arm

Software Defined AGL with SOAFEE

Matt Spencer
Technical Director, Arm

The Changing Automotive Development and Deployment Paradigm

Centralised Automotive Compute Architecture

Transition towards Software Defined Systems

Ability to monetise after sale

arm

Problem Statement

A Perfect Storm of Complexity

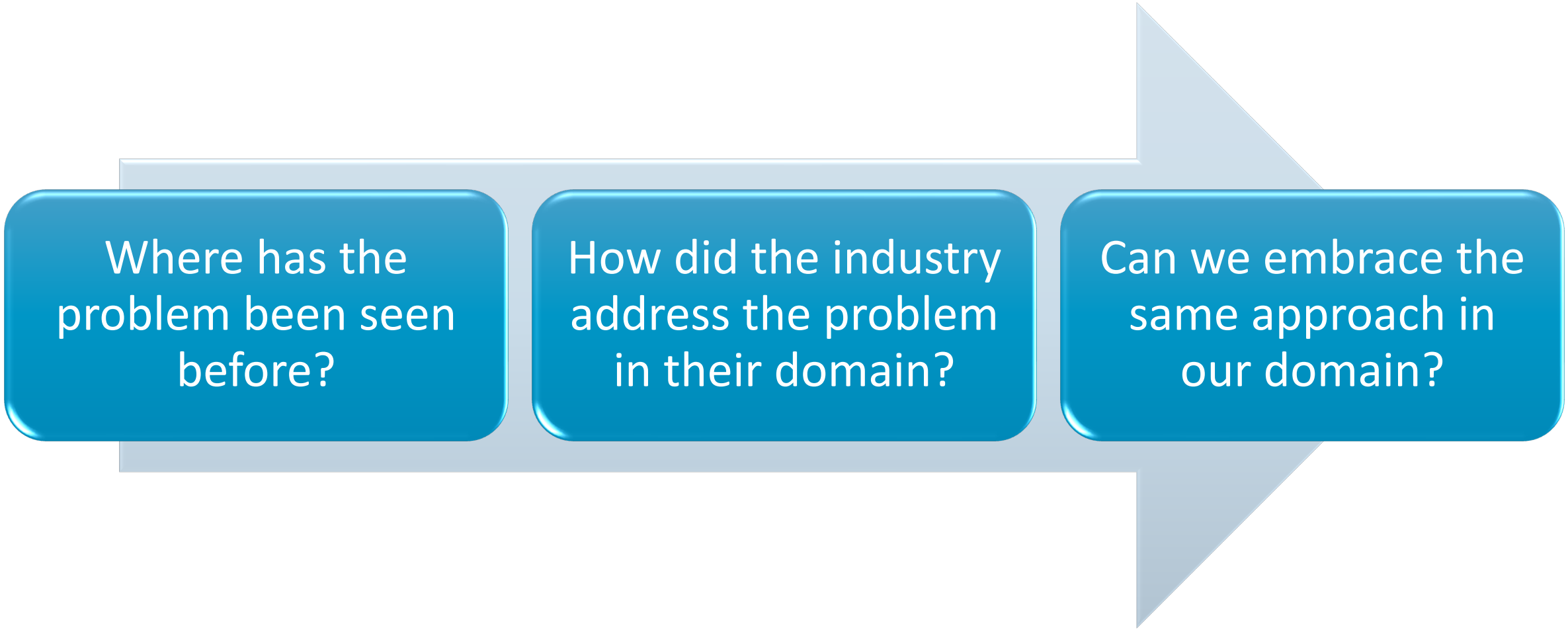
Software Complexity

- Transition towards ‘Software Defined’
  - More capabilities expressed in software
  - Volume of code in production vehicles going through exponential growth
- Software tied to ‘BSP’ through non-standard interfaces and characteristics
- Software is not portable
  - Requires huge amount of re-integration
  - Some OEM’s prefer to start again!
- Unable to amortise software cost over multiple production runs

