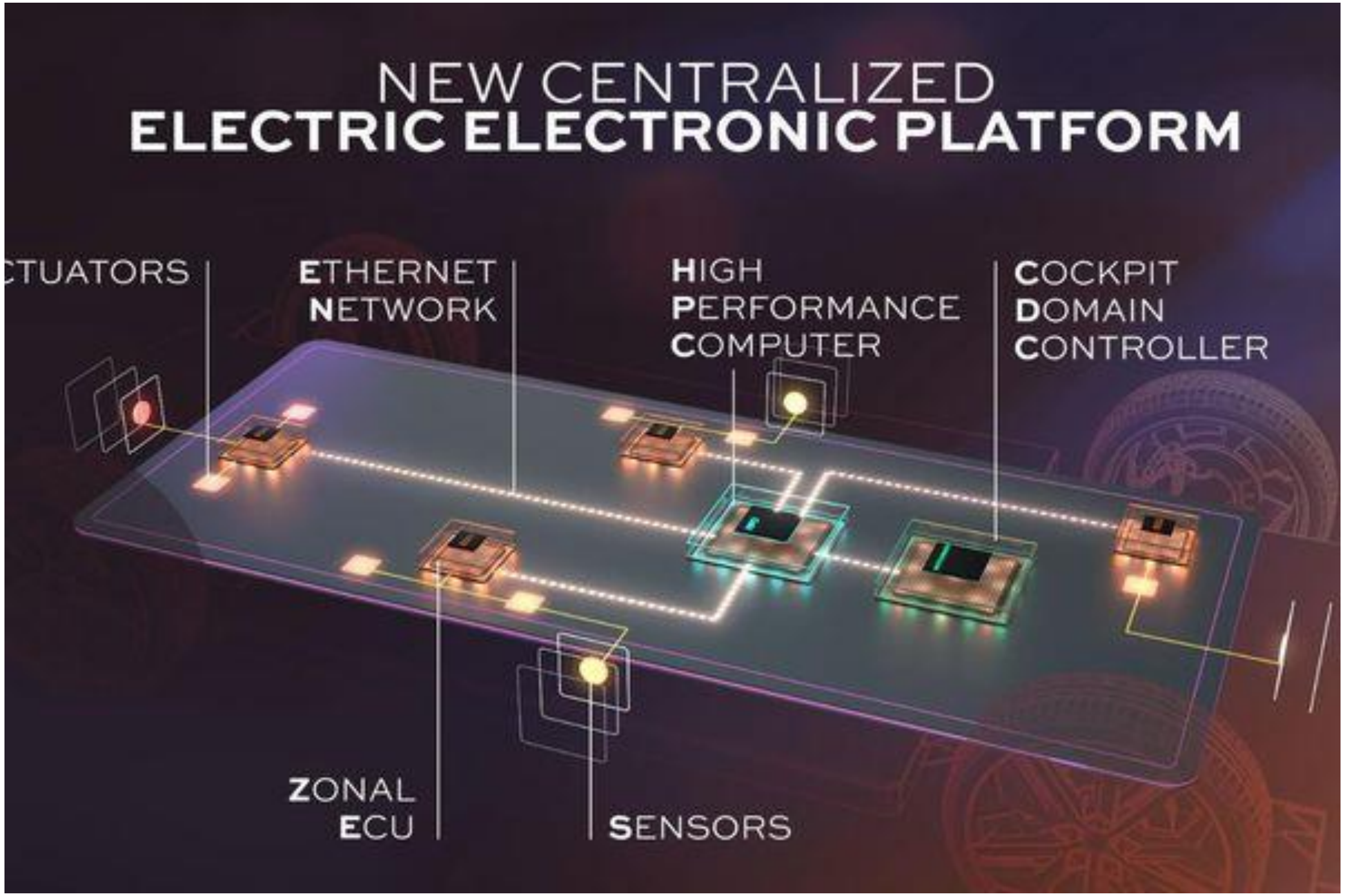
MathWorks

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

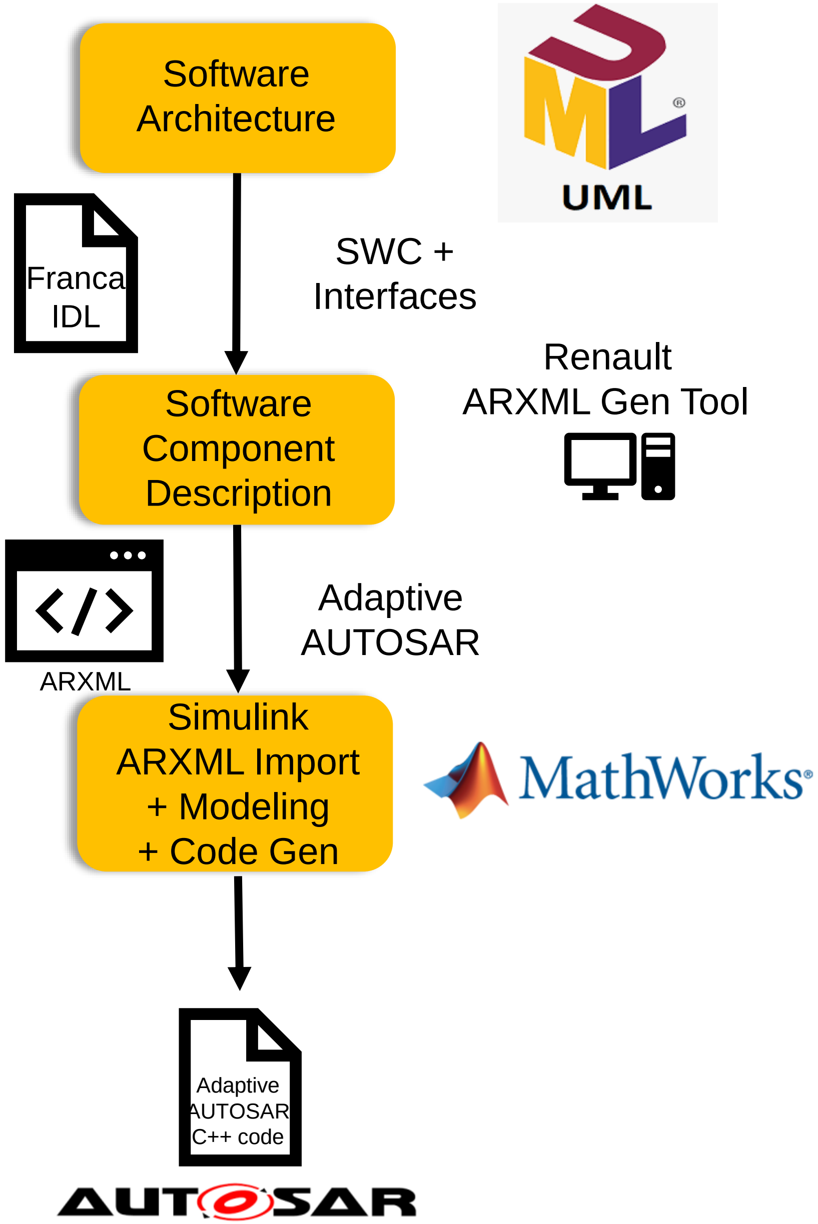
- Centralized EE Architecture
- Service Oriented Architecture (System + SW)
- Scalable and Upgradable Platform


