Blueprint Submission for Time-critical Edge Compute

Intel Corporation, Inc.

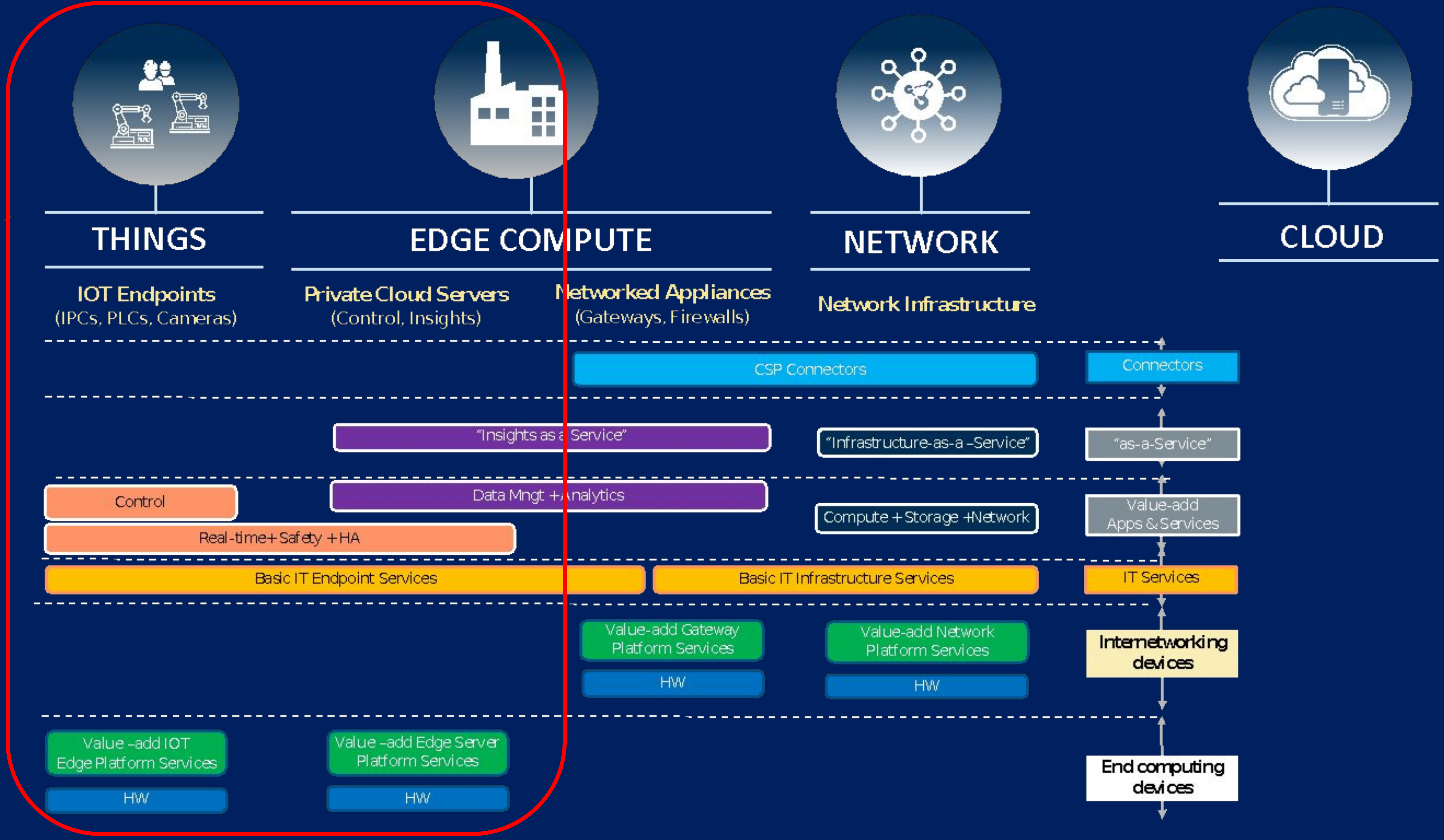
Time-critical Edge Compute Blueprint: Use Cases

- Use cases in Manufacturing, Smart Buildings, general IIOT
  - Virtualized PLC
  - Computer vision inference
  - Machine, sensor data inference
  - Process or discrete manufacturing closed loop control
  - Ethernet TSN
- Functional Safety capable use cases
  - Discrete manufacturing soft PLC
- Onramp for 5G-URLLC