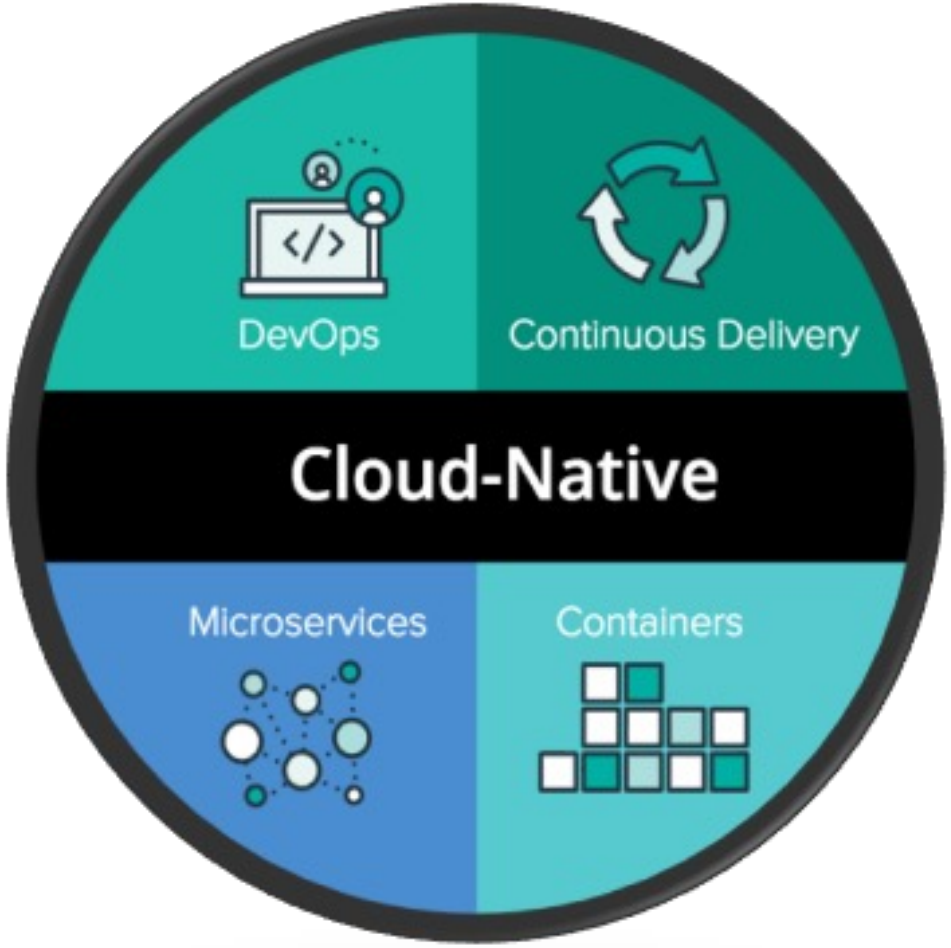
Hardware Complexity

- Heterogeneous SoC’s becoming increasingly complex
  - CPU/MCU/GPU/NPU/…
- Multiple compute islands within the system that have different execution characteristics
- Difficult to the software and developers to make efficient use of all hardware available
- Software becomes tied to a specific hardware architecture and system topography

Researching a solution



Cloud-native ++



- Functional Safety
- Realtime
- Heterogeneous Compute
- Accelerator aware
- IO aware
- …

arm

arm

SOAFEE

Scalable Open Architecture for Embedded Edge

Introducing SOAFEE

Scalable **O**pen **A**rchitecture **F**or **E**mbedded **E**dge

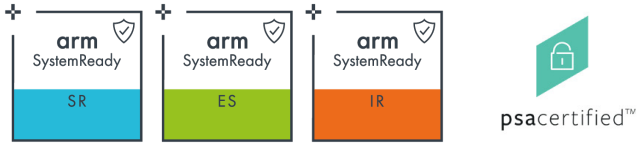
An **industry initiative** to extend cloud-native software experience to automotive workloads, incorporating a **Special Interest Group** (SIG)

A **software architecture** which enables cloud technologies to be combined with automotive functional safety and real-time requirements

An **open-source reference software implementation**, for seeding open-source / commercial ecosystem.