NEW CENTRALIZED
ELECTRIC ELECTRONIC PLATFORM
CTUATORS
ETHERNET NETWORK
HIGH PERFORMANCE COMPUTER
COCKPIT DOMAIN CONTROLLER
ZONAL ECU
SENSORS

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)



Software Architecture

UML

Franca IDL

SWC + Interfaces

Software Component Description

Renault ARXML Gen Tool

ARXML

Adaptive AUTOSAR

Simulink ARXML Import + Modeling + Code Gen

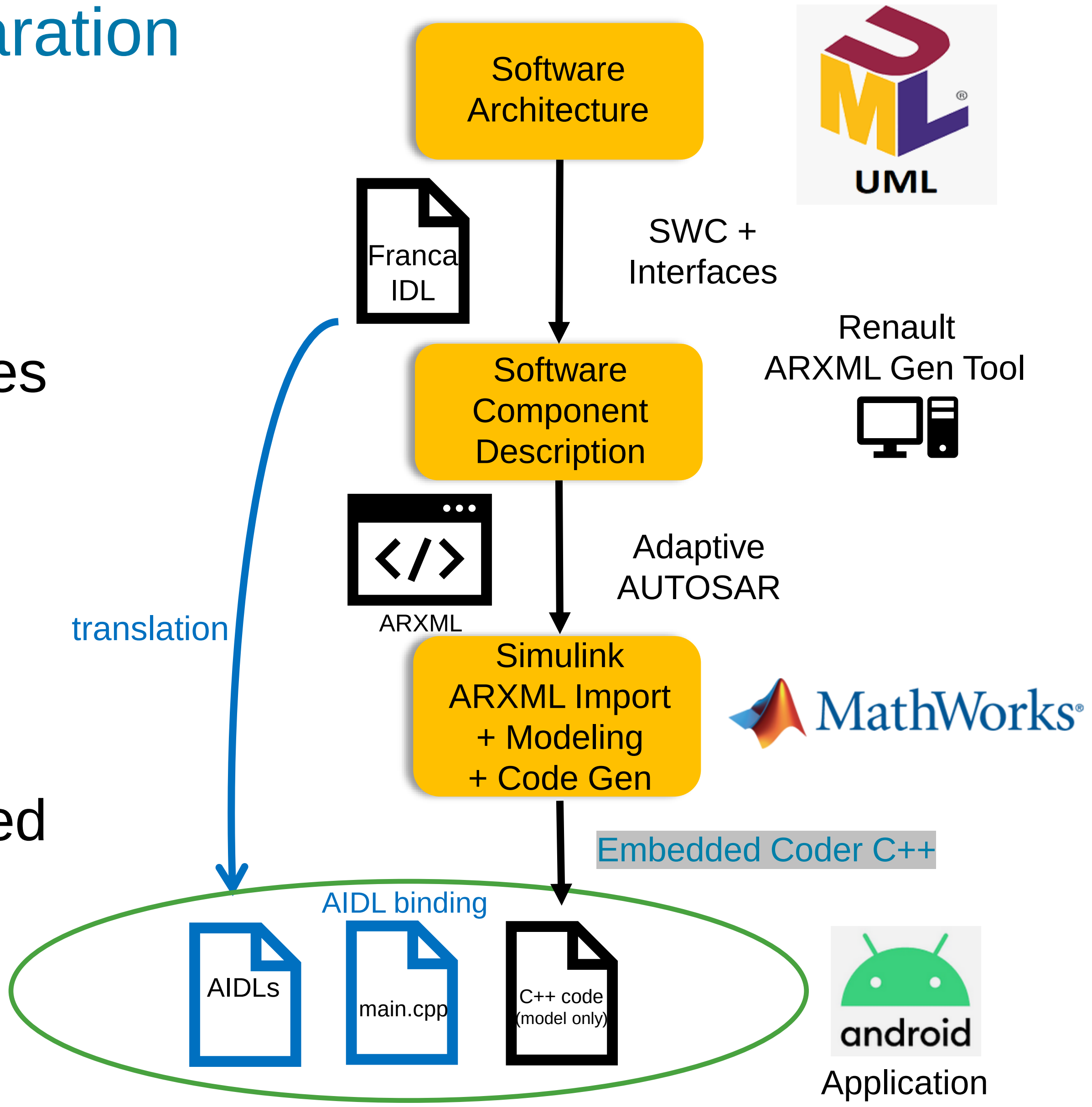
MathWorks

Adaptive AUTOSAR C++ code

AUTOSAR

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)