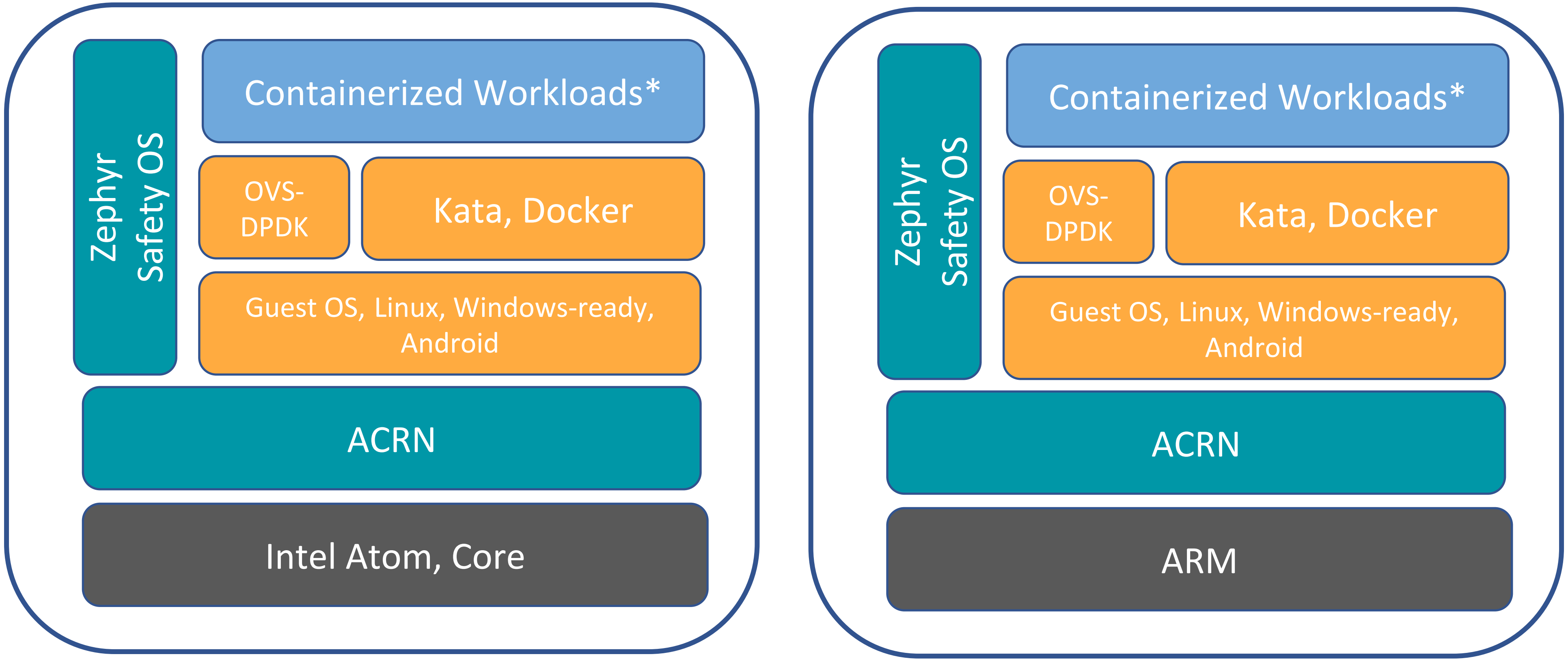
Time-critical Edge Compute Blueprint: Hardware and Partners

- Low power, ruggedized hardware
  - Dell 3000, 5000 IPC
  - Huawei XXX industrial gateway
- Potential to attract new members to Akraino project
  - Industrial ODM’s e.g. Advantech, Adlink
  - Industrial OEM’s/ISV’s e.g. TTTech, Nebbiolo, IOTech
  - Industrial end-users e.g. ExxonMobil

Time-critical Edge Compute Blueprint: Deployment Focus



Time-critical Edge Compute Blueprint: Deployment Scenarios

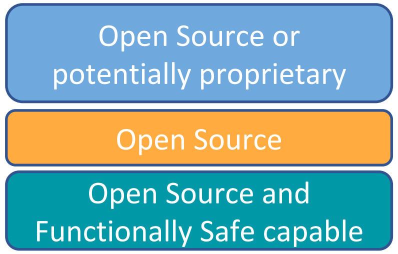


Virtualized, Functionally Safe workloads in addition to others

Easily Extensible and Expandable, by just adding more systems

Evaluating Airship for ZTP and deployment

See next page for some sample targets



Containerized edge workloads

- Containerized workloads orchestrated via Kubernetes tuned for lightweight, time-critical embedded deployments
- Sample workloads include
  - Tensorflow via Kubeflow
  - OpenVINO for Video and Inference
  - Closed loop control (e.g. IEC 61131)
  - EdgeX Foundry
  - Building controller

Demo

This stack is largely functional today.

Work ahead is in hardware software validation and validation of the workloads described.

Demo Link: https://youtu.be/1qkRJIulUSY

Backup: Deep dives for underlying technology

- Zephyr OS
- ACRN Hypervisor
- Kata Containers
- Celadon - A fully Open Source Android Stack
- OVS-DPDK

Zephyr™

A scalable real-time operating system (RTOS) supporting multiple hardware architectures, optimized for resource constrained devices, and built with security in mind.

https://www.zephyrproject.org/

Overview – A Fully Featured Open Source RTOS (since 2016)

Safety

- Thread Isolation
- Stack Protection (HW/SW)
- Quality Managed (QM)
- Build time configuration
- No dynamic memory allocation
- FuSA (2019)

Security

- User-space support
- Crypto Support
- Software Updates

Configurable & Modular

- Zephyr Kernel can be configured to run in as little as 8k RAM
- Enables application code to scale
- Configurable and Modular

Cross Platform

- Support for multiple architectures
- Native Port
- Developed on Linux, Windows and MacOS

Open Source

- Licensed under Apache II License
- Managed by the Linux Foundation*
- Transparent development
- Fork it on Github!