SOAFEE

Enabled by SystemReady, PSA, …

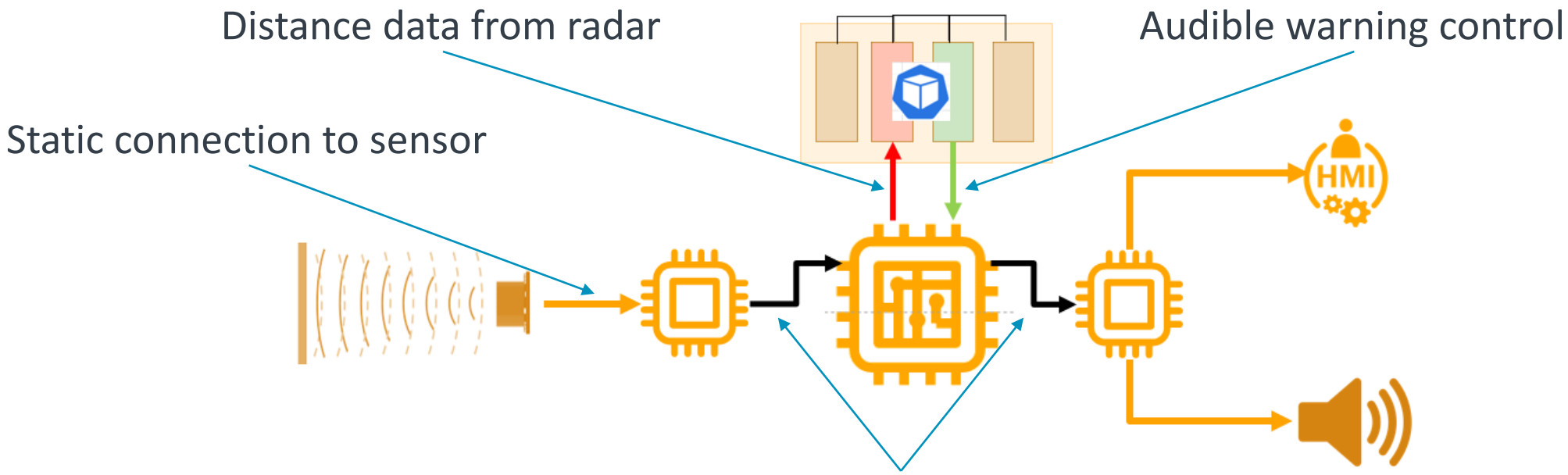


arm

arm

Use-case:
Radar proximity warning

Continental use-case



Continental

IoT Ecosystem Architecture Transformation
Public

November 15, 2021
Albert Kos © Continental AG

1

arm

Orchestrator Actor view of Continental use-case

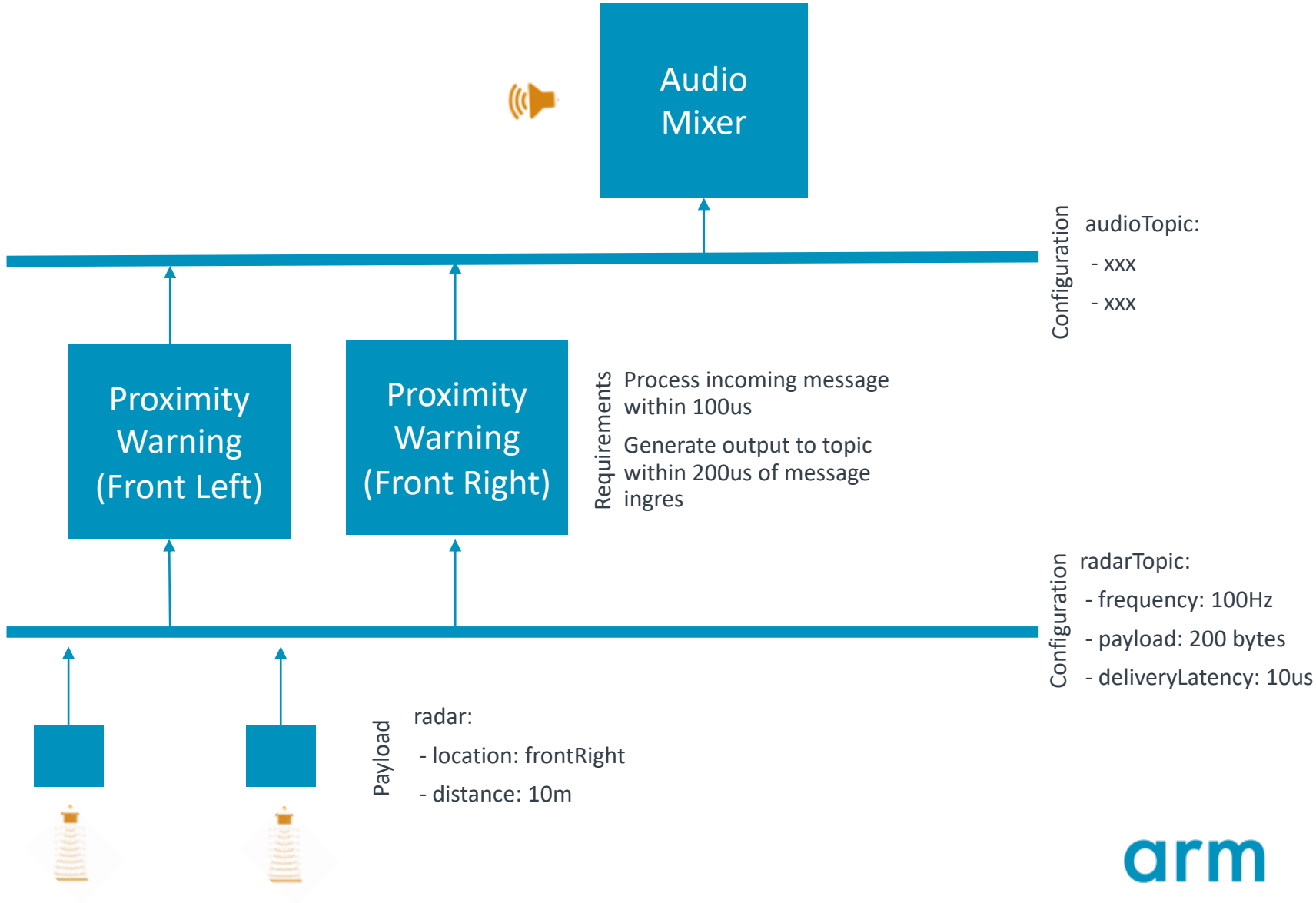
Pub/Sub Topic

Message direction (pub or sub)

Configuration: consumed by orchestrator to configure TSN network through CNI

Payload: Contract between Publisher and Subscriber

Orchestrator knows how many publishers are on a ‘topic’ so can ensure bandwidth availability and configure TSN subsystem appropriately through CNI.

