ECLIPSE CON2023

# Developing a Fully Portable Containerized Edge Architecture
for Industry-wide Adoption

Carlton Bale	Director - Digital Product Planning	Cummins Inc.
Dr. Martin Brown	Software Architect - Edge IoT	Cummins Inc.
Ankit Tarkas	Manager - Edge IoT Device Software	Cummins Inc.

Agenda

TLDR; WHAT YOU WILL HEAR TODAY

1

Production-Level Architecture

We've developed a portable, reusable, containerized edge solution for commercial vehicle industry adoption

2

Benefits

1. Reduced development costs
2. Faster time to market
3. FOSS contribution design

3

Proposed Contributions

1. Standard interface to CAN
2. Standard CAN Security

COMMERCIAL VEHICLE INDUSTRY OVERVIEW

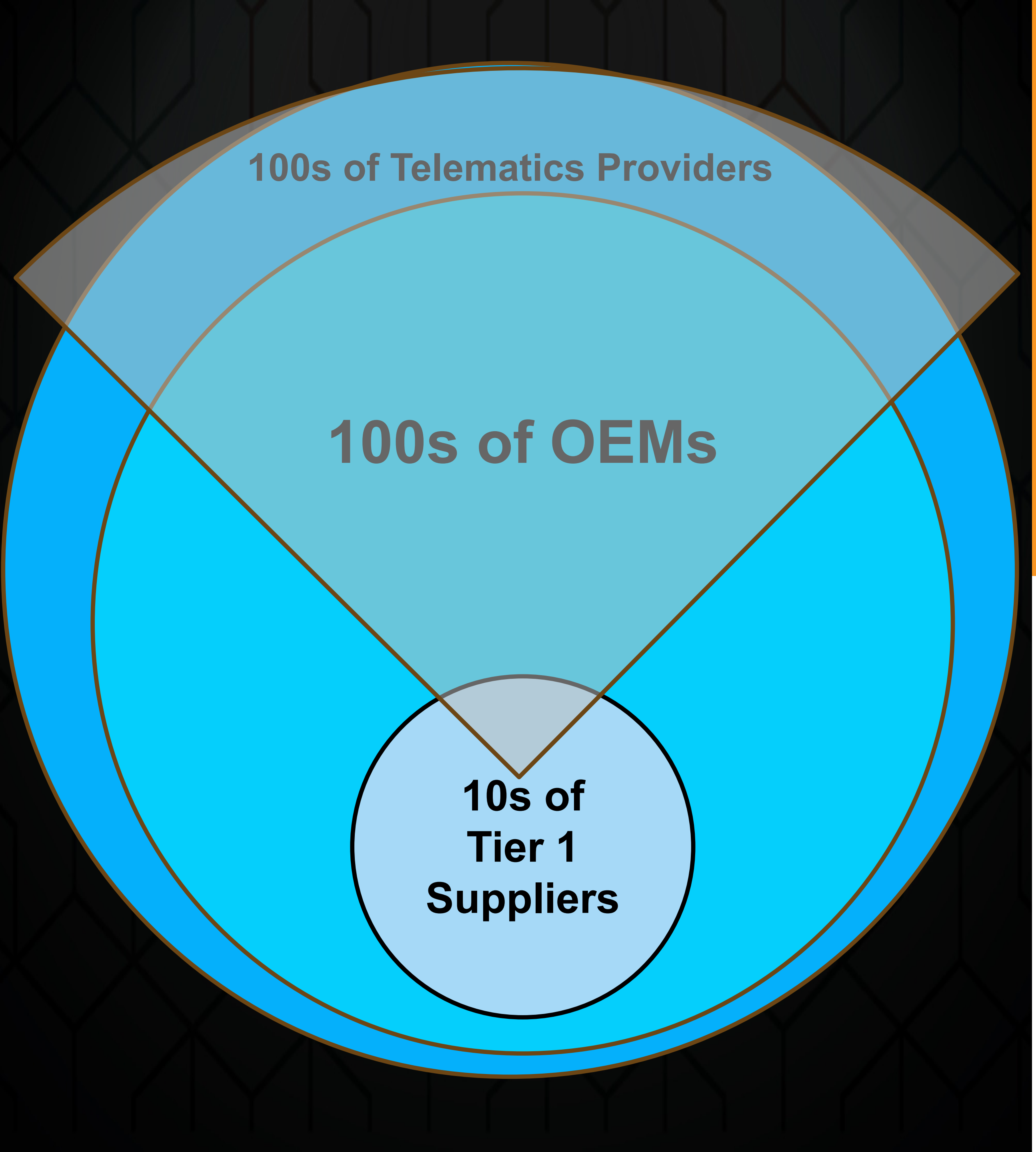
$32B USD Annual Revenue	**73,000** Employees	**52** Countries

Early Adoption of Telematics (1990s)

100s of OEMs

Low Volumes
(vs. passenger car)

Higher Variation
&
Smaller Scale



No Common Methodology
for Telematics Applications

Extremely High Implementation Costs
for Telematics

No Standard for CAN Access or Security

Extremely Long Times-to-Market
for Telematics

BUILDING CONTAINERIZED SOFTWARE PLATFORM FOR CUMMINS

KEY CONSIDERATIONS

EDGE DEVICE WITH LOW HARDWARE FOOTPRINT

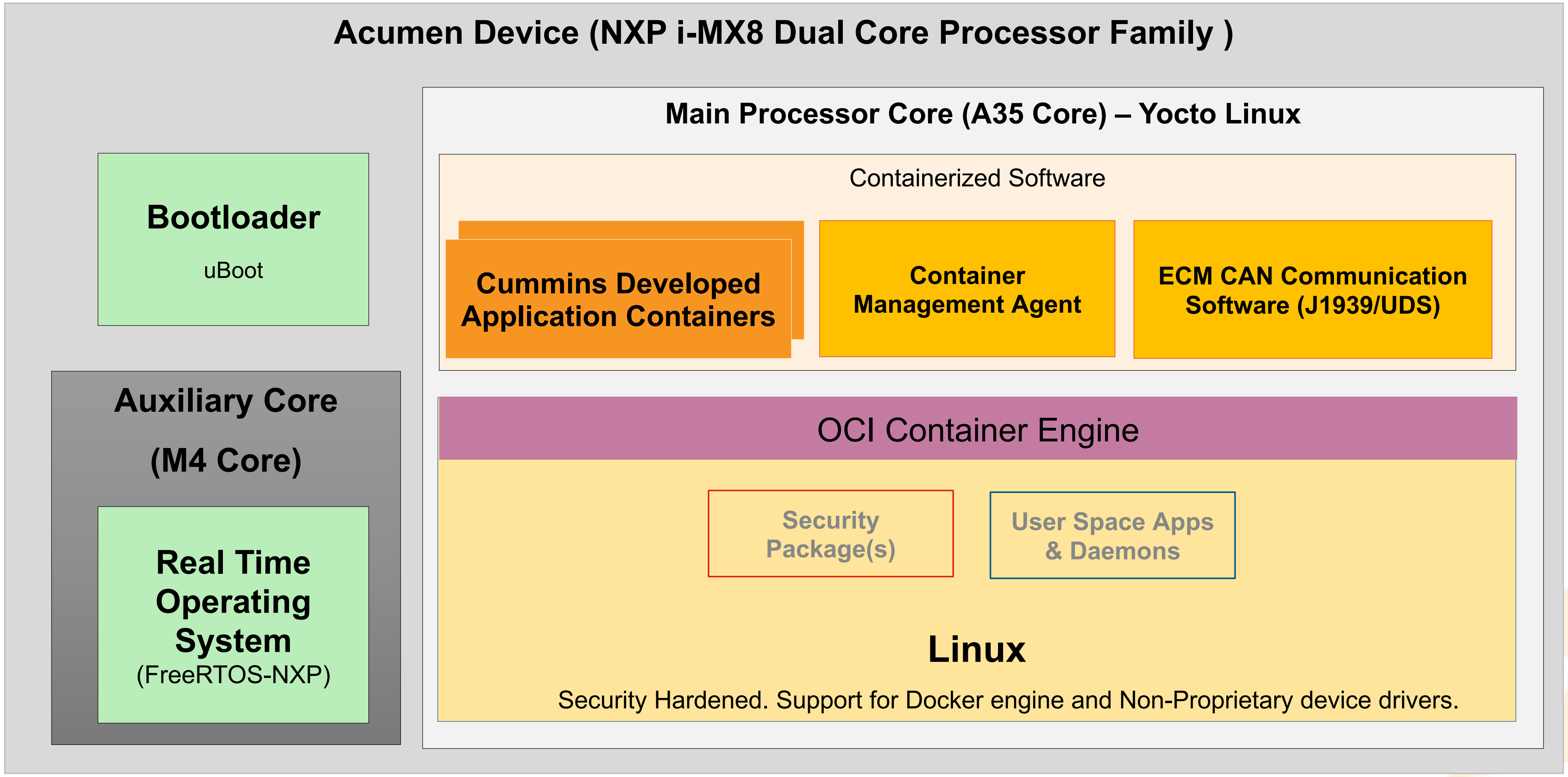
Processor	iMX8 Dual Core
RAM	1 GB
Flash	4 GB
Interfaces	CAN, Ethernet

HANDLES DEPENDENCIES FOR EASY INTEGRATION

UPDATES SOFTWARE APPLICATIONS INDEPENDENTLY

STANDARDIZED DATA EXCHANGE

CUMMINS SOFTWARE ARCHITECTURE – BIRD'S EYE VIEW



Acumen Device (NXP i-MX8 Dual Core Processor Family)
Bootloader
uBoot
Auxiliary Core
(M4 Core)
Real Time Operating System
(FreeRTOS-NXP)
Main Processor Core (A35 Core) – Yocto Linux
Containerized Software
Cummins Developed Application Containers
Container Management Agent
ECM CAN Communication Software (J1939/UDS)
OCI Container Engine
Security Package(s)
User Space Apps & Daemons
Linux
Security Hardened. Support for Docker engine and Non-Proprietary device drivers.