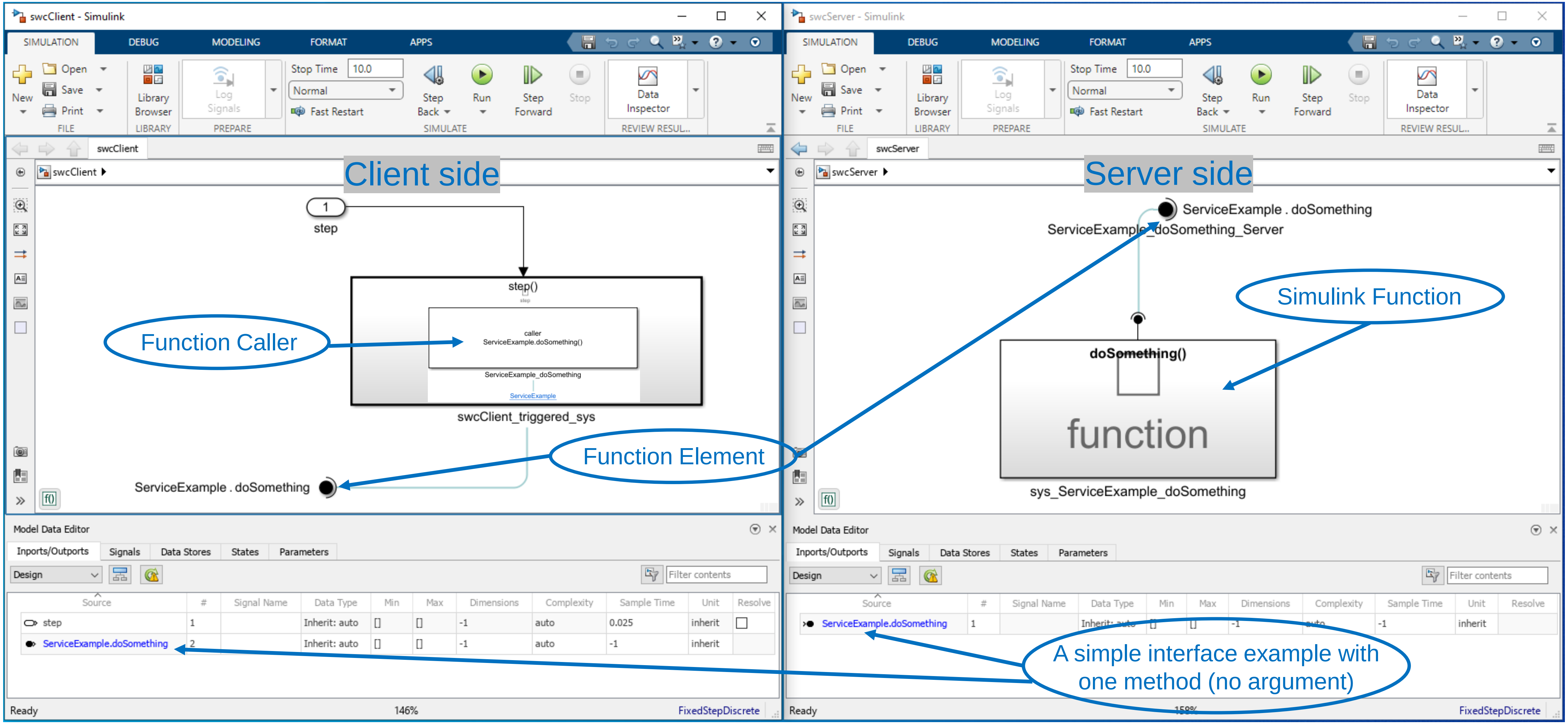


RPC: Remote Procedure Call
IPC: Inter-Process Communication

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



Client side

Server side

Function Caller

Function Element

Simulink Function

A simple interface example with one method (no argument)

Model Data Editor (swcClient)

Inports/Outports | Signals | Data Stores | States | Parameters

Design | Filter contents

Source	#	Signal Name	Data Type	Min	Max	Dimensions	Complexity	Sample Time	Unit	Resolve
step	1		Inherit: auto	[]	[]	-1	auto	0.025	inherit	□
ServiceExample.doSomething	2		Inherit: auto	[]	[]	-1	auto	-1	inherit	

Ready 146% FixedStepDiscrete

Model Data Editor (swcServer)

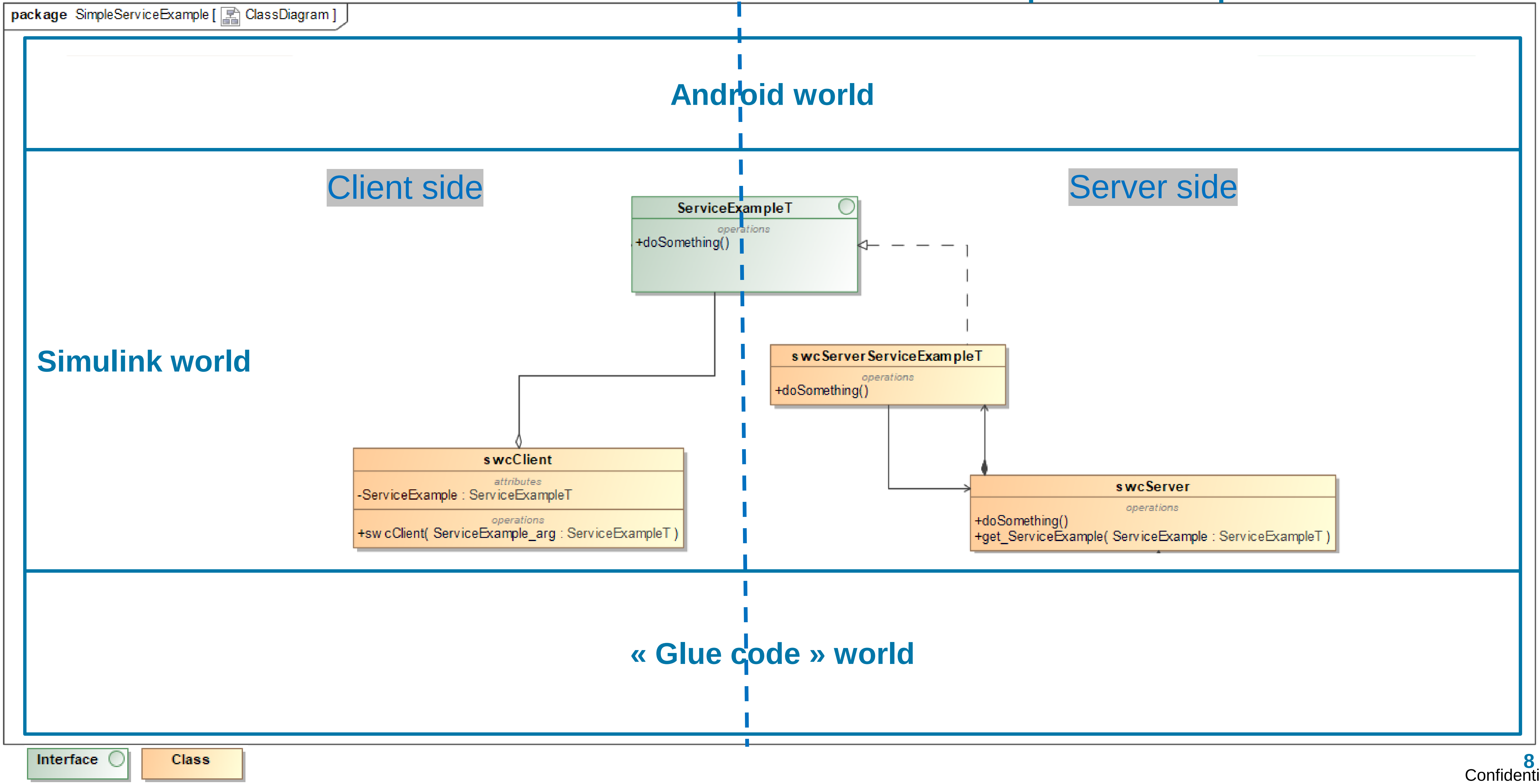
Inports/Outports | Signals | Data Stores | States | Parameters

Design | Filter contents

Source	#	Signal Name	Data Type	Min	Max	Dimensions	Complexity	Sample Time	Unit	Resolve
ServiceExample.doSomething	1		Inherit: auto	[]	[]	-1	auto	-1	inherit	

Ready 158% FixedStepDiscrete

Introduction to Android AIDL binders with a simple example



package SimpleServiceExample [ClassDiagram]

Android world

Client side

Server side

Simulink world

ServiceExampleT
operations
+doSomething()

swcServerServiceExampleT
operations
+doSomething()

swcClient
attributes
-ServiceExample : ServiceExampleT
operations
+swcClient(ServiceExample_arg : ServiceExampleT)