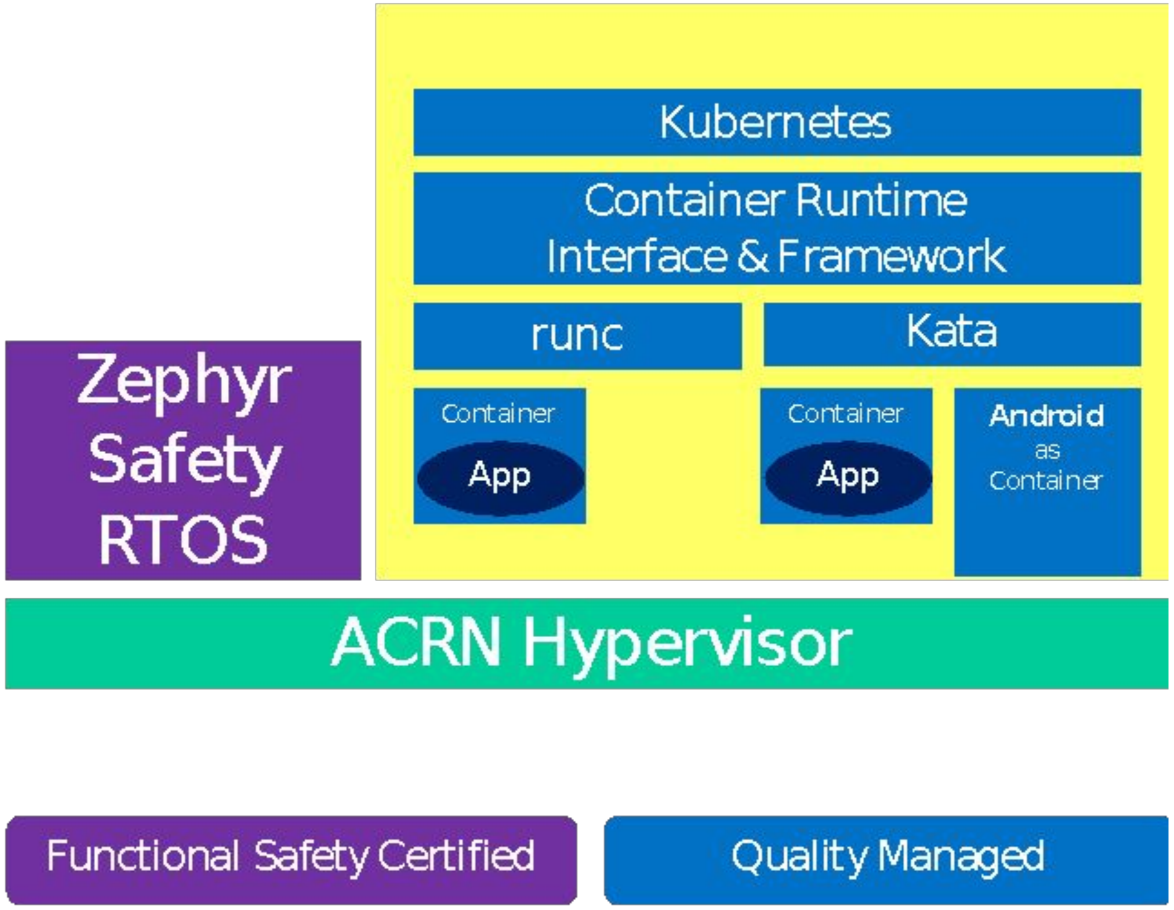
Connected

- Full Bluetooth 5.0 Support
- Bluetooth Controller
- BLE Mesh
- Thread Support
- Full featured native networking stack
- DFU (IP+BLE)

Zephyr™ is not an ingredient, Zephyr™ provides a complete solution.

Zephyr Enabled as a Safety Critical OS

- Runs on a custom hypervisor that is safety critical capable
- Security updates with the latest fixes
- Similar to Cloud Software Defined Infrastructure (SDI)



- Zephyr = FuSa (2019)
- Linux = Quality Managed
- Android = non-FuSa

Zephyr™ OS Direction

Safety & Security

- Functional Safety (FuSa) core OS certification: secure & harden kernel (IEC61508 SIL3).
- Development model & process with safety and security in mind.
- Trusted Execution Environments.

E2E Platform

- Bootloader.
- Device firmware updates.
- Cloud connectivity.
- Development tools.

Expanded Use Cases

- Industrial, safety, and security features.
- Deep embedded usages (BLE, 802.15.4 (zigbee), BT Mesh.
- Advanced configurations and use cases: Multicore, SMP, AMP.

Ecosystem & Portability

- Improve support on Mac and Windows.
- IDE integration.
- 3rd party tools: tracing, profiling, debugging.
- LLVM, commercial compilers.
- Standard APIs and portability: POSIX layer (PSE54), BSD socket, and CMSIS RTOS.

ACRN

A Big Little Hypervisor for IoT Development

What is ACRN™?