KEY DECISIONS IN OPTIMIZING CONTAINERS FOR THE EDGE

Lightweight OCI container orchestrator for the edge

Compact OCI images minimizing flash storage footprint

Programming Language minimizing CPU and RAM footprint

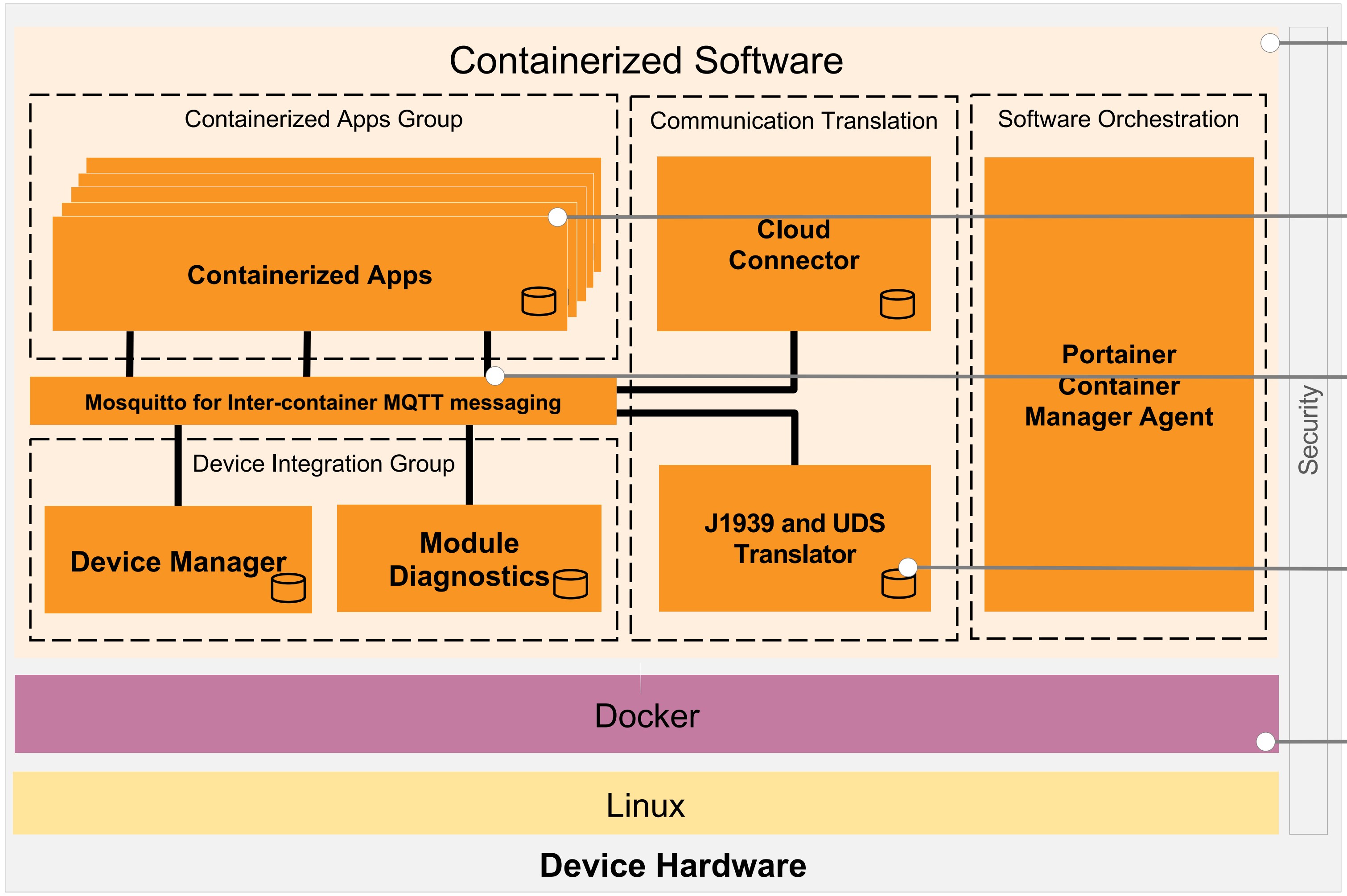
Microservices Architecture Pattern

Efficient communication between services

Lightweight embeddable database engine

ECLIPSE CON2023

Cummins Software Architecture Detail View: Modular with inter container communication



Microservices architecture promotes separation of concerns, enabling **faster development**, reusability, portability and testability, fault isolation.

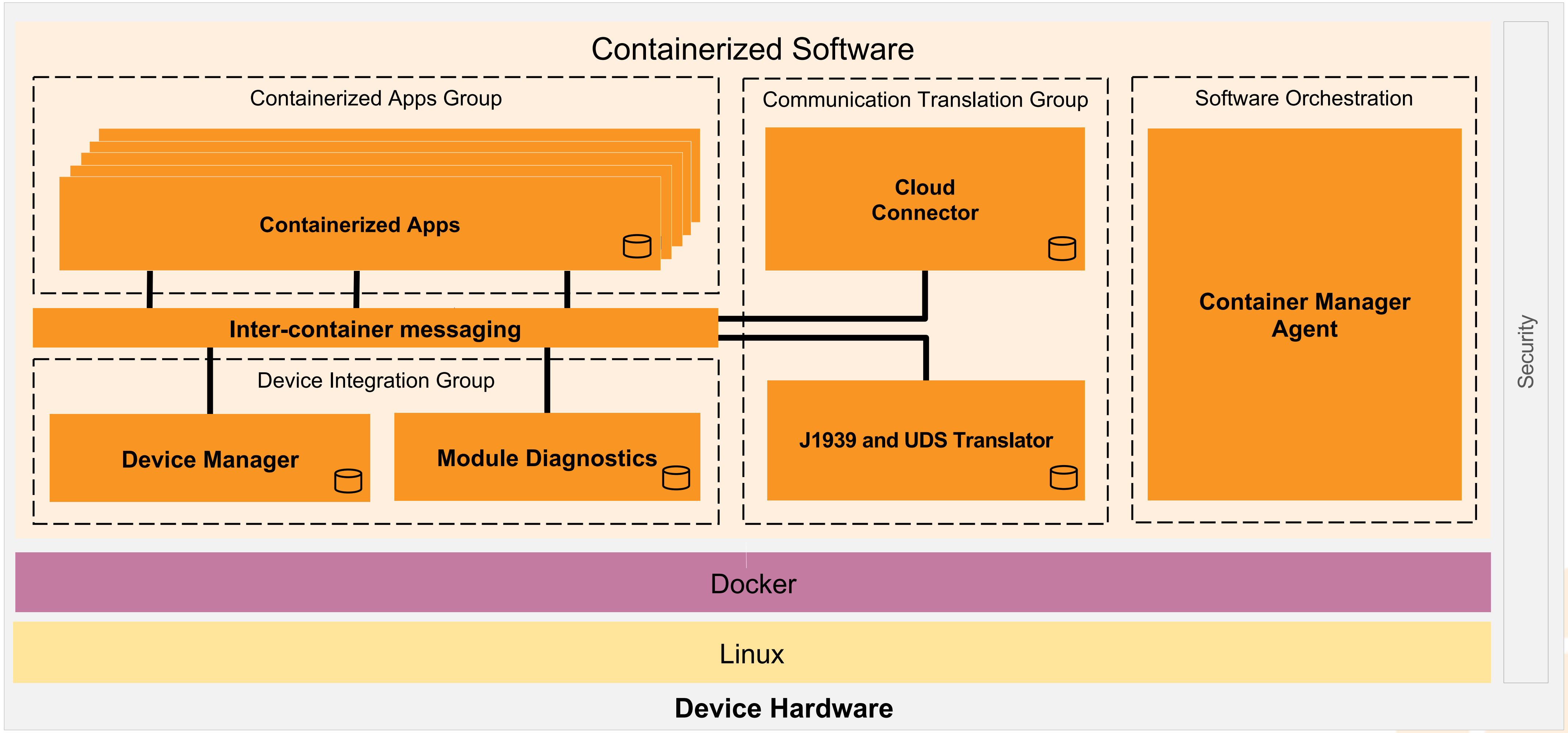
C++ applications sharing a common base Alpine Linux image leaves **low resource footprint**, saving space for edge analytics. Proofs-of-concept with **Rust** are ongoing.

Well-defined MQTT APIs formatted in JSON **speed up dev and test cycles** with the help of API mocking tools (could also benefit from code generation.)

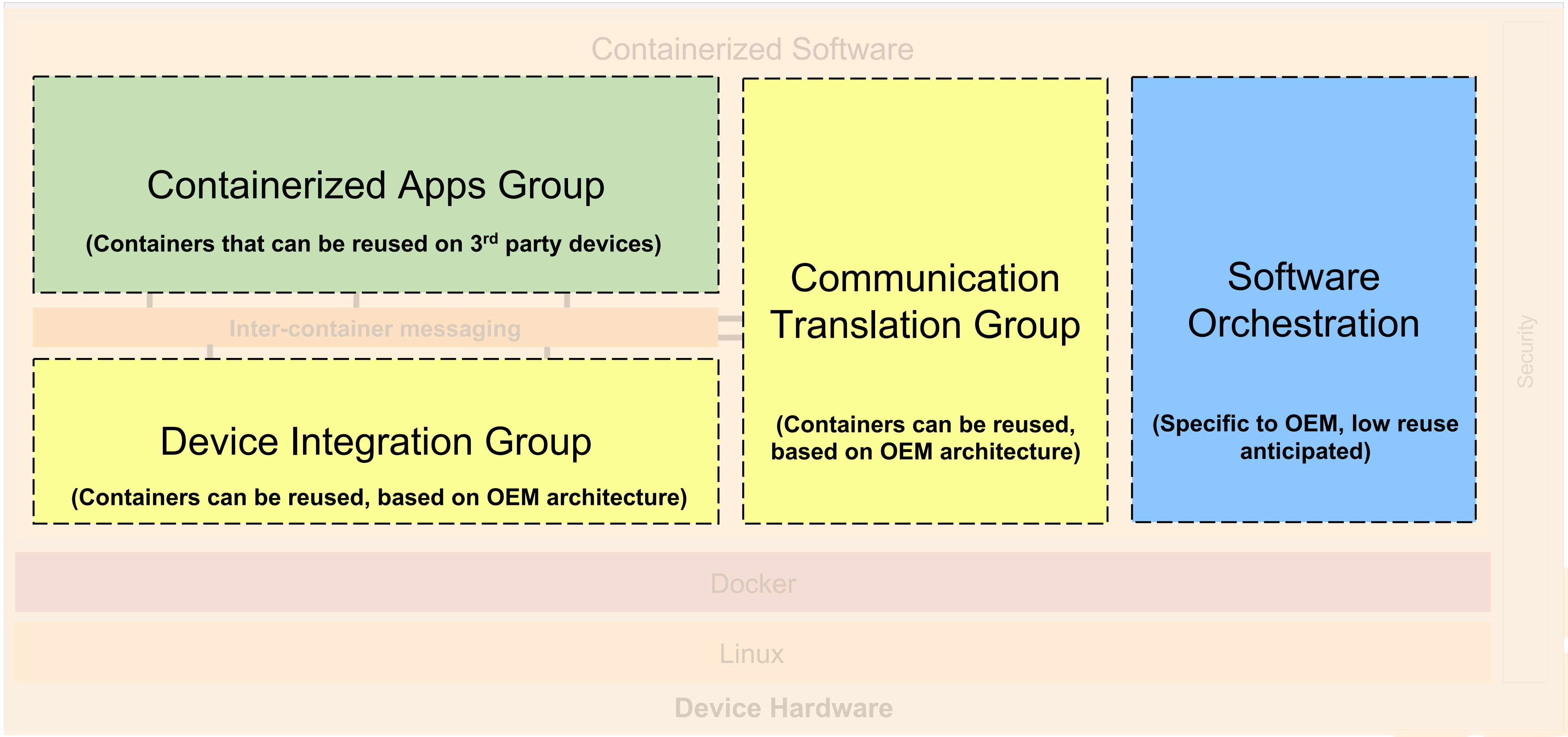
SQLite databases embedded into to applications leaves a tiny footprint, supports storing and **querying of JSON** data and **database change events**. Perfect for embedded development.