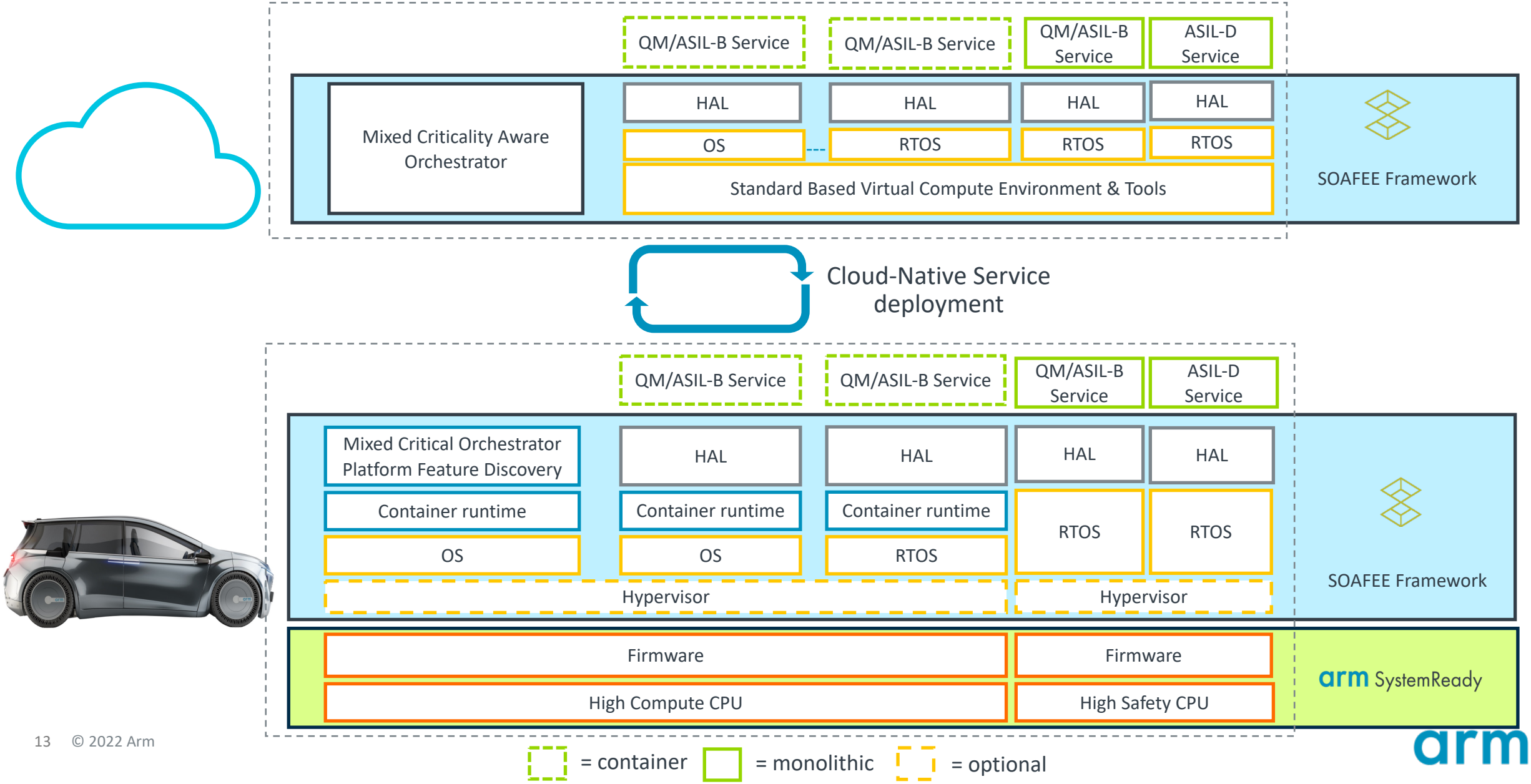


arm

SOAFEE Architecture

SOAFEE Cloud Native Architecture Vision

Framework for enabling mixed critical workload across cloud and vehicle



arm

SOAFEE Ecosystem

Members

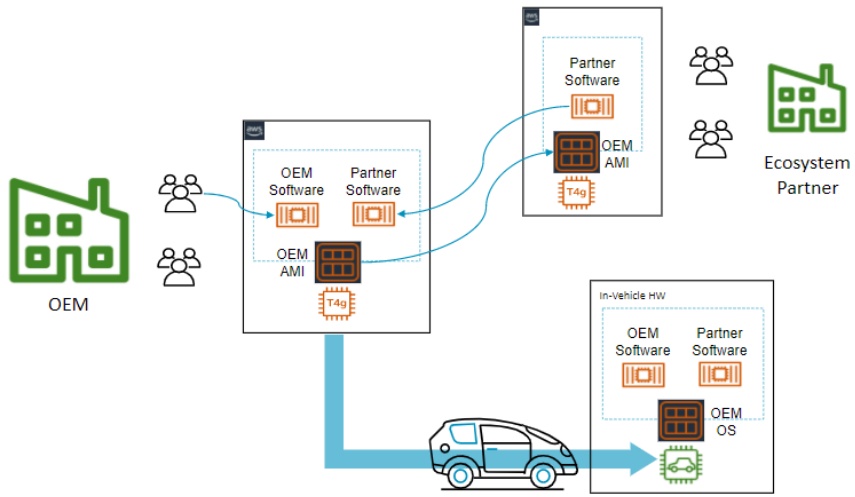
- A framework defined through collaboration
- Representation from across the industry

More being added every week… Details at https://soafee.io/about/members/

Workshops and Whitepapers

SOAFEE Home Blog Community

Accelerating Software-Defined Vehicles through Cloud-To-Vehicle Edge Environmental Parity



In this white paper we look at what it means to apply cloud-native approach to automotive system development, focusing particularly on achieving environmental parity between cloud to edge execution environments. The paper explores the impact this approach has in accelerating the time to market of software-defined vehicles and the role that SOAFEE plays as key technology enabler.

Read the white paper

aws workshop studio

Getting Started with Cloud-Native Automotive Software Development

Getting Started
1. Create a Build Pipeline
2. Deploy a Yocto-Linux AMI on EC2
3. Trigger Build Pipeline
4. Deploy Container Image
5. Evaluate Application
6. [Optional] Scale with AWS Batch
7. Deploy on Physical Hardware
Summary

Getting Started with Cloud-Native Automotive Software Development

Getting Started with Cloud-Native Automotive Software Development

Welcome to the cloud-native automotive software development workshop created by **Arm** and **Amazon Web Services (AWS)**!

This workshop will introduce a novel automotive-native software development infrastructure able to execute the same containerized workload - with **environmental parity** - in all the targeted compute elements: an AWS EC2 Graviton2 instance, a Raspberry Pi, and an AVA Developer Platform.

As defined by Kevin Hoffman in his book:

> *The purpose of [...]* ***environment parity*** *is to give your team and your entire organization the confidence that* ***the application will work everywhere****.*

This is a major element in achieving the objective of *automotive* cloud-native software-defined vehicles.