ACRN is a flexible, lightweight **reference hypervisor**, built with real-time and safety-criticality in mind, **optimized** to streamline **embedded development** through an open source platform.

A Big Little Hypervisor for IoT Development

ACRN is a registered trademark of the Linux Foundation

ACRN™ Features

- Small Footprint
- Built for IoT
- Adaptability
- Built for Real-Time
- Safety Criticality
- Truly Open Source

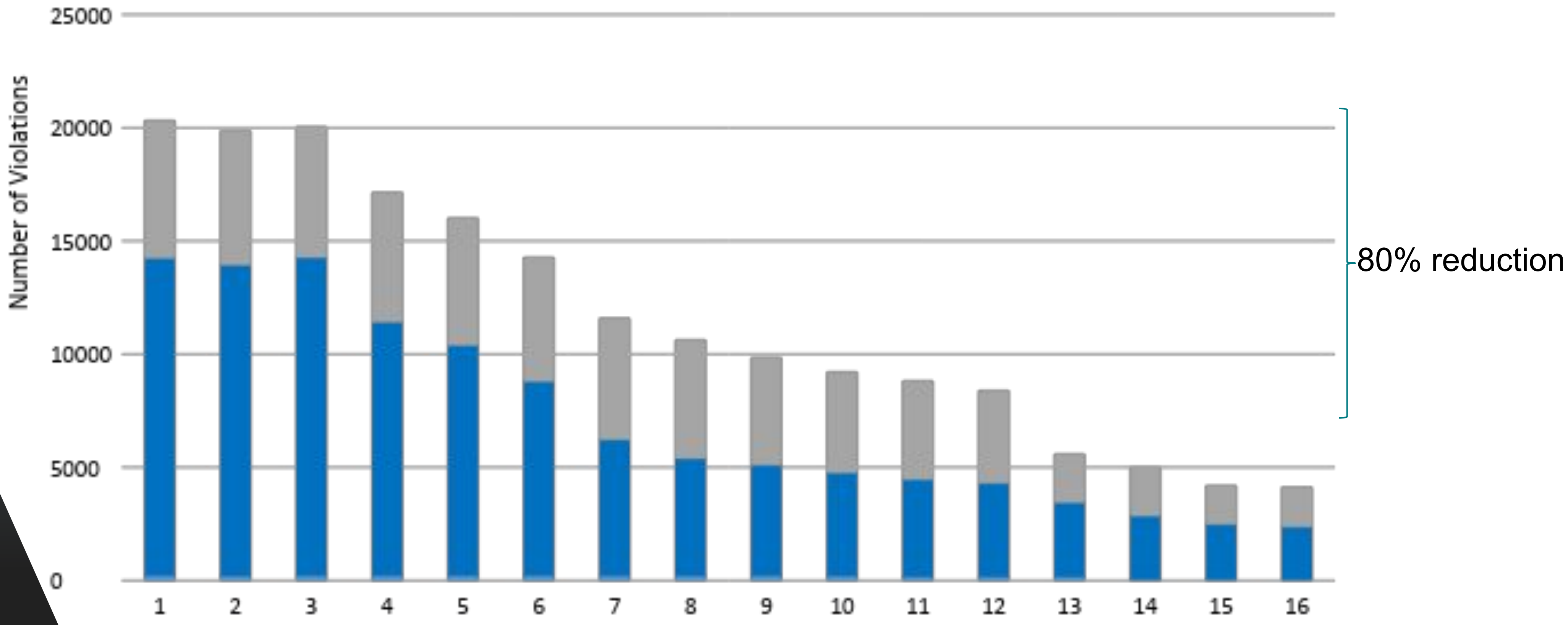
ACRN is a registered trademark of the Linux Foundation