Widely adopted file format eases storage, service delivery and **platform abstraction** for easier application deployments. Exploring **Podman** and **Kubernetes** as alternatives is planned.

Cummins Software Architecture Detail View: Modular with inter container communication

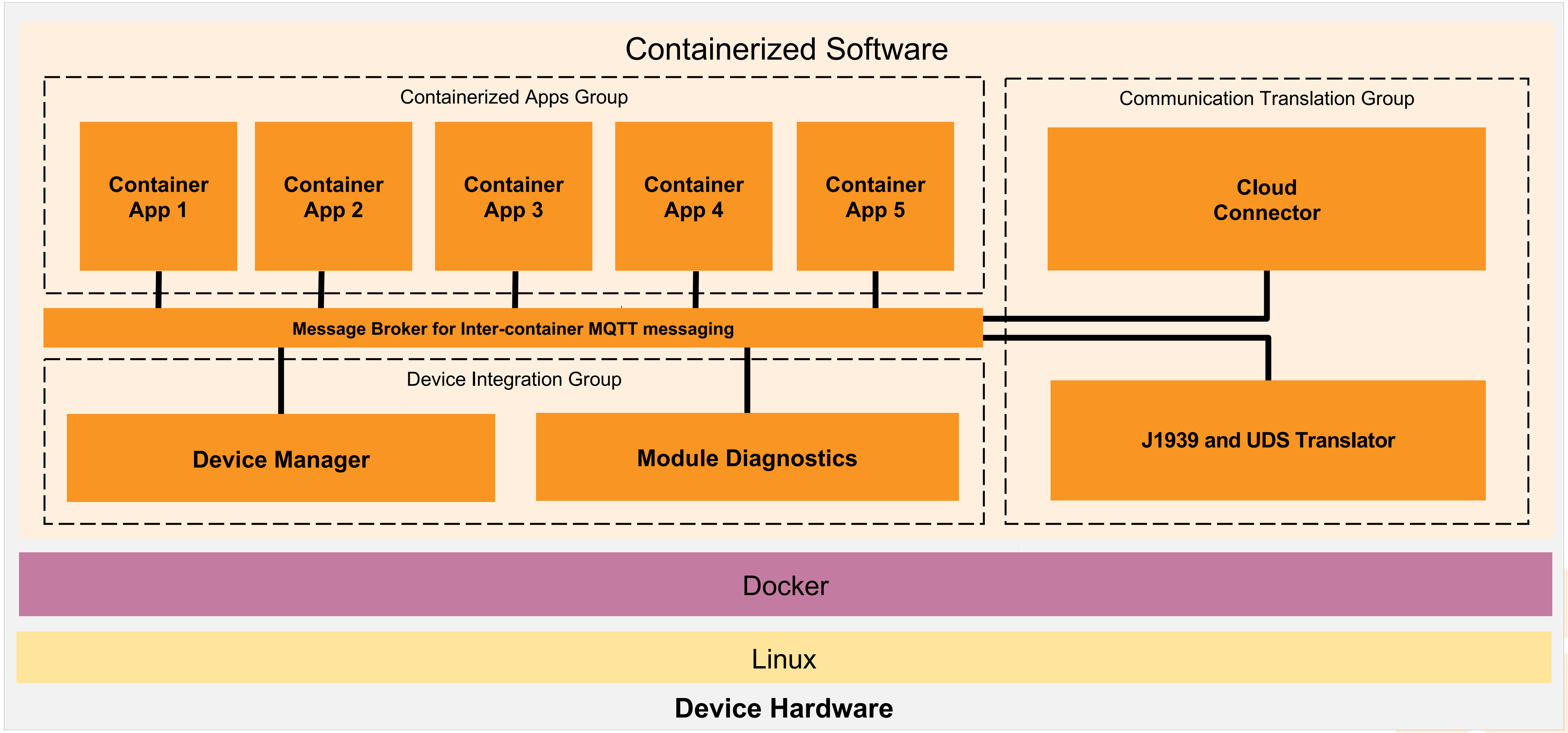


Cummins Software Architecture Detail View: Modular with inter container communication

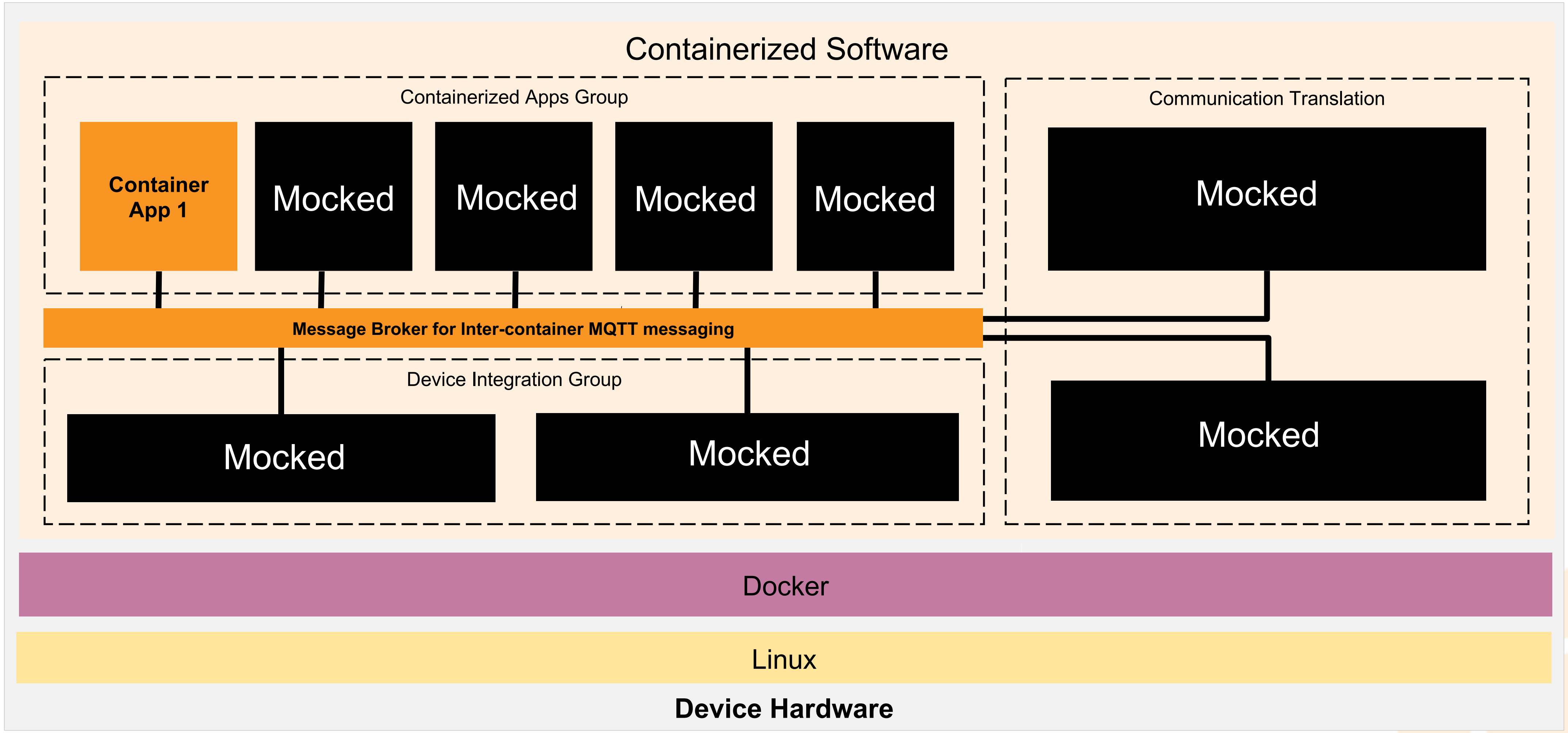


Containerized Software
Containerized Apps Group
(Containers that can be reused on 3rd party devices)
Inter-container messaging
Device Integration Group
(Containers can be reused, based on OEM architecture)
Communication Translation Group
(Containers can be reused, based on OEM architecture)
Software Orchestration
(Specific to OEM, low reuse anticipated)
Security
Docker
Linux
Device Hardware