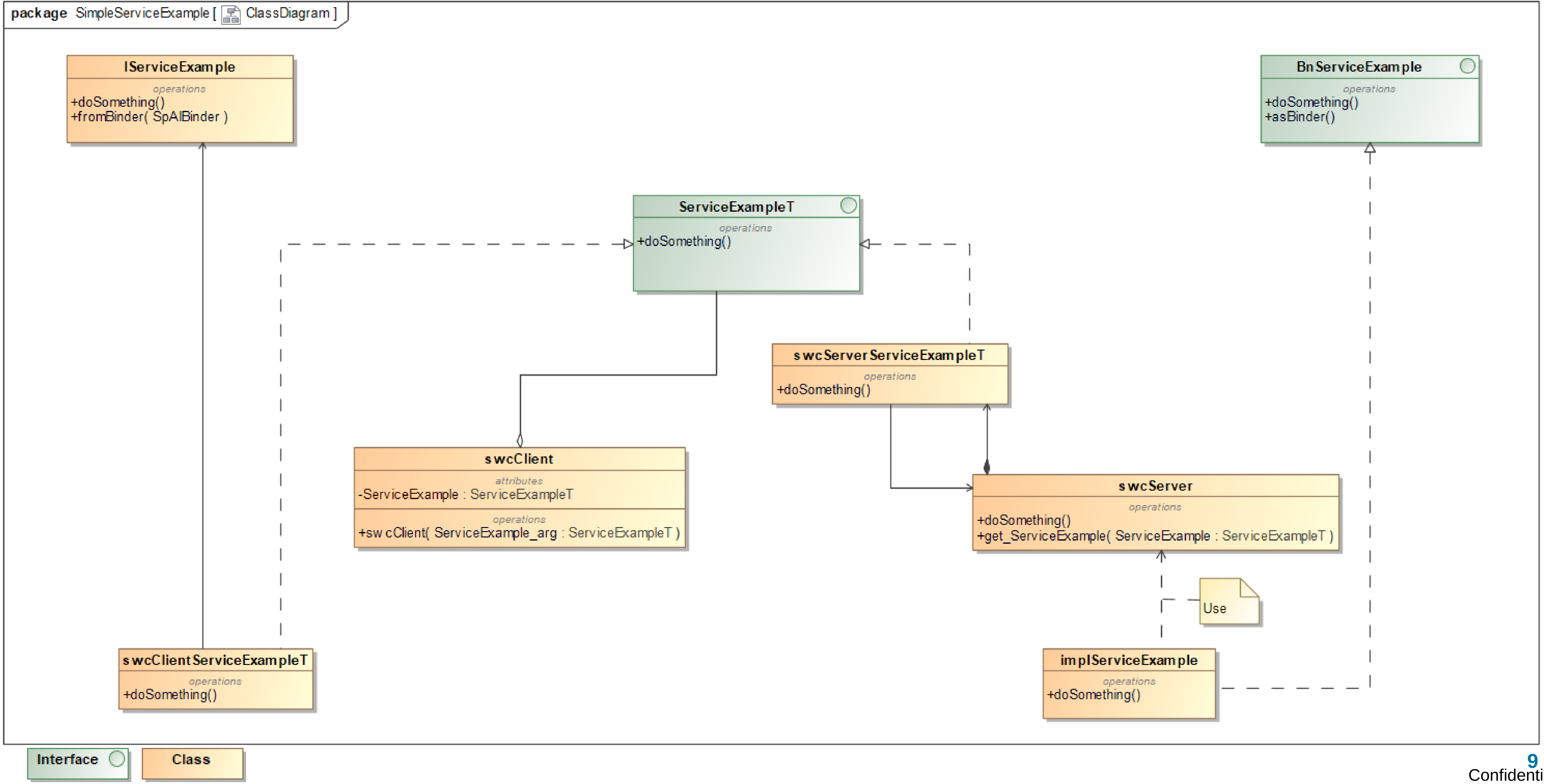
swcServer
operations
+doSomething()
+get_ServiceExample(ServiceExample : ServiceExampleT)

« Glue code » world

Interface ○

Class

Introduction to Android AIDL binders with a simple example



package SimpleServiceExample [ClassDiagram]

IServiceExample
operations
+doSomething()
+fromBinder(SpAIBinder)

BnServiceExample
operations
+doSomething()
+asBinder()

ServiceExampleT
operations
+doSomething()

swcServerServiceExampleT
operations
+doSomething()

swcClient
attributes
-ServiceExample : ServiceExampleT
operations
+swcClient(ServiceExample_arg : ServiceExampleT)

swcServer
operations
+doSomething()
+get_ServiceExample(ServiceExample : ServiceExampleT)

Use

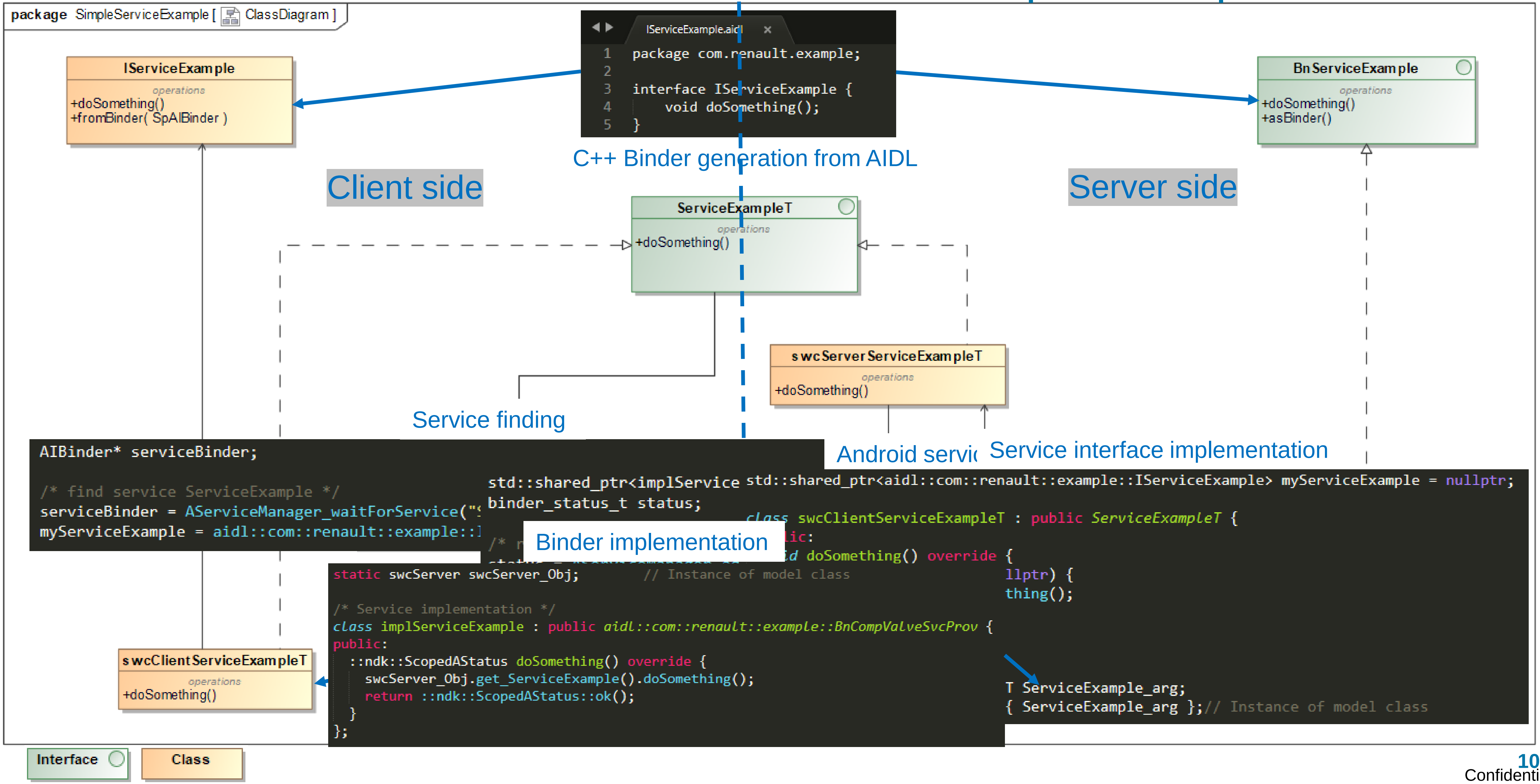
swcClientServiceExampleT
operations
+doSomething()

implServiceExample
operations
+doSomething()