Features Roadmap - Proposal

*	Limited to specific HW
PT	Pass through

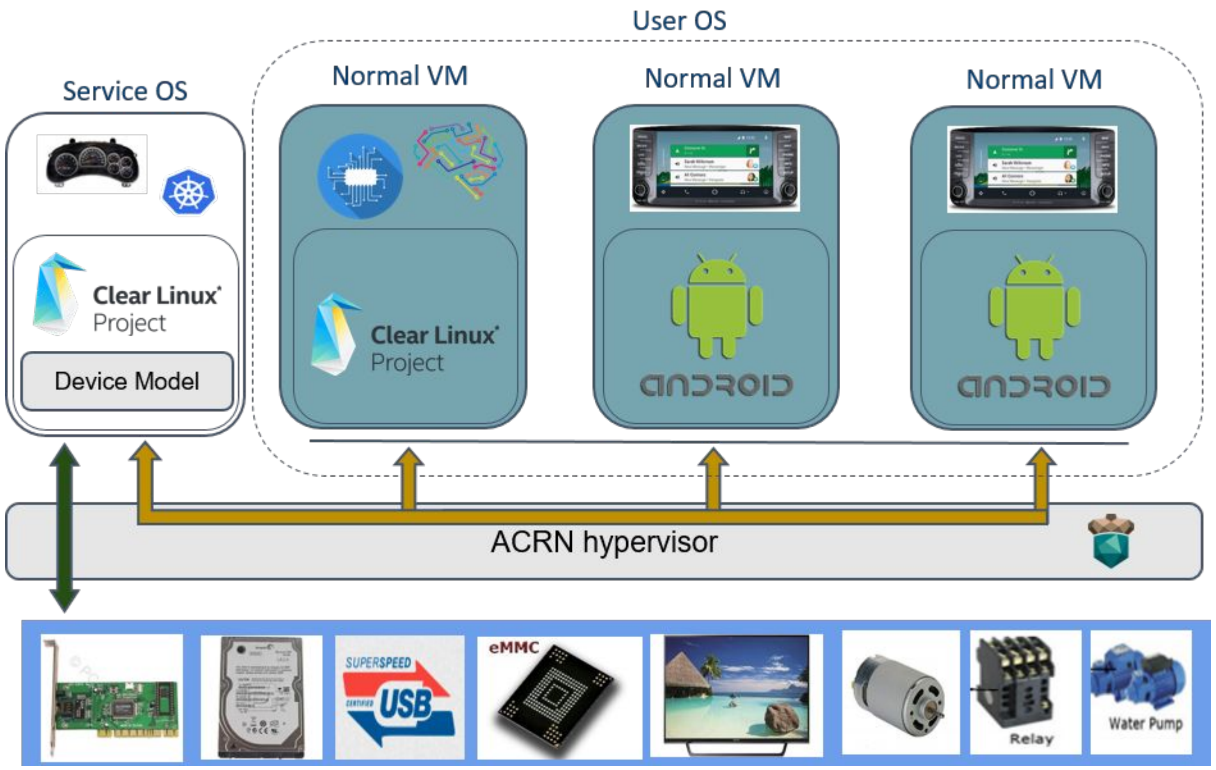
	Dates below are for reference only and subject to change				
Area	**v0.1@Q2'18**	**v0.2@Q3'18**	**V0.5@Q4'18**	**V1.0@Q1'19**	**V1.x@2019**
HW	• APL NUC (UEFI) • APL UP2 (UEFI)	• APL NUC (UEFI) • APL UP2 (UEFI)	• APL NUC (UEFI) • KBL NUC (UEFI) • APL UP2 (UEFI)	• APL NUC (UEFI) • KBL NUC (UEFI) • APL UP2 (UEFI)	• APL NUC (UEFI) • KBL NUC (UEFI) • APL UP2 (UEFI) • ARM
Hypervisor	• VT-x • VT-d • CPU static-partitioning • memory partitioning • Virtio (v0.95) • VHM • EFI boot • Clear Linux as guest	• Virtio (v1.0) • Power Management (Px/Cx) • VM management • ACRN debugging tool • vSBL	• Android as guest • AliOS as guest • Zephyr as guest • MISRA C compliance • Logical partitioning without SOS • Trusty (Security) • SBL boot	• vHost • Power Management (S3/S5) • Hybrid Mode (Privilege VM loaded by SOS) • Real Time phase I	• Real Time phase II • Hybrid Mode (Privilege VM loaded by hypervisor) • Windows as guest • VxWorks as guest • Functional Safety capable • CPU sharing • OVMF • ARM
I/O virtualization	• Storage • Ethernet • USB host controller (PT) • USB device controller (PT) • Audio (PT) • WiFi (PT)* • Touch (PT)	• GPU Sharing: • GPU Surface Sharing • IPU Sharing*	• GPU Prioritized Rendering • Touch sharing • IOC sharing* • Audio sharing • USB host controller Sharing • USB DRD virtualization	• GPIO virtualization	• HECI sharing (Security) • CSME/DAL sharing (Security) • TPM Sharing (Security) • eAVB/TSN Sharing • SR-IOV*