Modular components communicating over MQTT accelerates development

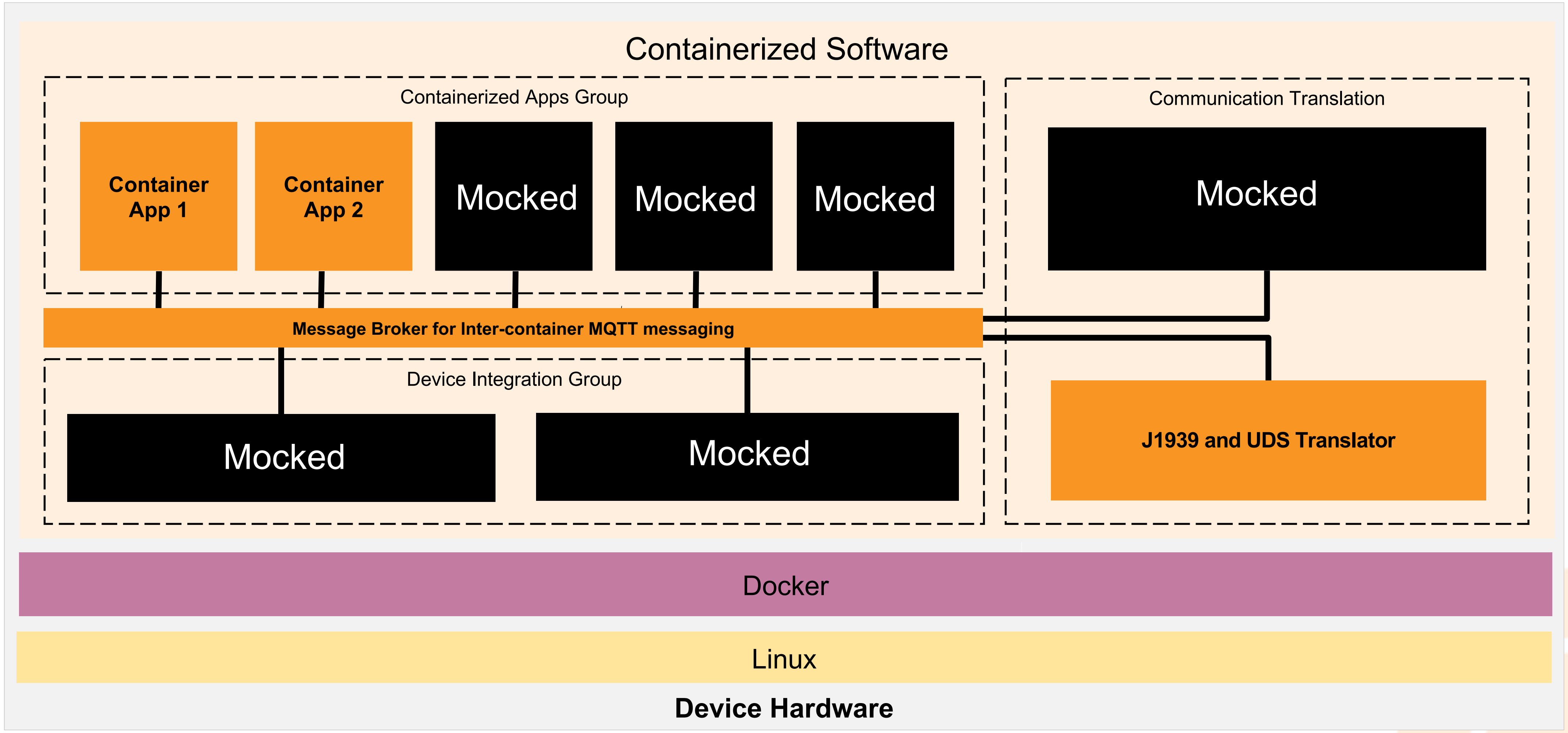


We mock the dependencies of the application-under-test until those components are ready



