Getting QEMU ready for the Automotive Industry

Sergio López Pascual

Senior Principal Software Engineer

Automotive Team

Red Hat

About the Automotive Industry

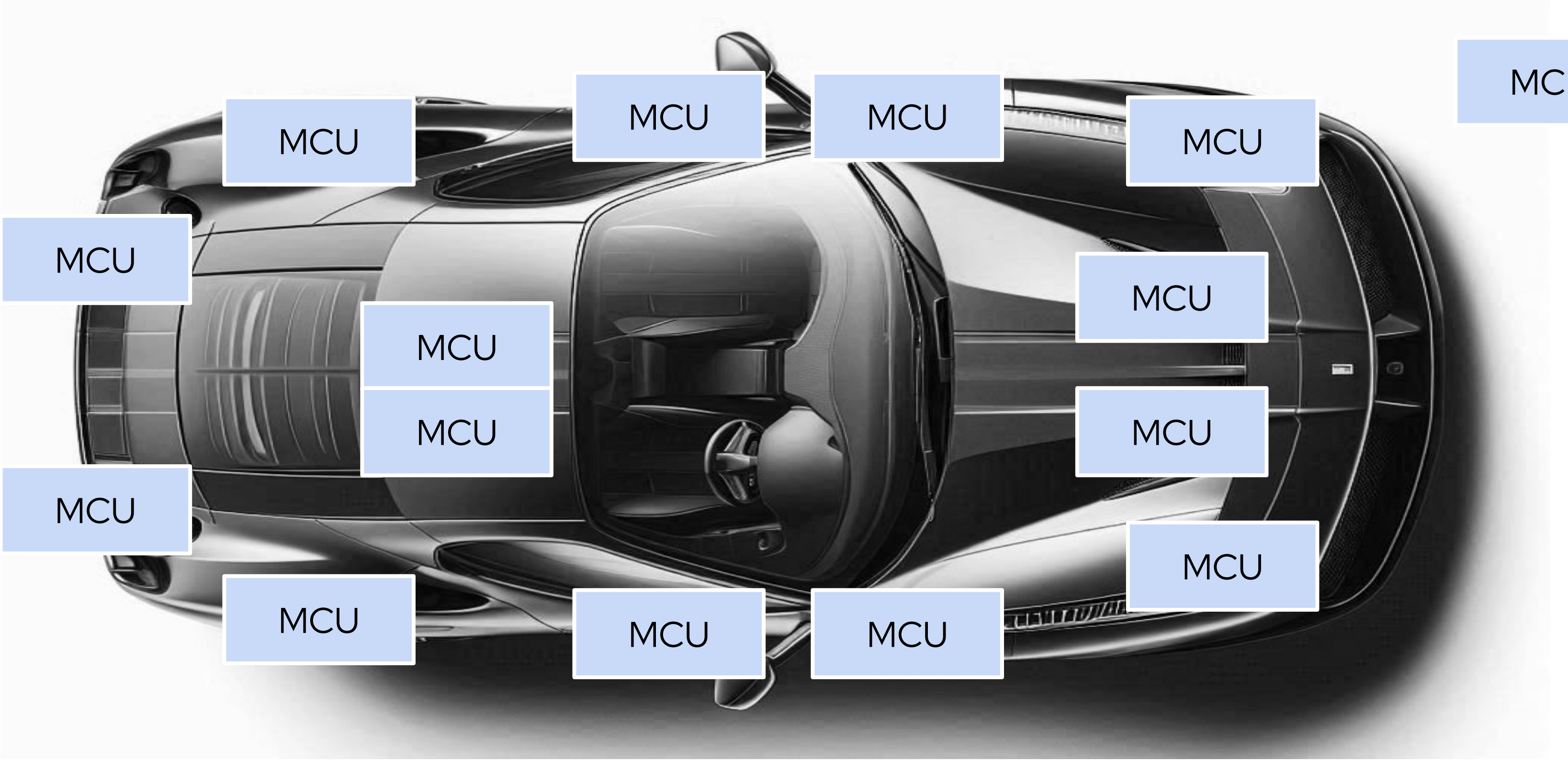
So, what’s so special about it?

We need to answer two questions:

- Why would a car need any form of Virtualization?
- How is the Automotive Industry different from other industries already using Virtualization?