Towards MISRA-C Compliance

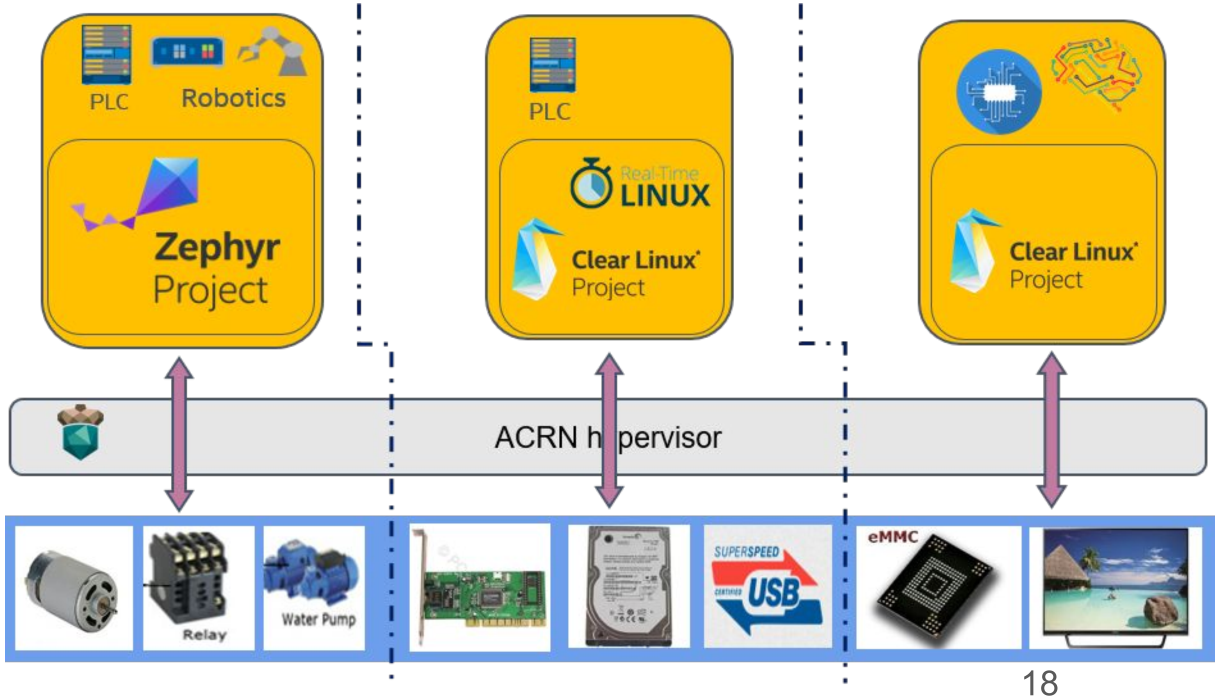


- Statistics from commercial safety-qualified checker.
- False positives and intended deviations tracked in weekly-updated sheets.
- Pull requests are scanned hunting for new violations.