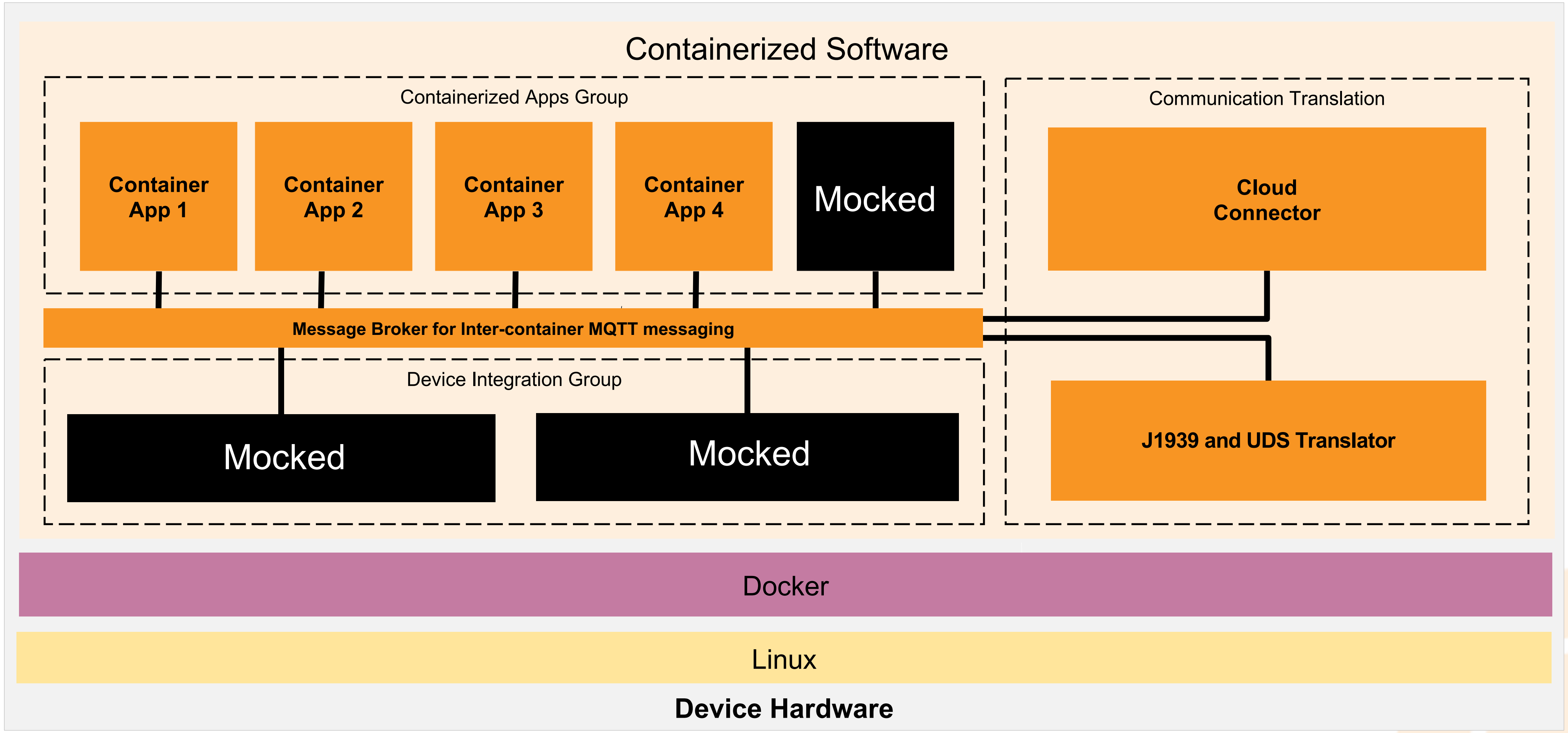
ECLIPSE CON2023

We integrate new applications and components when they become ready

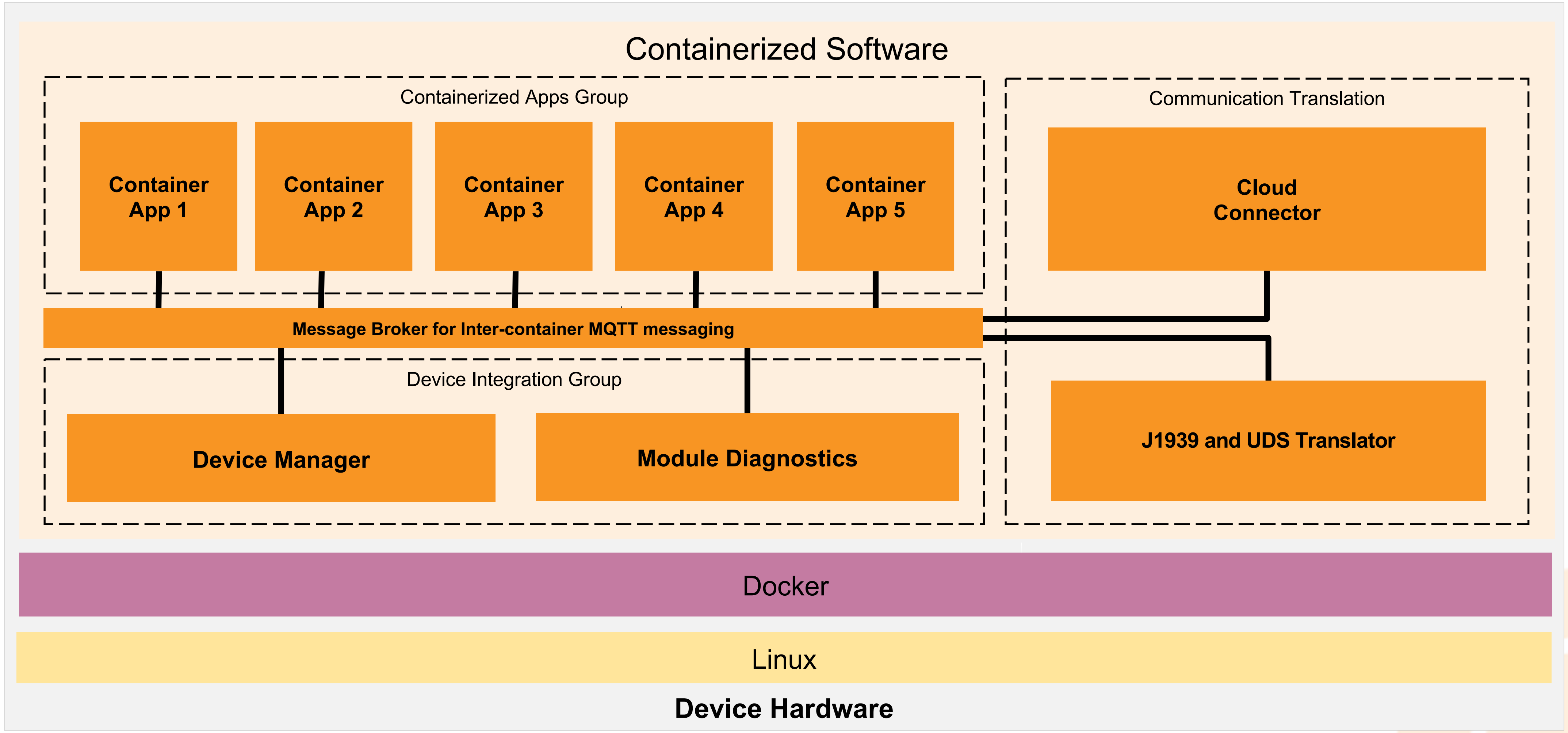


ECLIPSE CON2023

We do this iteratively

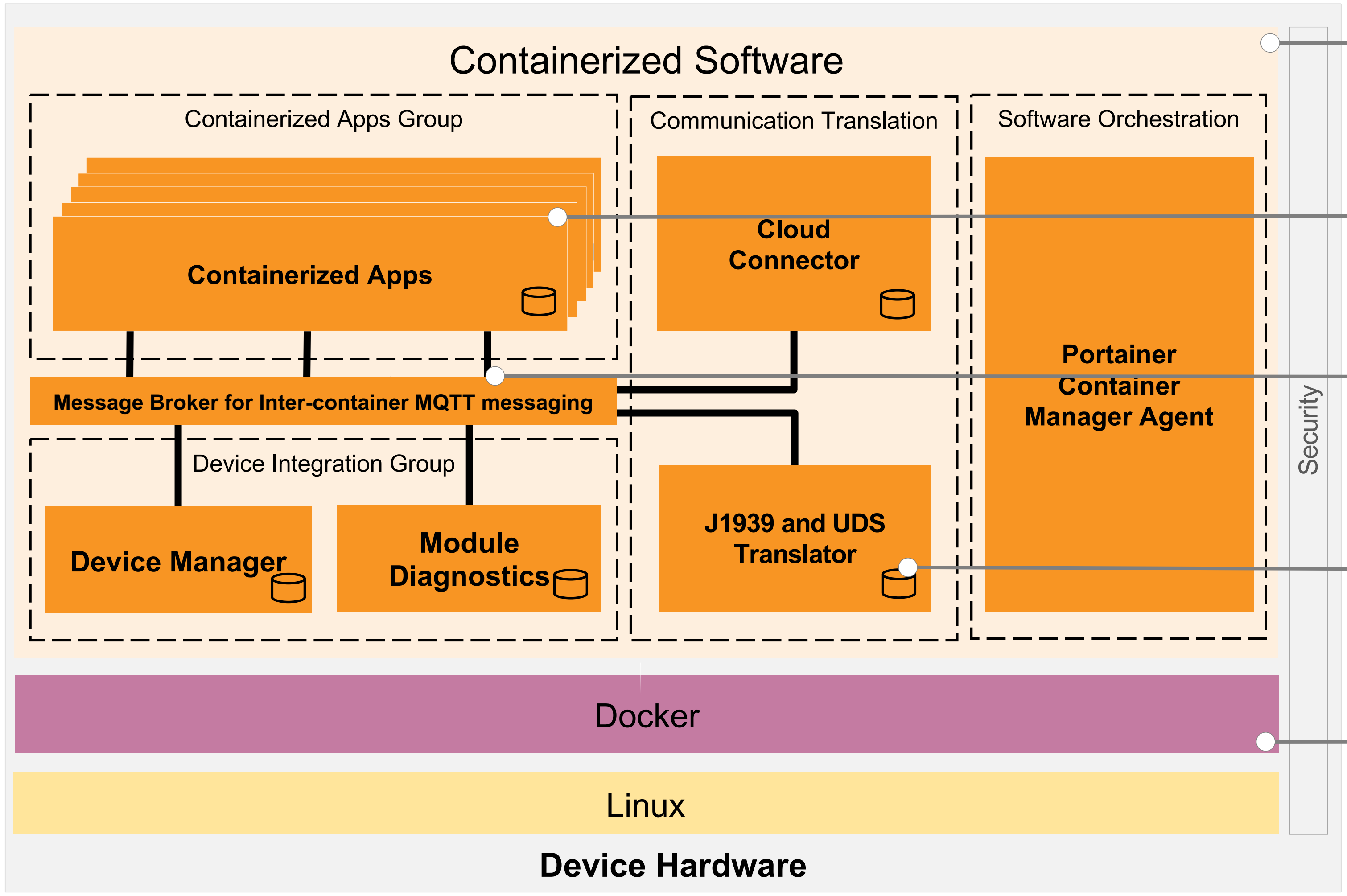


Until the complete system has been implemented



ECLIPSE CON2023

We did it! We developed a production-level portable and reusable containerized solution for commercial vehicle edge



Microservices architecture promotes separation of concerns, enabling **faster development**, reusability, portability and testability, fault isolation.

C++ applications sharing a common base Alpine Linux image leaves **low resource footprint**, saving space for edge analytics. Proofs-of-concept with **Rust** are ongoing.

Well-defined MQTT APIs formatted in JSON **speed up dev and test cycles** with the help of API mocking tools (could also benefit from code generation.)

SQLite databases embedded into to applications leaves a tiny footprint, supports storing and **querying of JSON** data and **database change events**. Perfect for embedded development.

Widely adopted file format eases storage, service delivery and **platform abstraction** for easier application deployments. Exploring **Podman** and **Kubernetes** as alternatives is planned.

ECLIPSE CON2023

KEY ARCHITECTURE DECISIONS WE MADE IN 2020. NEEDS TO BE REVISITED MOVING FORWARD.

Docker

2024: Podman
2026: Kubernetes Lite

Alpine

C++

2024: Rust

Microservices Architecture Pattern

Mosquitto
MQTT
JSON (2025: Protobuf)