Interface ○

Class

Introduction to Android AIDL binders with a simple example



```
package com.renault.example;

interface IServiceExample {
    void doSomething();
}
```

C++ Binder generation from AIDL

Client side

Server side

Service finding

```
AIBinder* serviceBinder;

/* find service ServiceExample */
serviceBinder = AServiceManager_waitForService("s
myServiceExample = aidl::com::renault::example::I
```

Android servi

```
std::shared_ptr<implService
binder_status_t status;

/* r
```

Binder implementation

```
static swcServer swcServer_Obj;        // Instance of model class

/* Service implementation */
class implServiceExample : public aidl::com::renault::example::BnCompValveSvcProv {
public:
  ::ndk::ScopedAStatus doSomething() override {
    swcServer_Obj.get_ServiceExample().doSomething();
    return ::ndk::ScopedAStatus::ok();
  }
};
```

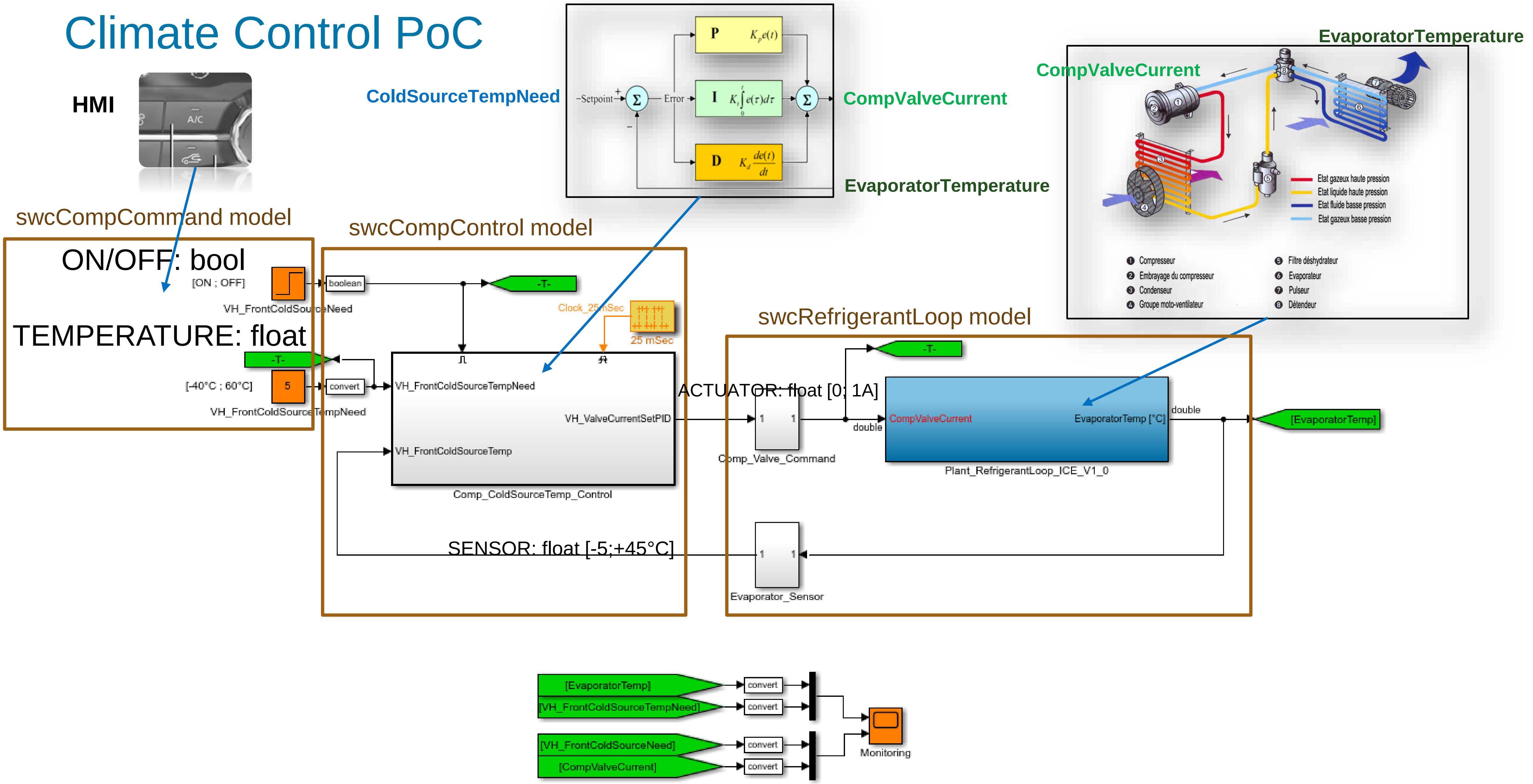
Service interface implementation

```
std::shared_ptr<aidl::com::renault::example::IServiceExample> myServiceExample = nullptr;

class swcClientServiceExampleT : public ServiceExampleT {
lic:
id doSomething() override {
llptr) {
thing();

T ServiceExample_arg;
{ ServiceExample_arg };// Instance of model class
```