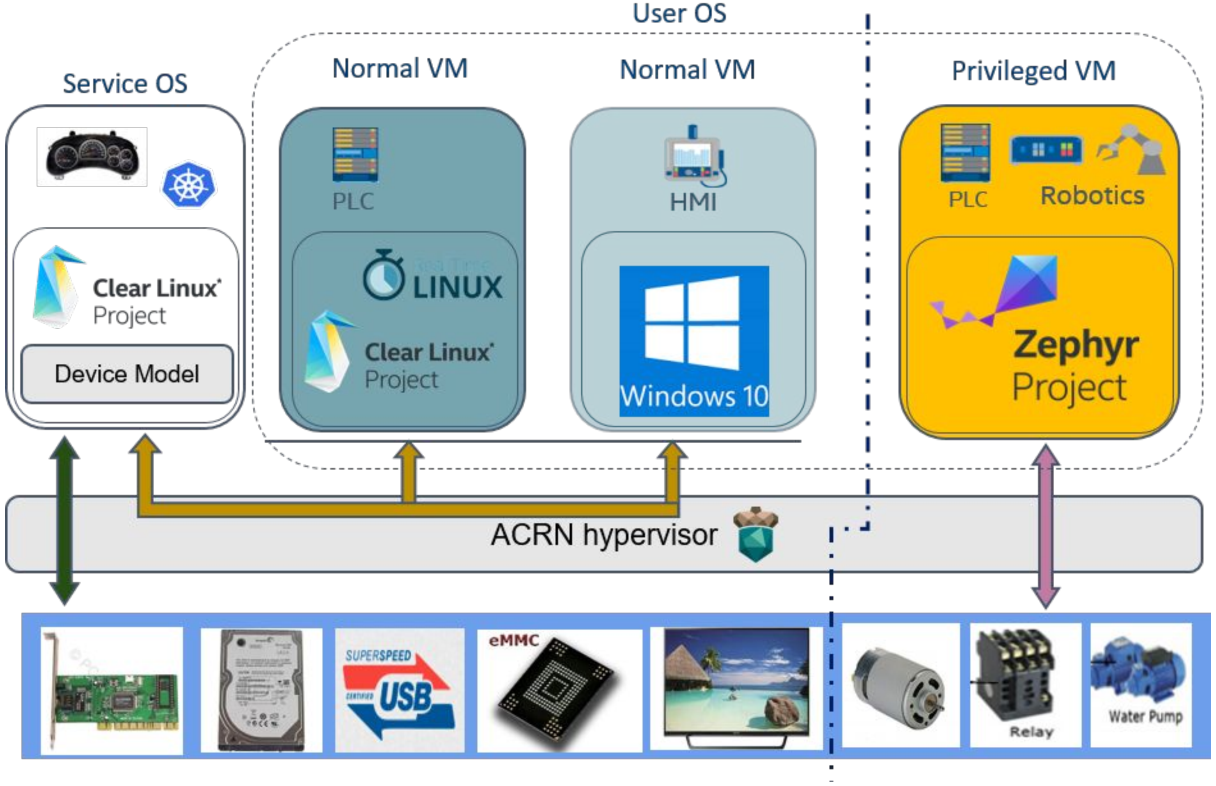
Sharing Mode

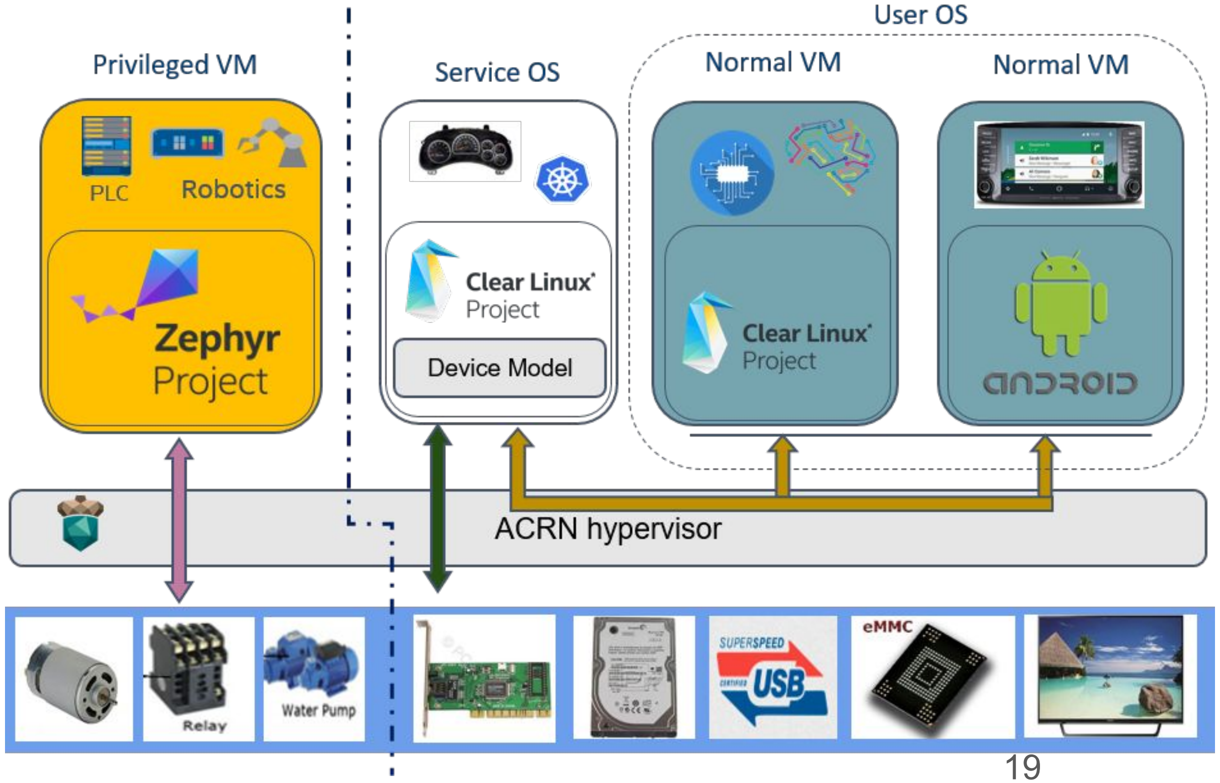


Partition Mode



Hybrid Mode


User OS
Service OS
Normal VM
Normal VM
Privileged VM
PLC
HMI
PLC
Robotics
Clear Linux Project
LINUX
Clear Linux Project
Windows 10
Zephyr Project
Device Model
ACRN hypervisor
SUPERSPEED USB
eMMC
Relay
Water Pump


User OS
Privileged VM
Service OS
Normal VM
Normal VM
PLC
Robotics
Zephyr Project
Clear Linux Project
Device Model
Clear Linux Project
ANDROID
ACRN hypervisor
Relay
Water Pump
SUPERSPEED USB
eMMC

Kata Container Project

https://katacontainers.io/

Project Overview, Status

What is Kata?

- kata-runtime, an OCI (Open Containers Initiative) compliant runtime
    - *Seamless* integration into cloud native ecosystem
- “Providing the speed of containers with the security of virtual machines”
    - Light-weight enough to be used with micro-services design patterns
    - More than just security of virtual machines, it is an additional layer on top of existing container security primitives.
    - Each container/pod is created within its on virtual machine

Who is Kata?

- Open source, open governance project with original contributions from Intel's Clear Containers and Hyper.sh's runV
- Under the Openstack Foundation Umbrella (not managed by openstack)
- Architecture Committee: Google, Huawei, Hyper.sh, Intel
- Contributors include: AMD, ARM, Branch, IBM, Intel, Google, Huawei, Hyper.sh, Microsoft, Nvidia, Openstack Fountain, Redhat, Suse, ZTE, 99Cloud …

Where does Kata make sense?

- Regulated and sensitive production environments
- Too many capabilities required which increase attack surface
- Desire to easily run on multiple or custom kernel versions
  - Legacy applications on older kernels in containerized environment
  - Custom kernel features required
  - Testing on cutting edge kernels

Where else does Kata make sense?