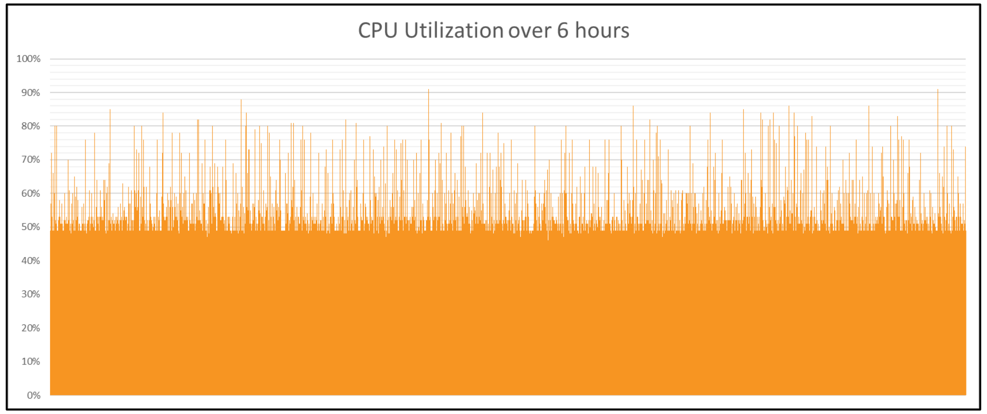
SQLite

2024: eKuiper

Resource Utilization of Acumen

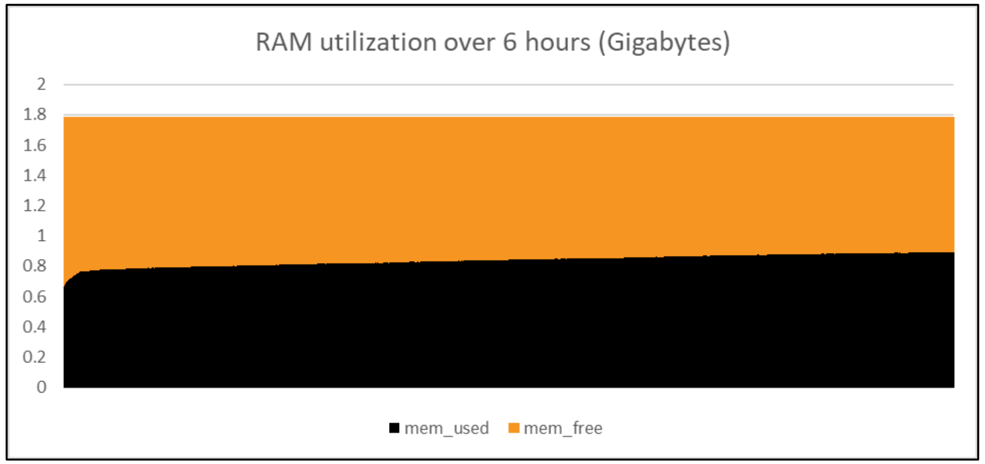
Average CPU %

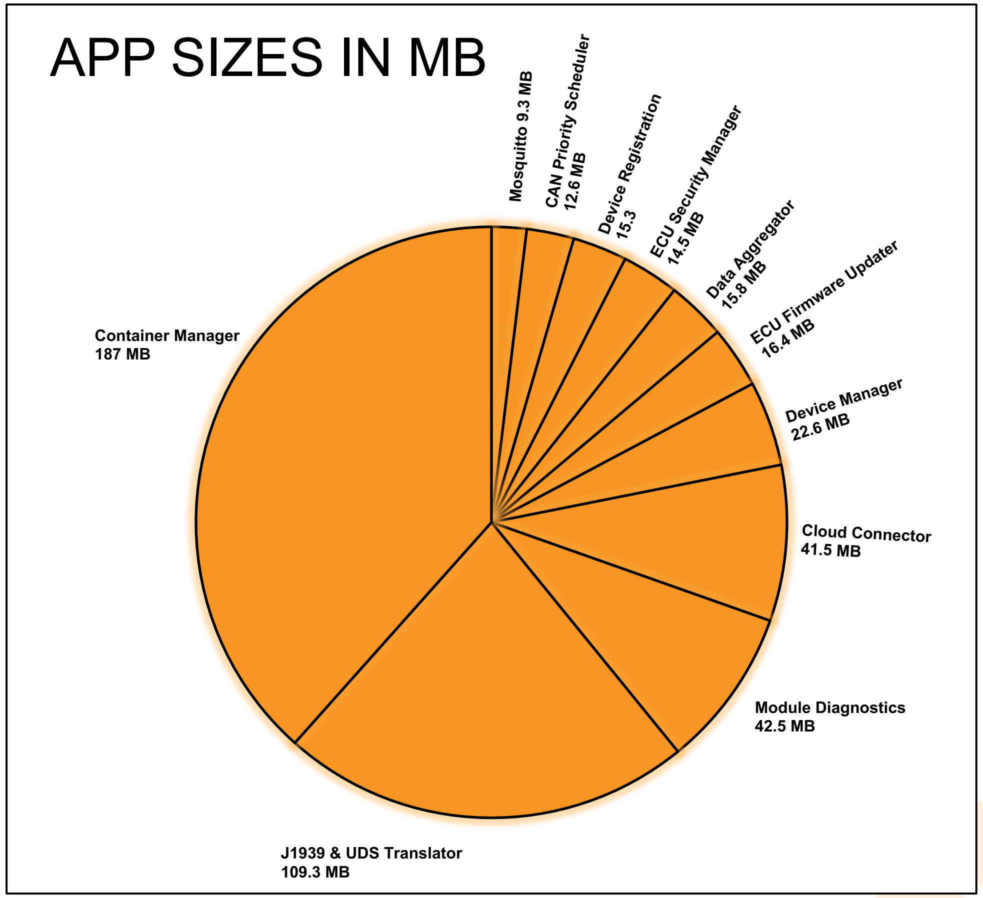
48%



Average RAM

0.8 GB





Average Container Size

44 MB

Average Cummins Container Size

20 MB

Total Virtual Size of Containers

486 MB

Total Actual Sizes of Containers when using shared Alpine image layers

452 MB

This strategy saves ~6MB per additional container

CONTRIBUTING ARCHITECTURE TO OPEN SOURCE

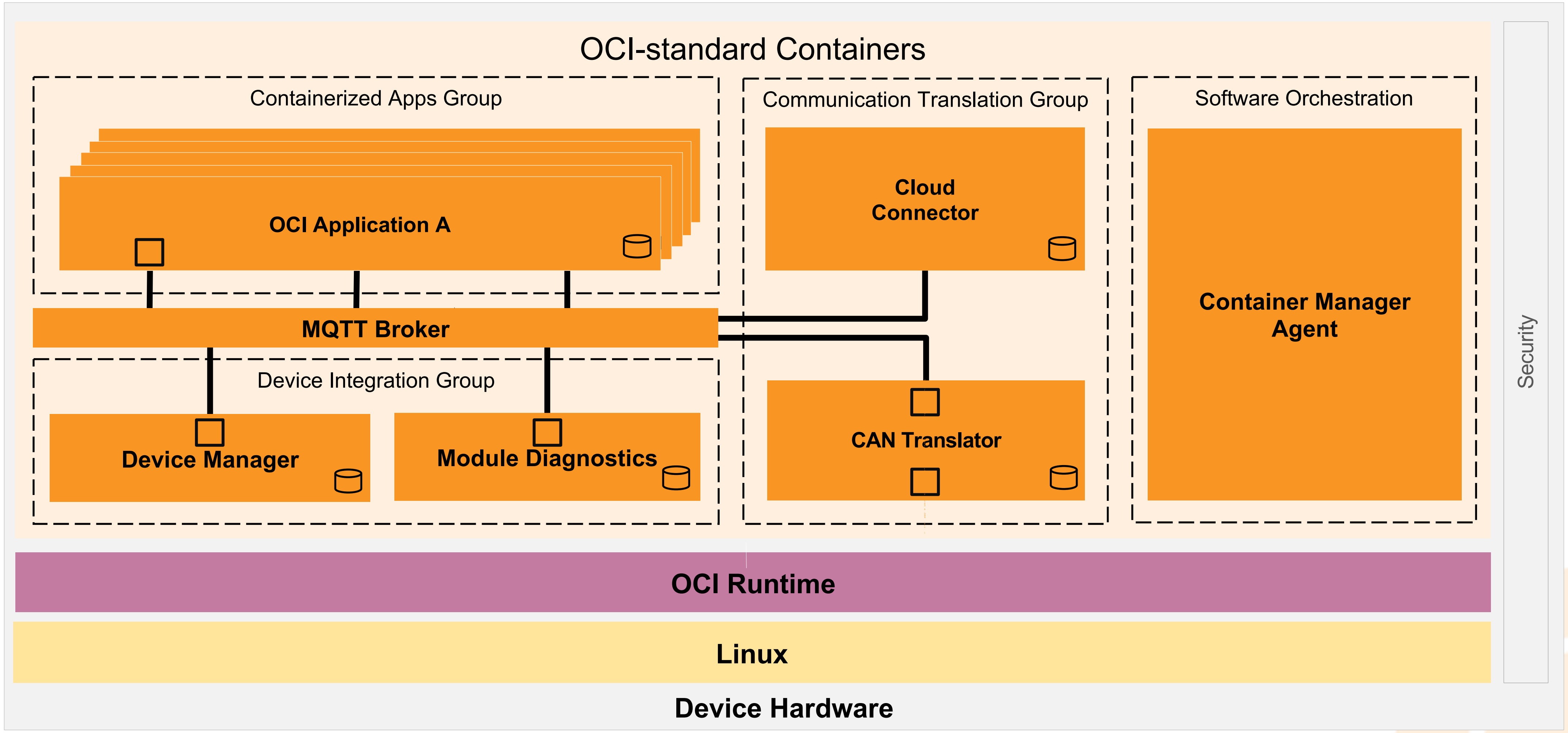
Common Application Architecture

- Cummins originally planned to contribute our entire OCI Container Architecture to Eclipse SDV
- Eclipse SDV Kanto Project is already addressing this capability
- Cummins will contribute to and help validate Kanto

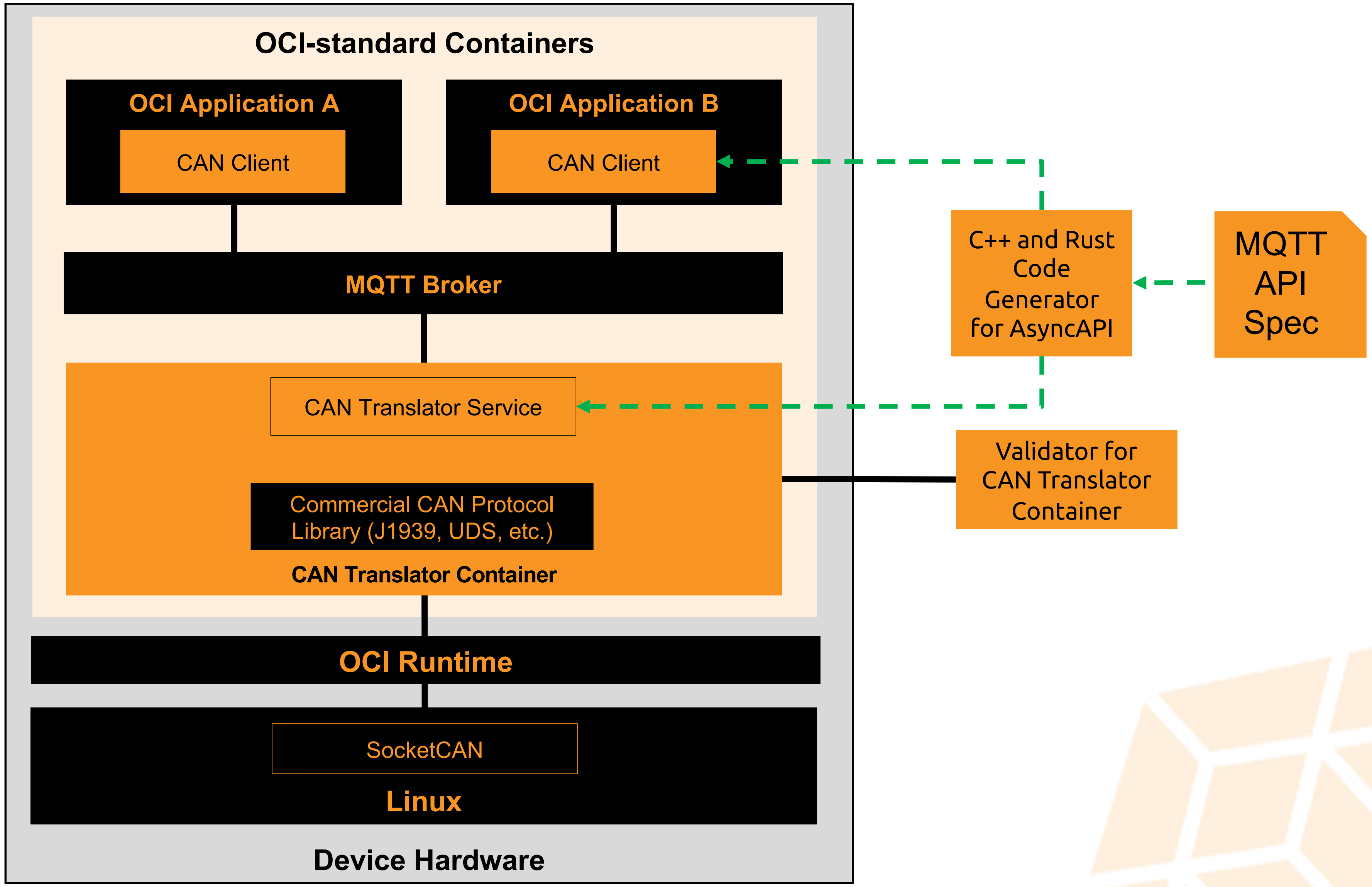
Standardizing Proprietary Interfaces: CAN Protocol Translation, CAN Security

- **Communications Abstraction Components**
  - Open-source libraries to standardized the interface to proprietary Commercial Libraries required for CAN access (J1939, UDS, etc.)
  - Inter-container communications standardization
  - Working with COVESA to take ownership of the standard
- **CAN Security Manager**
  - Per-application CAN Source Address restrictions (certificate-based authentication)
  - Per-application CAN bandwidth throttling