You will learn how to utilize AWS to create a CI/CD pipeline that builds, containerizes, evaluates, and enables deployment - at scale in the cloud and on embedded devices - of a perception network, YOLO. This network is used as a stand-in for any automotive application workload to demonstrate the design paradigm. The specific version of YOLO used in this workshop is the one implemented in the Autoware stack (YOLOv2-Tiny), running on Ubuntu Linux 20.04.

The full System Under Test (SUT) stack will include the Operating System (a **Yocto-Linux** distribution) and Arm's Edge Workload Abstraction and

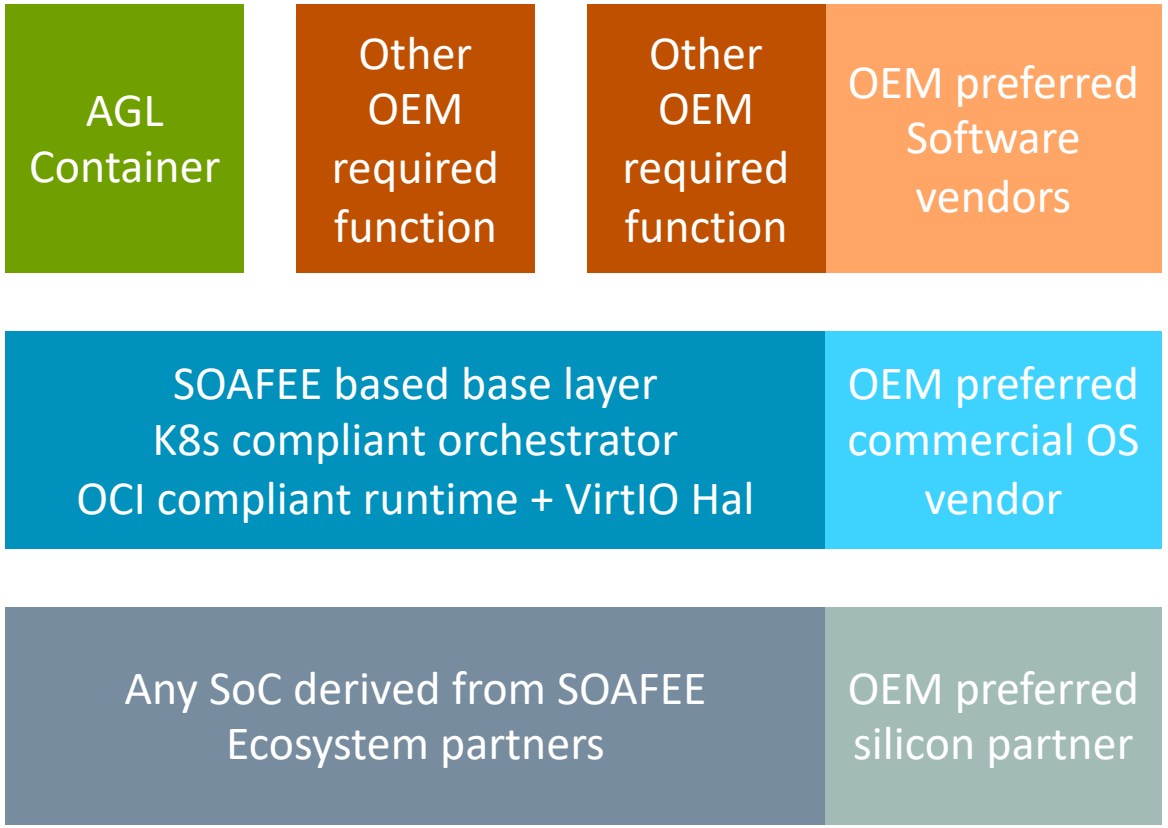
More details available at https://soafee.io/blog/