- Bare-metal infrastructure
- Mixed levels of trust
  - Multiple tenants
  - Untrusted workloads

Kata Updates since release

V1.0 (May 2018)	V1.2 (August 2018)
• Seamless integration with Kubernetes (CRI), Docker • Hardware isolation using KVM/QEMU • Optimizations for minimal footprint and boot-time • Seamless integration with major networking plugins ◦ Advanced networking available through DPDK (VPP/OVS and SR-IOV) - High bandwidth, low latency networking ▪ Ability to run custom kernels at the container or pod level • Direct device assignment (GPU, RDMA, QAT, etc.)	• Support multiple architectures • VM-Factory support [1] • Vsock support [2] • K8S deployment through container based daemonset [3] • Bug fixes, enhancements [1] - https://github.com/kata-containers/runtime/pull/303 [2] - https://github.com/kata-containers/runtime/issues/383 [3] - https://github.com/kata-containers/packaging/pull/65

Kata Roadmap

V1.3 (September 2018)	Looking forward
• Full network hotplug • Full storage hotplug • Open-tracing support (Jaeger) • CNI-Macvlan support • Containerd v2 shim	• Runtimeclass • More native integration with CRI (containerdv2 for CRIO) • Security Enhancements • Live upgrade • Performance optimizations See https://github.com/orgs/kata-containers/projects/12

Project Celadon

https://01.org/projectceladon/

Project Celadon: Elements & Benefits

Code transparency

open source code **provides freedom and flexibility to customize and accelerate development**

Turnkey system

supports a wide range of hardware components optimized for Intel architecture making it **easy for rapid prototyping and building new applications**

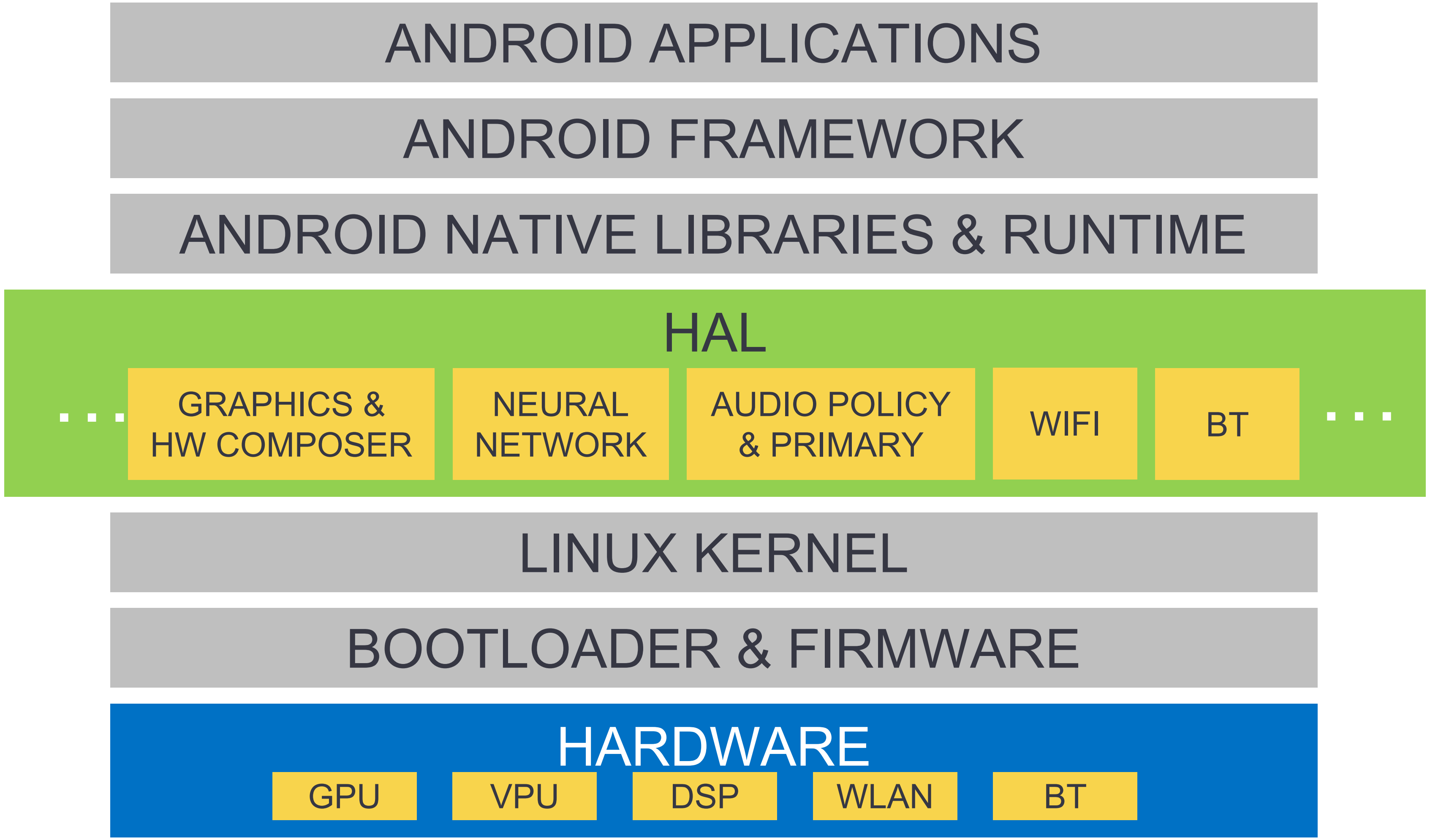
Regularly updated

opportunity to realize new features and improvements by developing on the latest hardware implementations and Android software updates

Verified compatibility

basic Android compatibility **ensures consistent application and hardware environment and experience**

Architecture



Built on standard and familiar android stack architecture

DPDK

DATA PLANE DEVELOPMENT KIT

https://www.dpdk.org/