CONTRIBUTING ARCHITECTURE TO OPEN SOURCE



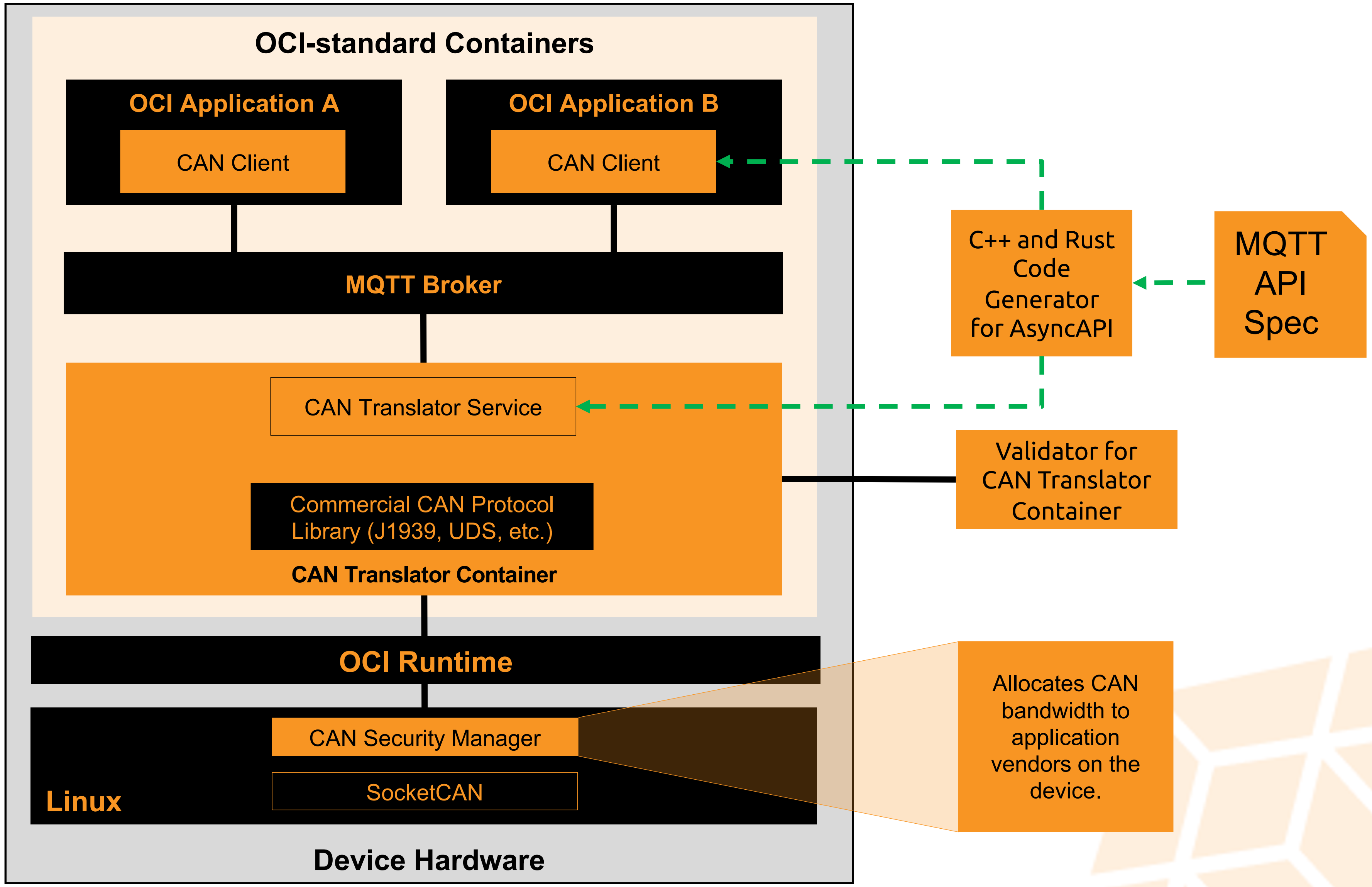
CONTRIBUTING ARCHITECTURE TO OPEN SOURCE



Legend

Bring your own
In scope

CONTRIBUTING ARCHITECTURE TO OPEN SOURCE

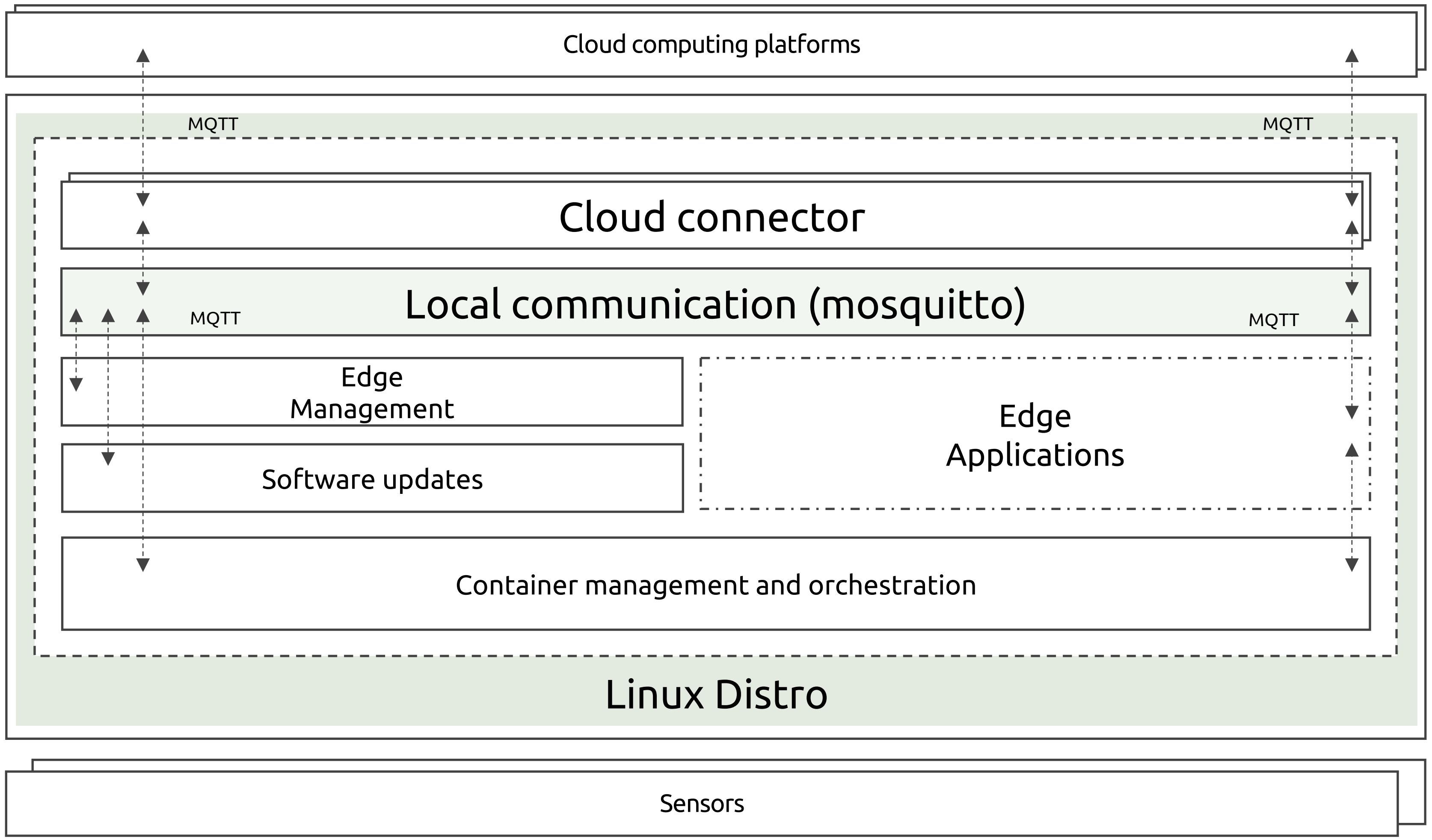


Legend

Bring your own
In scope

ECLIPSE CON2023

Cummins Proposed Integration into Eclipse Kanto (System View)

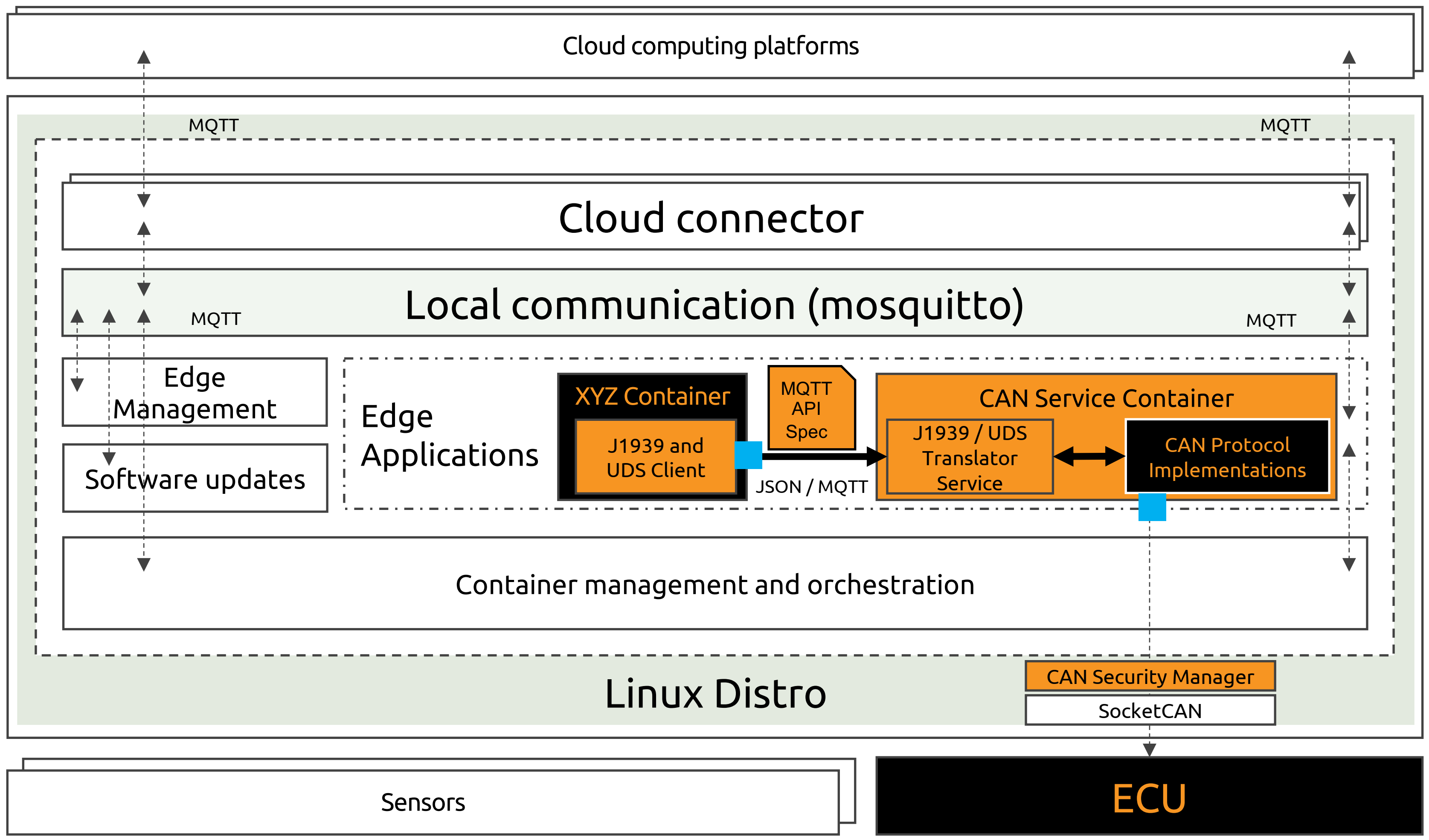


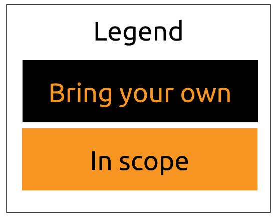
Legend

Bring your own

In scope

Cummins Proposed Integration into Eclipse Kanto (System View)