MCU-based design

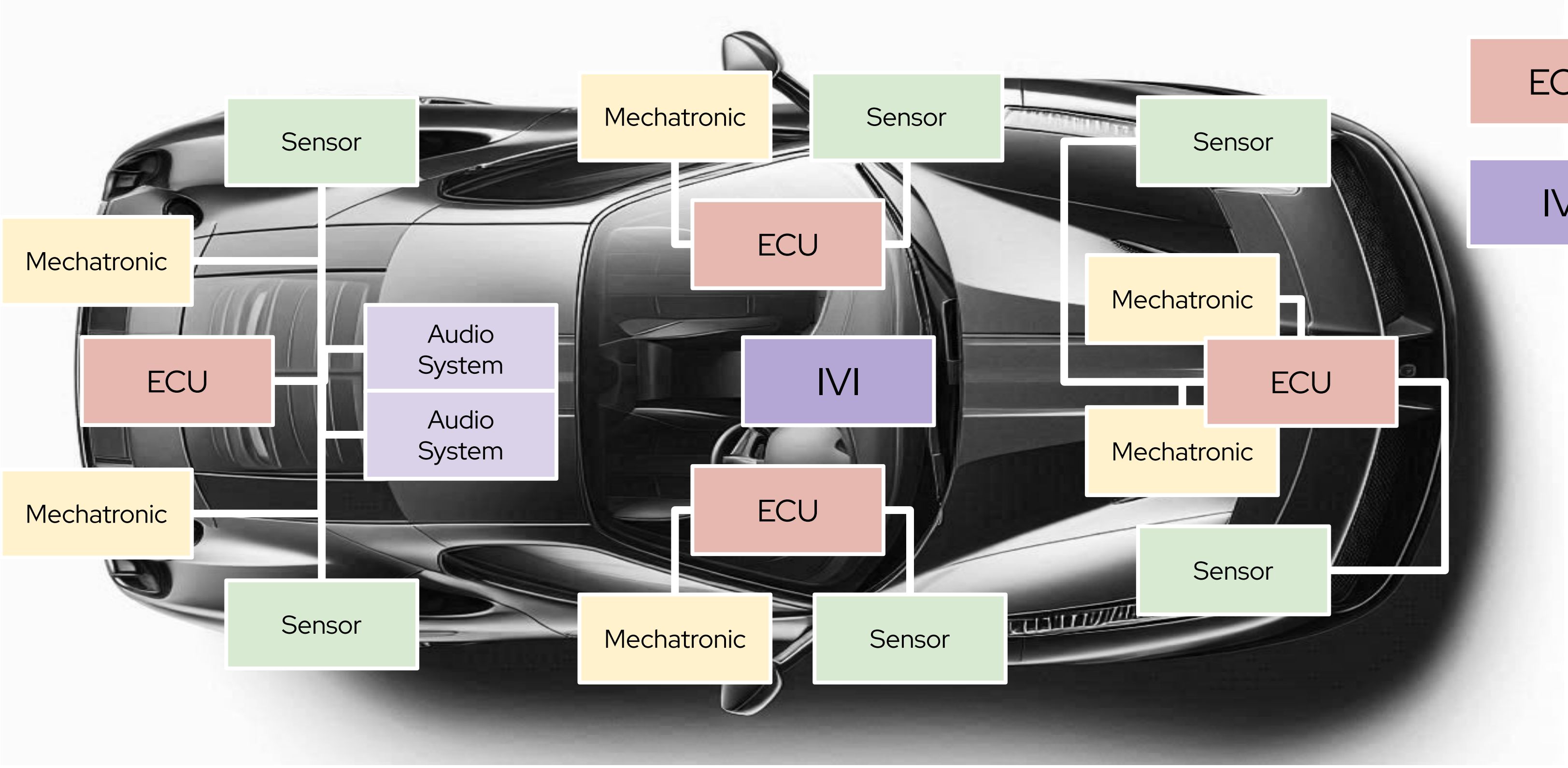
Independent, static systems



MCU Microcontroller Unit

ECU-based design

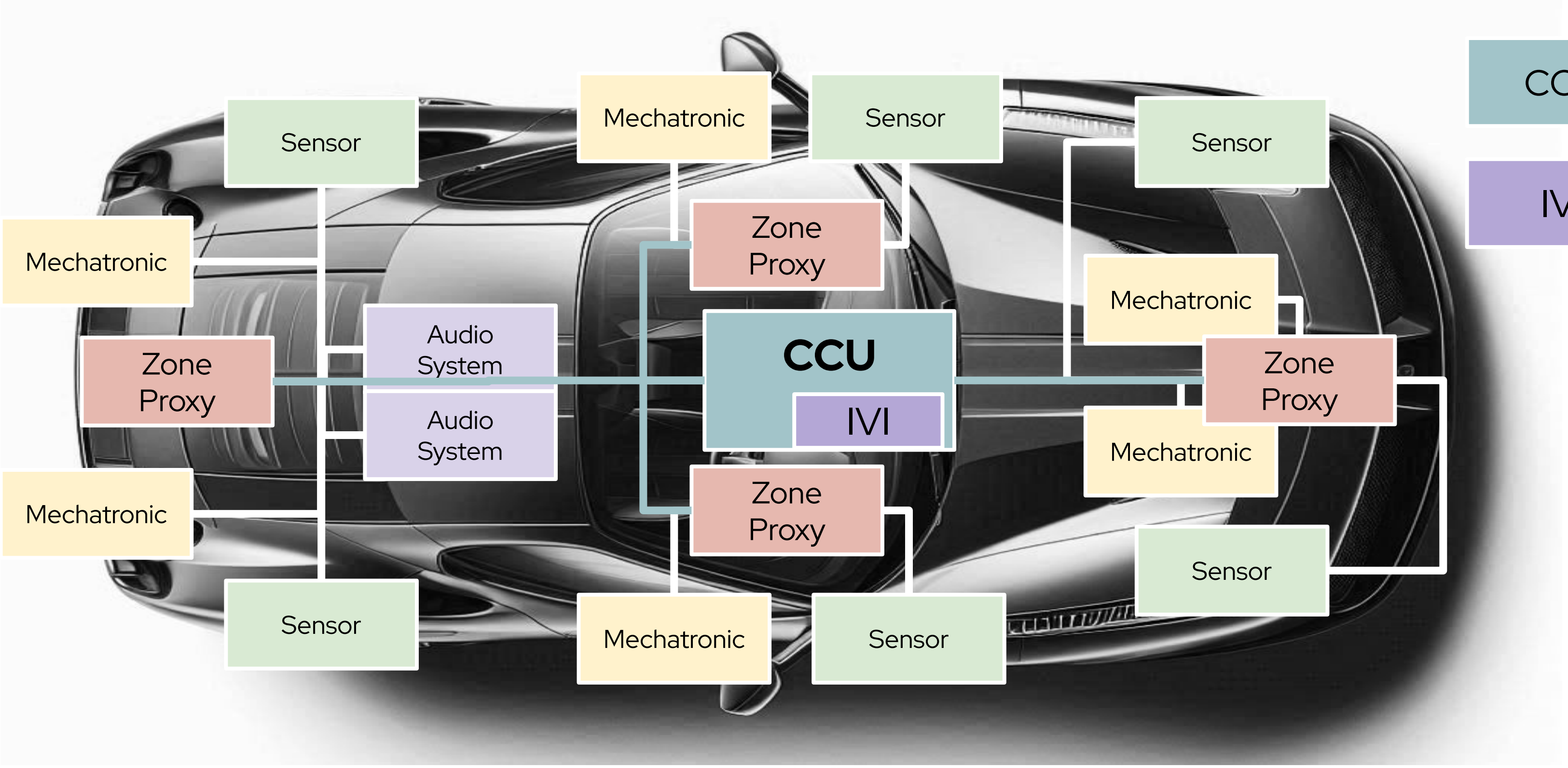
Grouped systems, updatable ECUs



ECU	Electronic Control Unit
IVI	In-Vehicle Infotainment

CCU-based design

A Central Unit To Rule Them All



CCU	Central Control Unit
IVI	In-Vehicle Infotainment

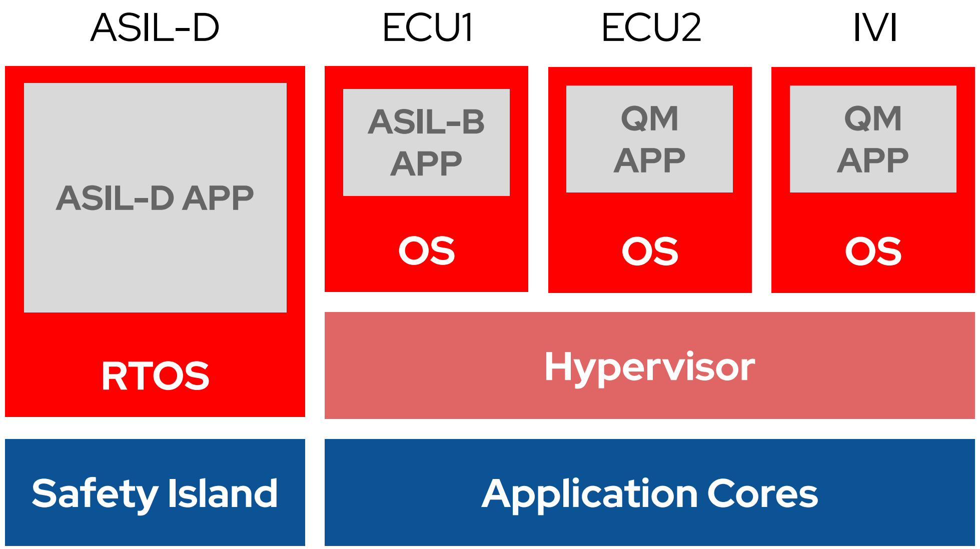
A little bit about Functional Safety

Just some terminology

- FuSa is an industry standard (ISO26262) that provides a systematic approach to analyze risk in electrical and electronic (this includes software components) components in a car.
  - Also provides a common language for integrating components from different companies.
- Factors in various aspects (harm to individuals, time sensitiveness…) to classify components.
  - ASIL-A (lowest criticality) to ASIL-D (highest criticality).
  - QM (Quality Managed), no requirements apply.