Software Defined AGL enabled by SOAFEE

Two routes forward

AGL as a SOAFEE compliant workload

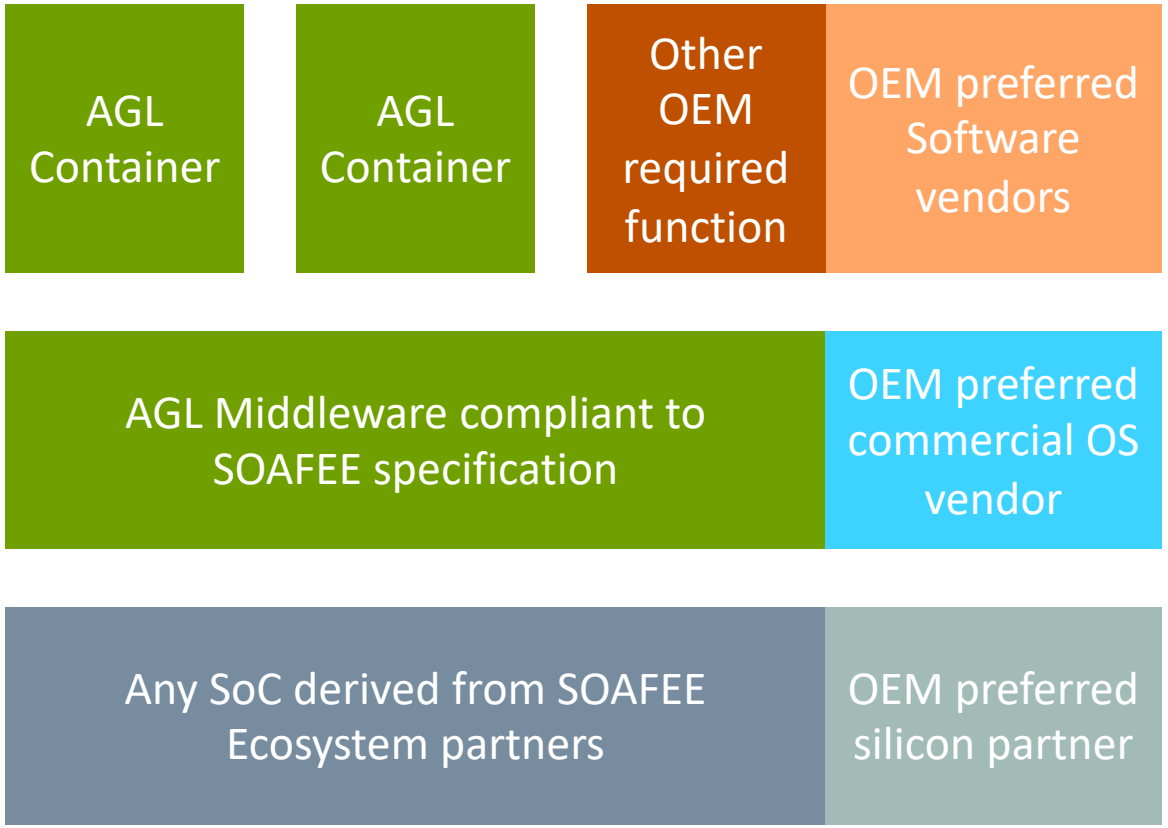
- AGL as a containerised workload
  - Makes use of SOAFEE defined portability
  - All Open Standards based (VirtIO etc)
  - Aiming for binary compatibility
- Deployable to any SOAFEE compliant distro
  - Allows OEM's to select middleware independent of workloads
  - Makes AGL more attractive to OEM's
- Eases route to adoption by OEM's
  - Not single sourcing for any component in system delivery



Two routes forward

AGL as a SOAFEE compliant middleware

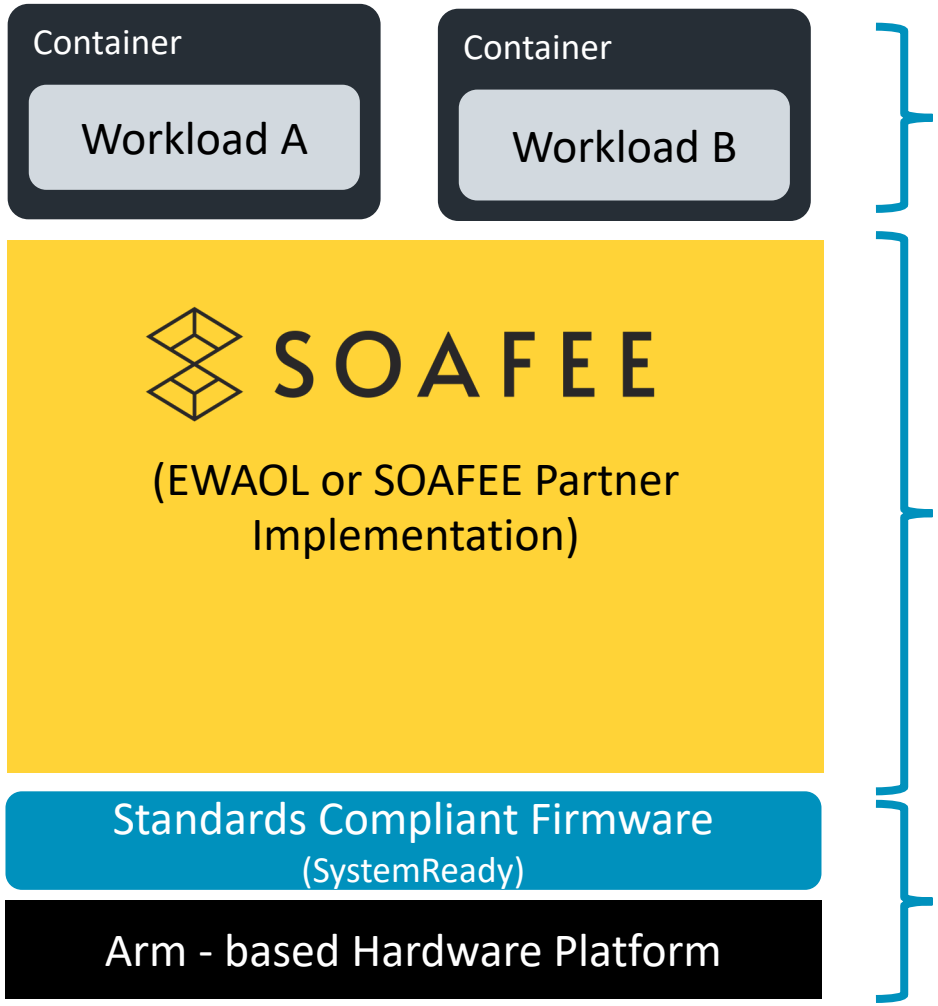
- SOAFEE is a software architecture expressed through standards
  - AGL can implement to these standards
  - Pull in the SOAFEE vision of mixed critical, real-time and safety
- Ease deployment of third party workloads into the AGL ecosystem
  - For example Autonomous/ADAS functions through OpenAD Kit
  - Enable technology enabled by the SOAFEE ecosystem to deploy to AGL



SOAFEE Blueprints with Project Apollo

Moonshot soon to be renamed – possibly Apollo

SOAFEE Blueprint



Solution SW Elements

Application Specific Workload(s)