BENEFITS OF OPEN SOURCE ARCHTECTURE

Reduce Development Costs

- **25% Decrease in Program Cost despite broader scope**
  - vs. previous program with proprietary non-portable architecture

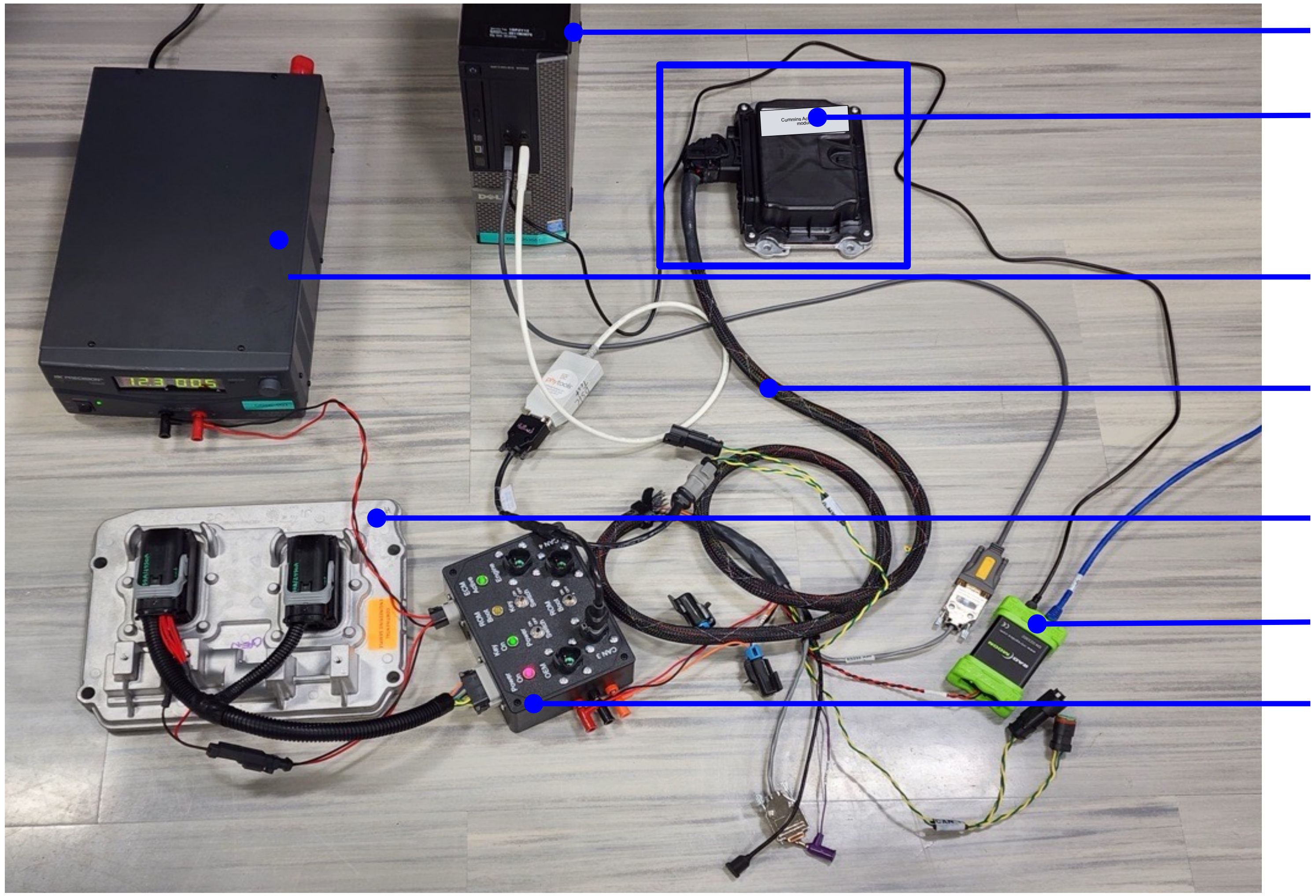
Enable Rapid Time-to-Market

- **50% Reduction in Program Development Time**

Support an Industry-Wide Ecosystem

- **Numerous industry partners collaborating on requirements, use cases, adoption, and code contribution**

Acumen Hardware Setup for Verification and Validation



Remote PC for software V&V

Acumen IoT module (engine mounted)

Power supply for powering the Electronic Control Unit (ECU)

Harness for connecting the ECU to the Acumen module

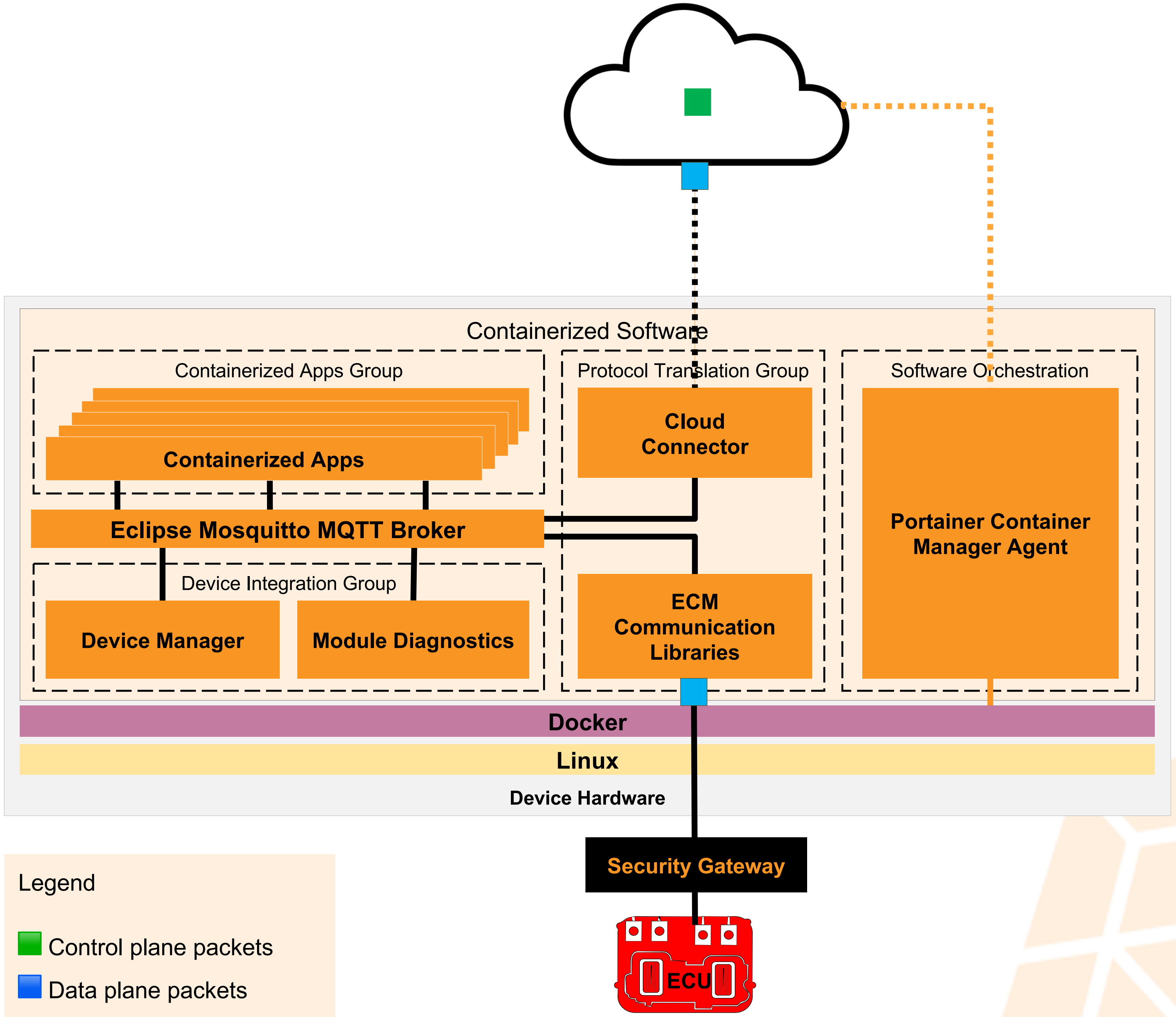
ECU

Ethernet converter to convert automotive Ethernet to standard Ethernet

ROM box for key-switch and power-switch functionality

Acumen Demo

1. Operator sends deployment command from the cloud over the control plane to the container manager agent on the device
2. Container Manager Agent and Docker work together to bring up the containers
3. Containers start comms with each other over MQTT, and with the ECM and Cloud via protocol translation services



Legend

🟩 Control plane packets

🟦 Data plane packets

Thank you!

Carlton Bale	Director - Digital Product Planning
Dr. Martin Brown	Software Architect - Edge IoT
Ankit Tarkas	Manager - Edge IoT Device Software

ECLIPSE CON2023