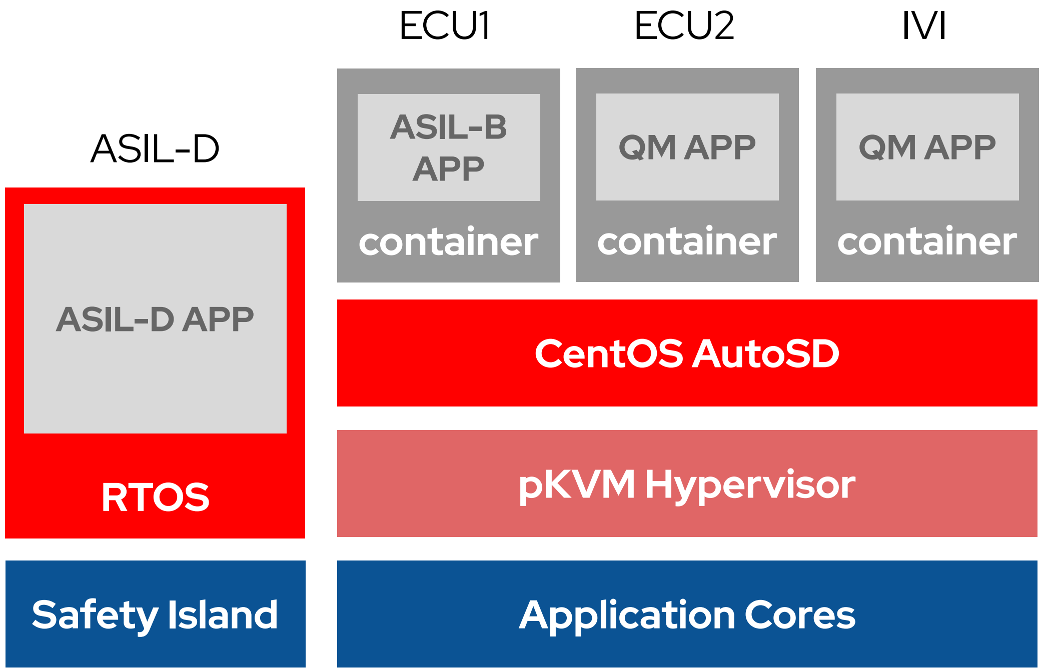
Conventional approach to Virtualization

Consolidating ECUs and IVI into the CCU



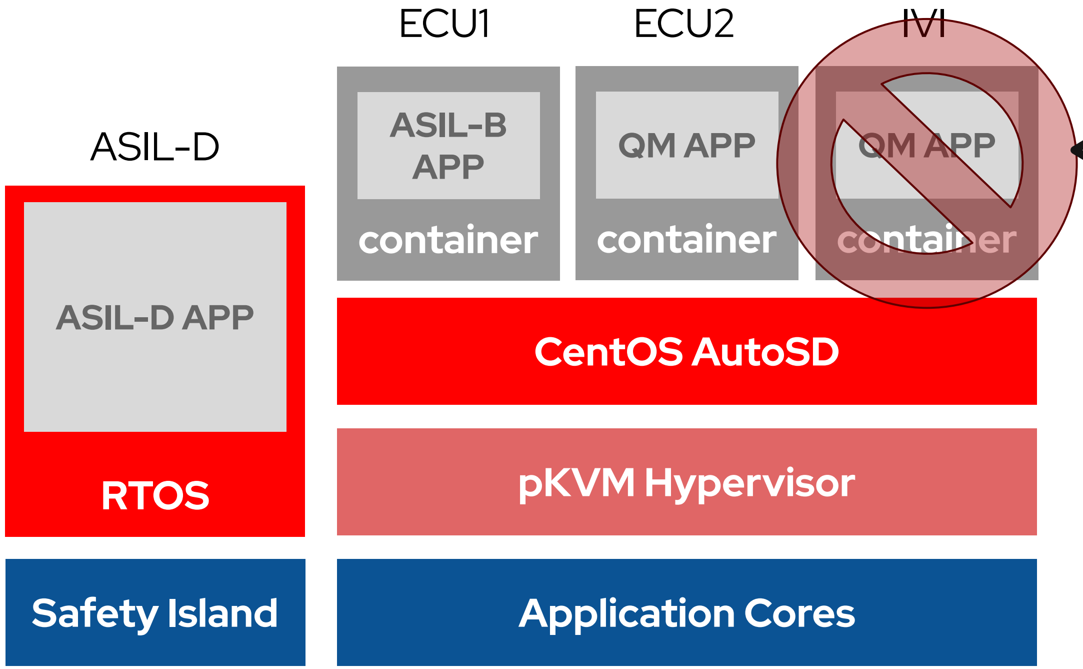
CentOS AutoSD approach to Virtualization

