Holger Dormann, Kaloyan Rusev
July 06, 2023
Automotive HPCs vs. cloud native

Why existing cloud-native tools do not fit to automotive HPCs

Containerized applications provide advantages like

• Increased portability allowing separate updates and better reuse
• Increased security through better encapsulation
• Development and test in the cloud

But

• Existing tools like Kubernetes cannot be used in automotive HPCs
• Designed for huge distributed data centers
• Eventual consistency vs. real-time and safety
• Neither programming languages like Go nor development process are suited for embedded automotive requirements
• Performance and resource usage not suited for in-vehicle platforms
• Big share of functionality not required in automotive
Eclipse Ankaios

- Developed from ground up for automotive HPCs in Rust
- Slim yet powerful solution to manage containerized applications
- Supports various container runtimes like Podman and also native applications
- Supports Automotive SPICE process with requirement tracing
- Multiple nodes and VMs are managed by Ankaios with a single unique API
- Provides a central place to manage automotive applications
- Existing communication frameworks like SOME/IP, DDS or REST API can be used with Ankaios workloads as well
- Supports Kubernetes pod resources
Ankaios – high level overview

Server

Server reads a startup config file

Push config to Agents

Start app

simple app

security app

Start app with dependency

Start app by another app

health manager

ADAS app

Agent A

Agent N

Connect to server

depends on

starts

© Elektrobit 2023 | Public

July 06, 2023
Ankaios – deployment (example)
Ankaios – high level architecture

Ankaios Server

Ankaios Agent

Runtime adapter

Container engine / OS

Workloads

Persistency

Init system

Control Interface

Cloud Connector

Developer

ank (Ankaios CLI)

1..n

Backend

<<extern>>

<<workload>>