Base SW Stack

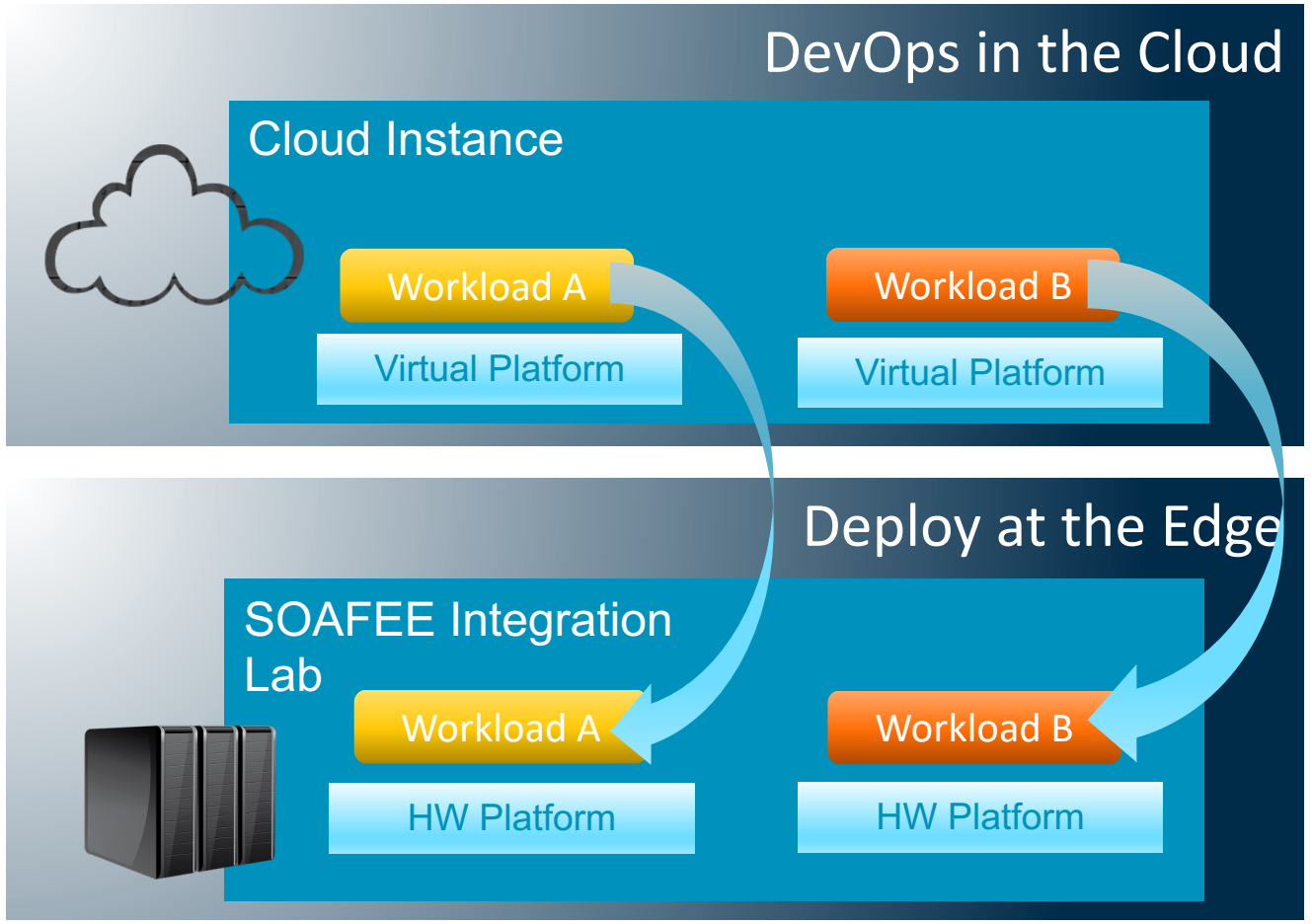
- Cloud Native
- Containerized Workloads
- Orchestration
- Tooling
- Validated on Standard Firmware