Containers instead of VMs



CentOS AutoSD approach to Virtualization

It’s never that easy…



Popular IVIs such as Android Automotive OS (AAOS) can’t be easily containerized

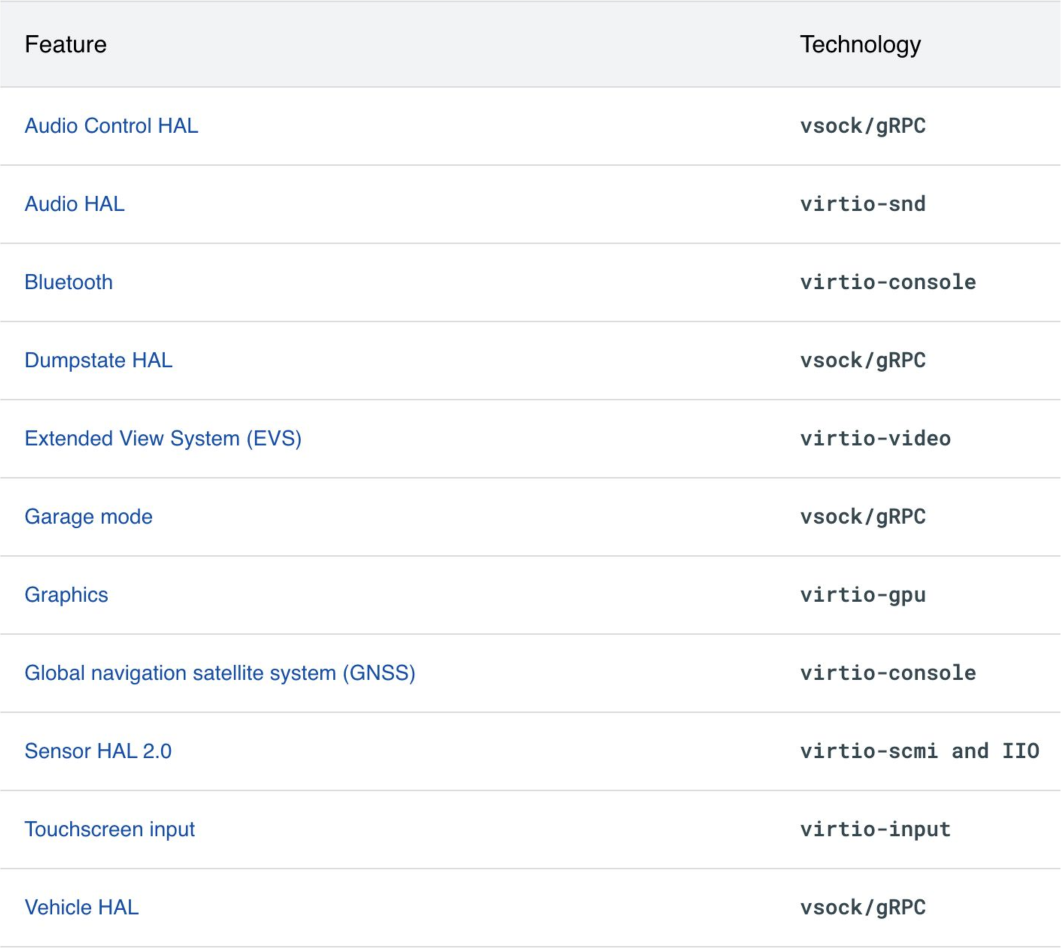
About Android Automotive OS

Android flavored IVI

- An AOSP (Android Open Source Project) variant targeted towards automotive.
- Applications are developed targeting a distraction-prevention oriented framework that's aware of the car state.
- Reference implementation targets a virtual platform (originally trout, now cuttlefish-auto)

Android Automotive OS Requirements

A nice set of virtual devices



Feature	Technology
Audio Control HAL	`vsock/gRPC`
Audio HAL	`virtio-snd`
Bluetooth	`virtio-console`
Dumpstate HAL	`vsock/gRPC`
Extended View System (EVS)	`virtio-video`
Garage mode	`vsock/gRPC`
Graphics	`virtio-gpu`
Global navigation satellite system (GNSS)	`virtio-console`