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

CompValveCurrent

EvaporatorTemperature

swcCompCommand model

ON/OFF: bool

TEMPERATURE: float

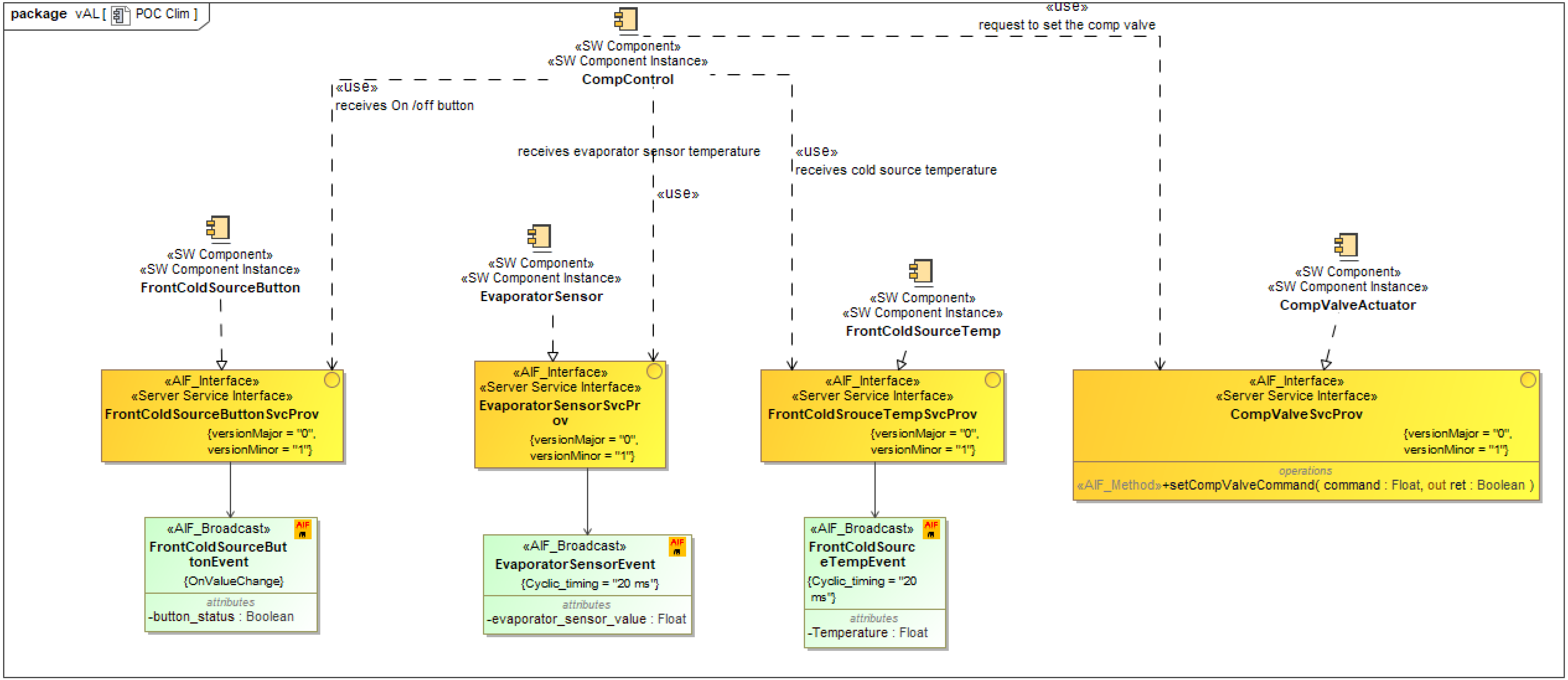
swcCompControl model

swcRefrigerantLoop model

ACTUATOR: float [0; 1A]

SENSOR: float [-5;+45°C]

Climate Control PoC Software Architecture (UML Diagram)



package vAL [POC Clim]

«SW Component»
«SW Component Instance»
CompControl

«use»
request to set the comp valve

«use»
receives On /off button

receives evaporator sensor temperature
«use»

«use»
receives cold source temperature

«SW Component»
«SW Component Instance»
FrontColdSourceButton

«SW Component»
«SW Component Instance»
EvaporatorSensor

«SW Component»
«SW Component Instance»
FrontColdSourceTemp

«SW Component»
«SW Component Instance»
CompValveActuator

«AIF_Interface»
«Server Service Interface»
FrontColdSourceButtonSvcProv
{versionMajor = "0",
versionMinor = "1"}

«AIF_Interface»
«Server Service Interface»
EvaporatorSensorSvcProv
{versionMajor = "0",
versionMinor = "1"}

«AIF_Interface»
«Server Service Interface»
FrontColdSrouceTempSvcProv
{versionMajor = "0",
versionMinor = "1"}

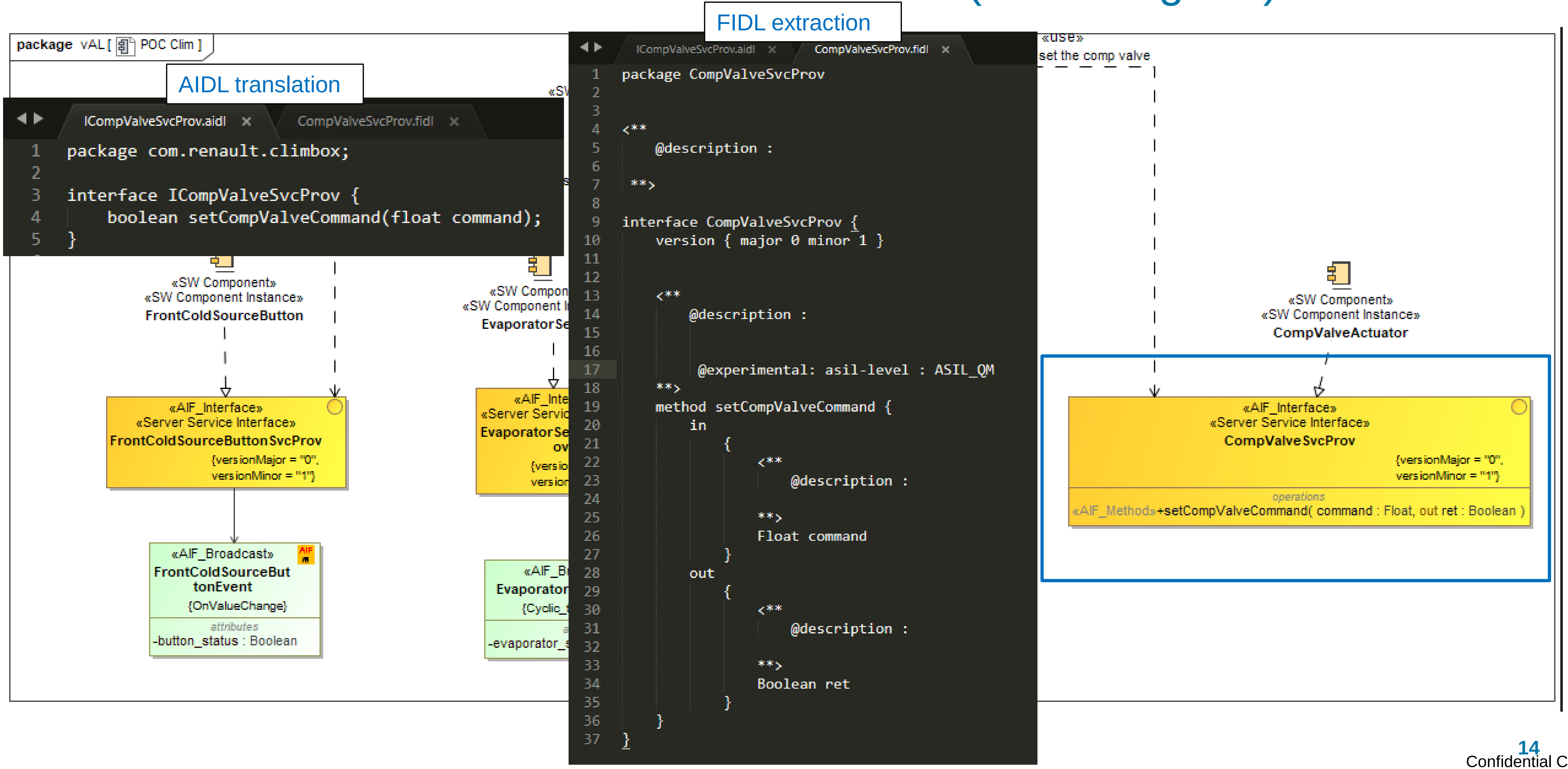
«AIF_Interface»
«Server Service Interface»
CompValveSvcProv
{versionMajor = "0",
versionMinor = "1"}
operations
«AIF_Method»+setCompValveCommand(command : Float, out ret : Boolean)