Compute Platform

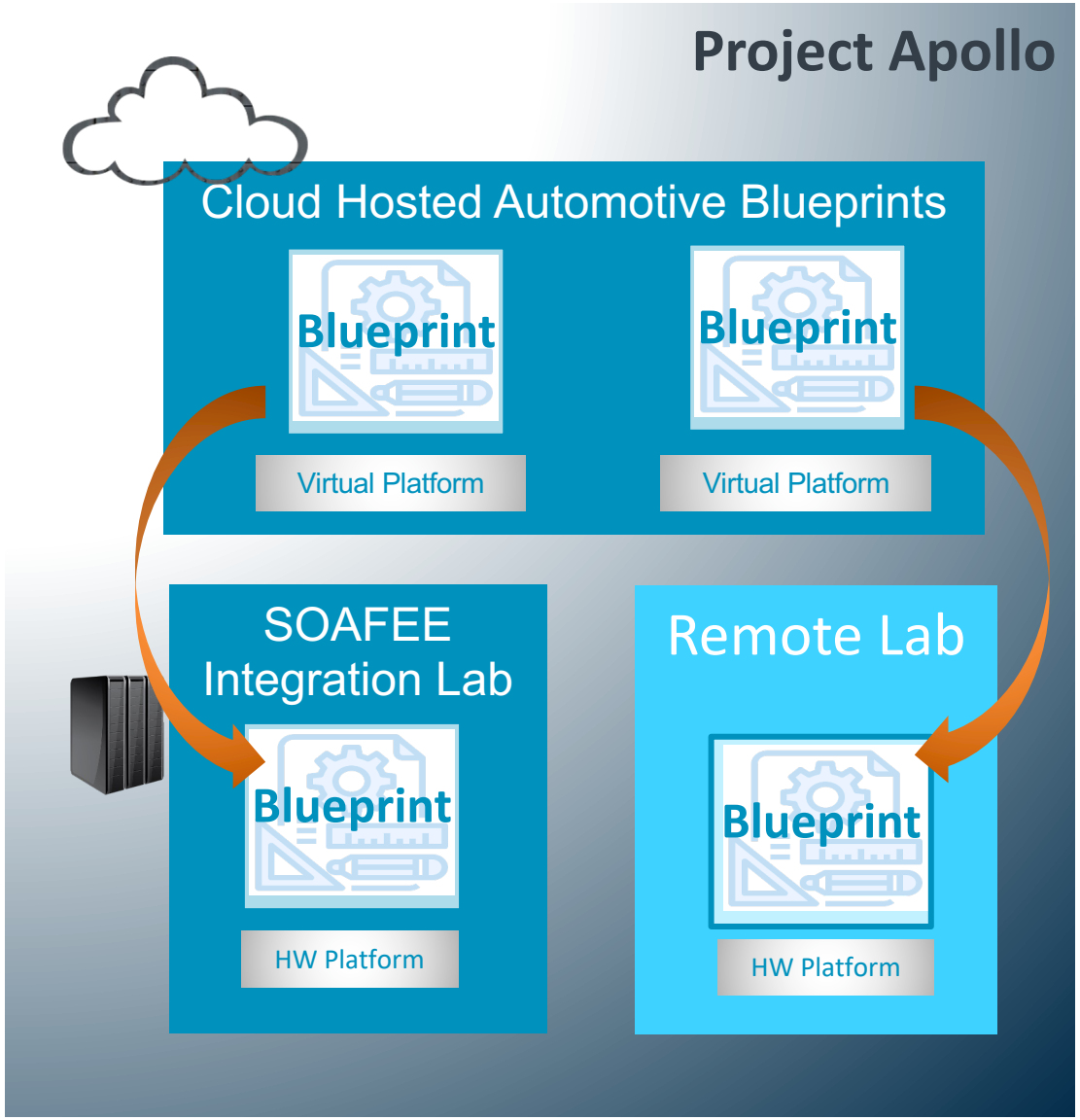
- An opportunity to showcase new platforms

Project Apollo (f.k.a. Moonshot)

- DevOps in the Cloud
  - Native software development
  - Software integration and testing
  - Automated software testing
  - Enabled by existing standards, e.g., VirtIO, etc.
- Deployment at the Edge
  - Container orchestration
  - Real-time workloads
  - Safety-critical workloads
  - Use de-facto standards such as Kubernetes

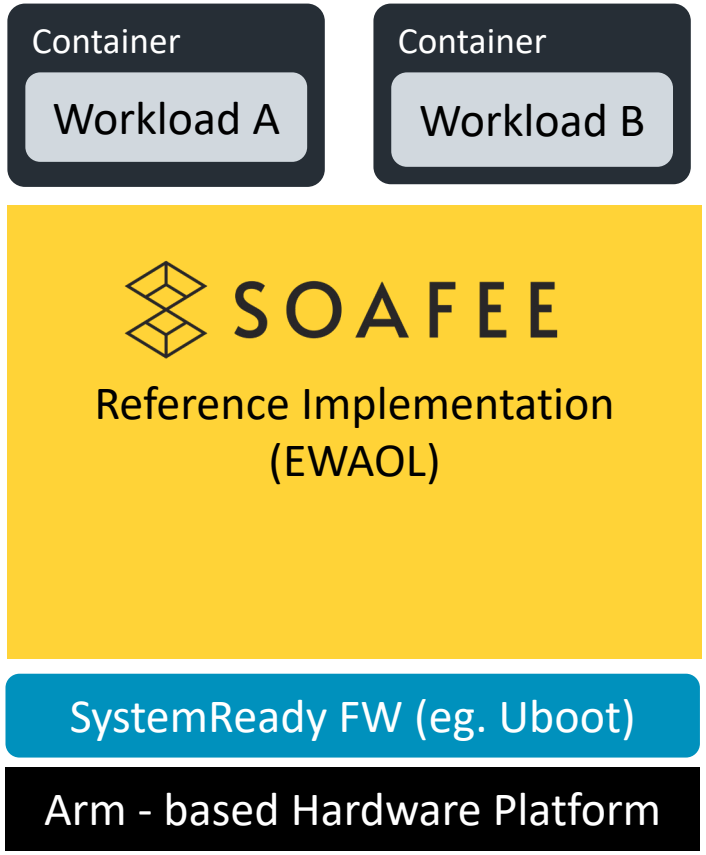


SOAFEE Integration Lab



SOAFEE Integration Lab

- Provide reference implementation feature development (part of the reference implementation WG)
- Provide needed infrastructure to regression test the ref. implementation and ensure a stable platform that current and future Blueprints can use
- Validate Blueprint workloads against targeted HW platforms
- Prove the DevOps Workflow by leveraging the Blueprints as test and validation scenarios



arm

Thank You
Danke
Gracias
Grazie
谢谢
ありがとう
Asante
Merci
감사합니다
धन्यवाद
Kiitos
شكرًا
ধন্যবাদ
תודה