Sensor HAL 2.0	`virtio-scmi and IIO`
Touchscreen input	`virtio-input`
Vehicle HAL	`vsock/gRPC`

Getting QEMU ready for AAOS (I)

Extending virtual device support

- Multi-touch support for virtio-input
- New vhost-user devices:
  - vhost-user-scmi
  - vhost-user-sound
  - vhost-user-media: specs and Rust implementation in progress
  - vhost-user-gpu: reimplemented in Rust with gfxstream support

Getting QEMU ready for AAOS (II)

Consuming AAOS disk images

- cvd2img
  - AAOS build system produces phone-like disk images.
  - Transforms AAOS disk images into raw disk images, including the embedded device properties.
- start-avm
  - AAOS includes *launch_cvd*, which starts both the VMM and a number of requires services, but doesn’t fit our use case.
  - Starts the required services and QEMU with the blessed command line, in a container-friendly way.

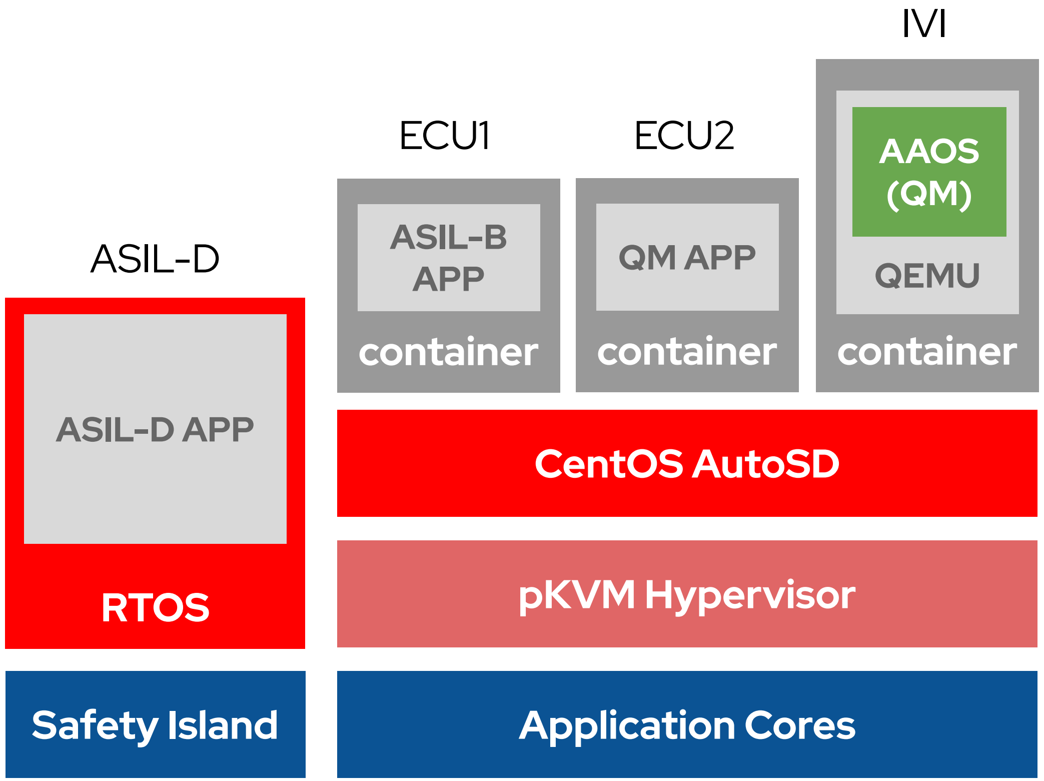
Getting QEMU ready for AAOS (III)