«AIF_Broadcast»
FrontColdSourceButtonEvent
{OnValueChange}
attributes
-button_status : Boolean

«AIF_Broadcast»
EvaporatorSensorEvent
{Cyclic_timing = "20 ms"}
attributes
-evaporator_sensor_value : Float

«AIF_Broadcast»
FrontColdSourceTempEvent
{Cyclic_timing = "20 ms"}
attributes
-Temperature : Float

Climate Control PoC Software Architecture (UML Diagram)



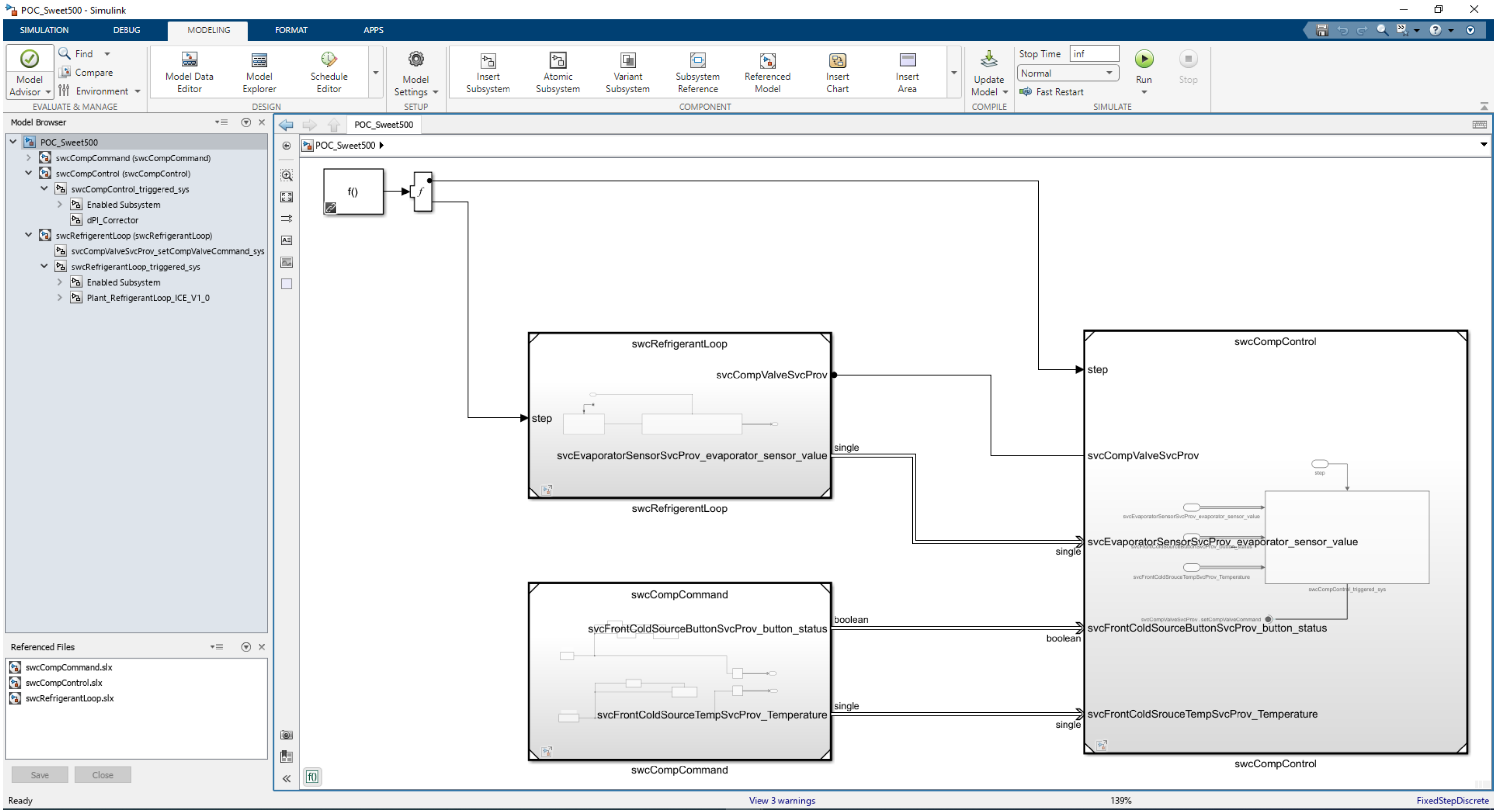
AIDL translation

```
package com.renault.climbox;

interface ICompValveSvcProv {
    boolean setCompValveCommand(float command);
}
```

FIDL extraction

```
package CompValveSvcProv


<**
    @description :

 **>

interface CompValveSvcProv {
    version { major 0 minor 1 }


    <**
        @description :



         @experimental: asil-level : ASIL_QM
    **>
    method setCompValveCommand {
        in
            {
                <**
                    @description :

                **>
                Float command
            }
        out
            {
                <**
                    @description :

                **>
                Boolean ret
            }
    }
}
```