Putting everything into a container

- Container entrypoint is **start-avm**
  - Needs access to */dev/dri* (for GPU acceleration) and to a secondary *dbus* server.
- Display and input is managed by an external program in a different container which communicates with QEMU via *dbus*.
- Disk images may be bundled or provided as volumes (-*v* option in podman), depending on the update mechanism for AAOS.
  - Raw storage is also possible.

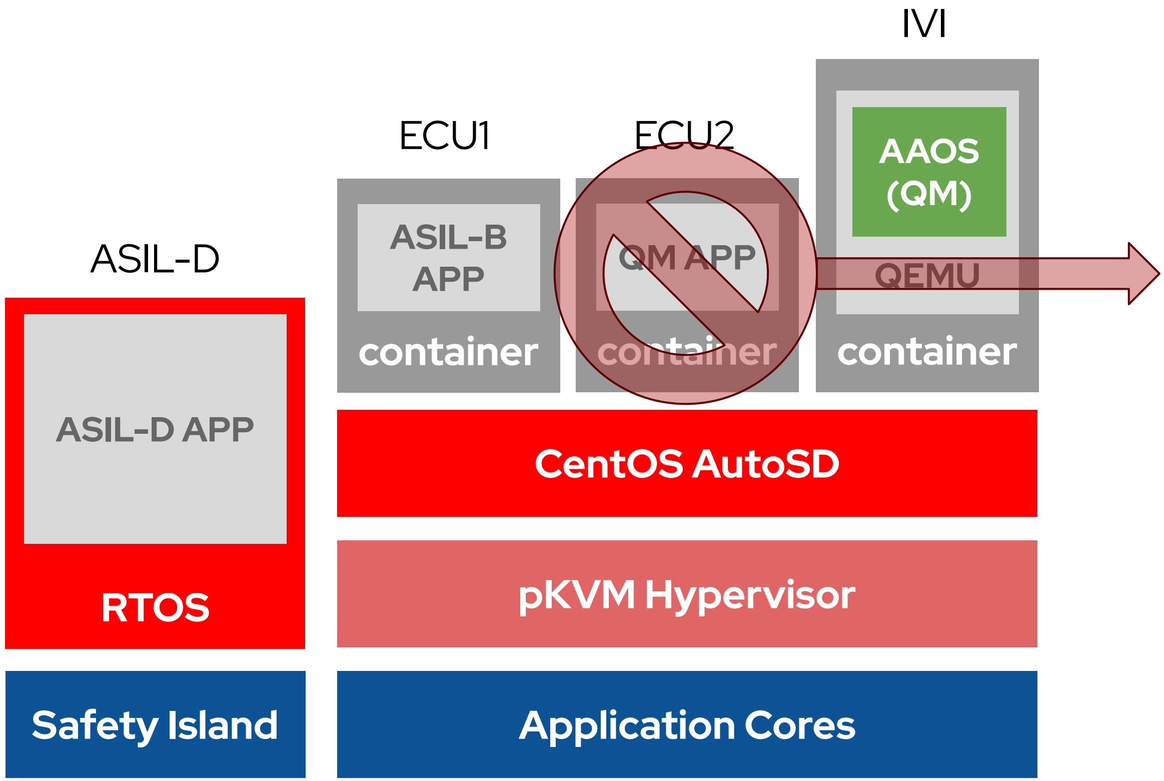
CentOS AutoSD approach to Virtualization with AAOS

It works now!



CentOS AutoSD approach to Virtualization with AAOS

Well, almost...



Certain markets may require the use of Virtualization for some components

Enabling Virtualization for regular QM apps (I)

Requirements

- Preserve the container development and deployment flow as much as possible.
  - Everything should be done through **podman** and be compatible with **quadlets**
- Disk images aren’t an option.
- Startup time is **extremely** important.

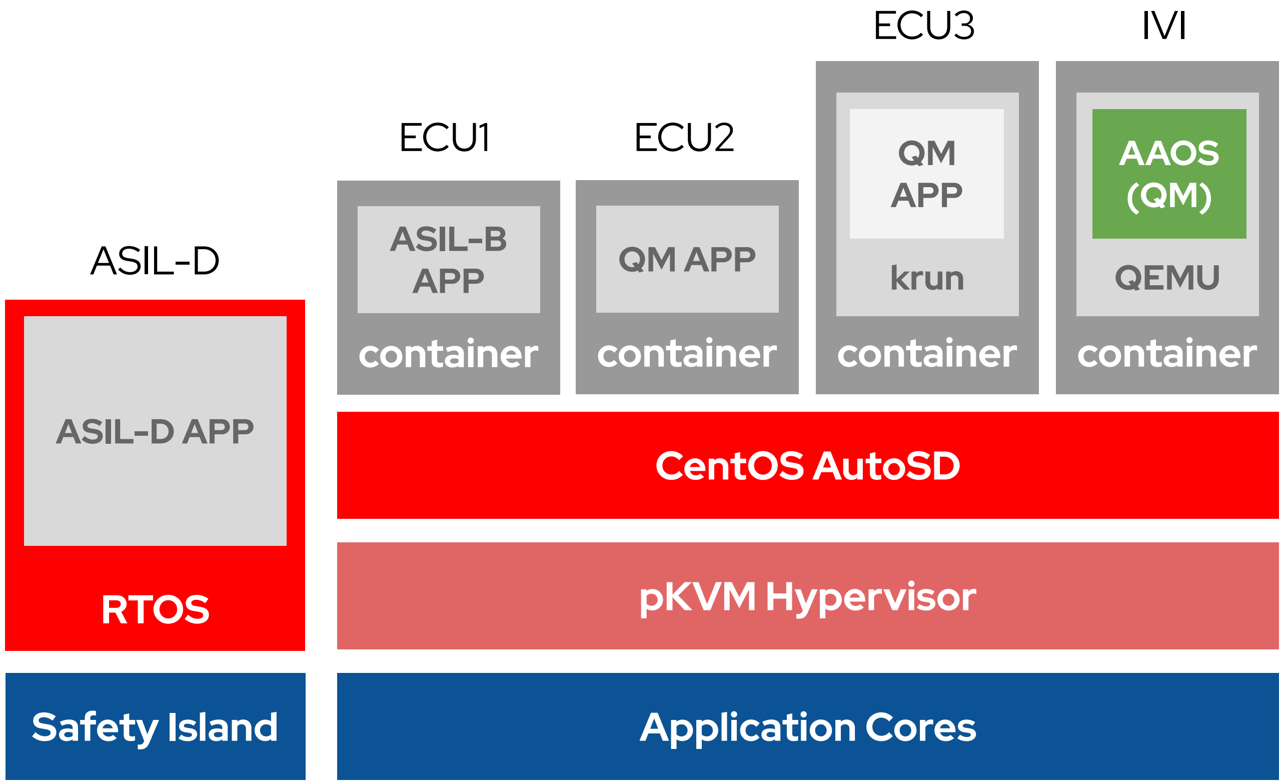
libkrun in CentOS AutoSD

Enabling podman to use Virtualization

- Written in Rust and provided as a dynamic library.
  - Integrated in podman through **crun**.
- In-process virtio-fs support allows avoiding disk images.
- Reduced size and complexity makes it future-proof towards possible FuSa requirements.
- Focus on reduced boot time.

Complete CentOS AutoSD approach to Virtualization

Now we’re really there



Thank you

Red Hat is the world's leading provider of enterprise open source software solutions. Award-winning support, training, and consulting services make Red Hat a trusted adviser to the Fortune 500.

in linkedin.com/company/red-hat

youtube.com/user/RedHatVideos

facebook.com/redhatinc

twitter.com/RedHat