Climate Control PoC Simulink models



POC_Sweet500 - Simulink

Stop Time: inf

Simulation mode: Normal

Model Browser

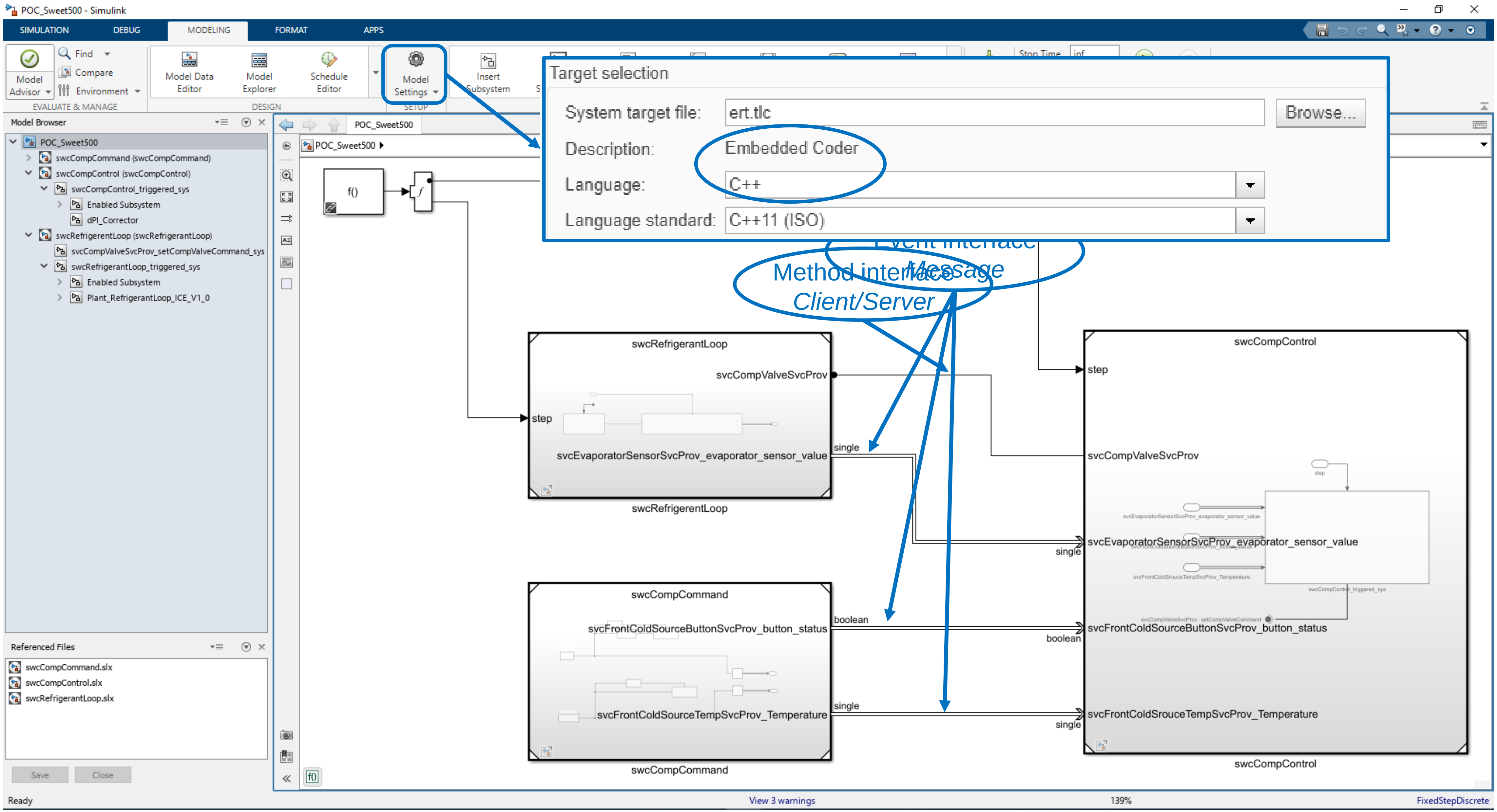
- POC_Sweet500
  - swcCompCommand (swcCompCommand)
  - swcCompControl (swcCompControl)
    - swcCompControl_triggered_sys
      - Enabled Subsystem
      - dPI_Corrector
  - swcRefrigerantLoop (swcRefrigerantLoop)
    - svcCompValveSvcProv_setCompValveCommand_sys
    - swcRefrigerantLoop_triggered_sys
      - Enabled Subsystem
      - Plant_RefrigerantLoop_ICE_V1_0

Referenced Files

- swcCompCommand.slx
- swcCompControl.slx
- swcRefrigerantLoop.slx

Ready | View 3 warnings | 139% | FixedStepDiscrete

Climate Control PoC Simulink models



Target selection

System target file: ert.tlc

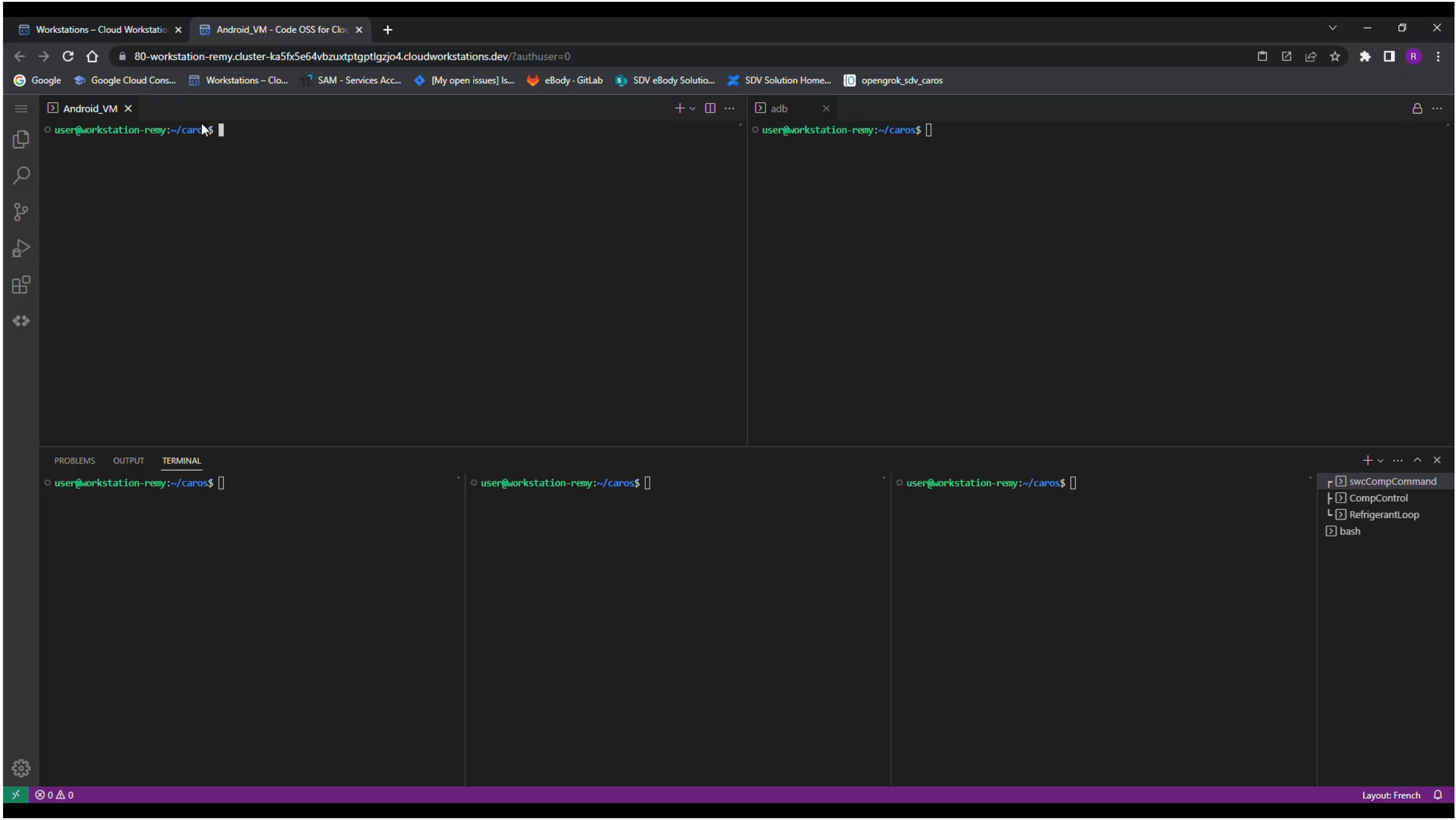
Description: Embedded Coder

Language: C++

Language standard: C++11 (ISO)

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



```
user@workstation-remy:~/caros$
```

```
user@workstation-remy:~/caros$
```

```
user@workstation-remy:~/caros$
```

```
user@workstation-remy:~/caros$
```

```
user@workstation-remy:~/caros$
```

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?