Applying Linux to the Civil Infrastructure

Yoshitake Kobayashi[1)], Toshiba
Urs Gleim, Siemens AG

LinuxCon Japan 2015
3-5 June 2015

1) CE Workgroup

Scope of this presentation

- **Create a place for collaboration**
  - Share opinions with audience about the “future” of civil infrastructure systems
  - Collect “requirements” for civil infrastructure systems.
  - Recruit companies/developers to work with us in this area.

Outline

- **Definition of Civil infrastructure**
- **Motivation and goal**
- **Target Platform Building blocks and Technical requirements**
- **Current status**

Definition

Civil Infrastructure Systems

are technical systems responsible for supervision, control, and management of infrastructure supporting human activities, including, for example, electric power generation and energy distribution, oil and gas, water and wastewater, healthcare, communications, transportation, and the collections of buildings that make up urban and rural communities. These networks deliver essential services, provide shelter, and support social interactions and economic development. They are society's lifelines.[1)]

Note: Most of Japanese company use “Social infrastructure” instead of “Civil infrastructure”. However, Civil Infrastructure is more suitable term in other countries.

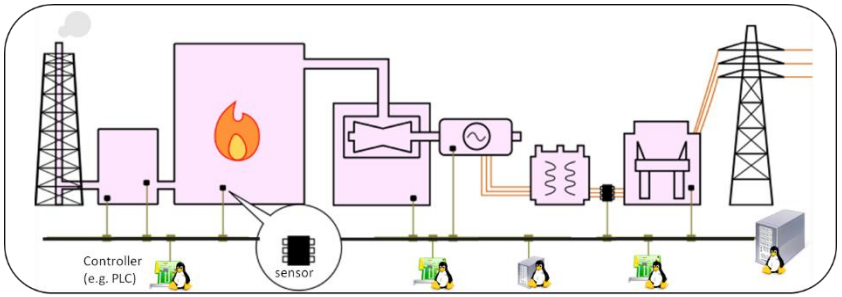
1) adapted from https://www.ce.udel.edu/current/graduate_program/civil.html

Motivation

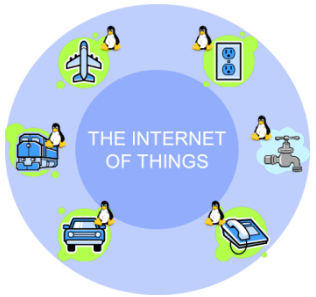
1. Civil infrastructure systems are currently built from the ground up for each product, with **little re-use of existing software building blocks**, for example:
   - **Operating systems**
   - **Virtualization technologies**
   - **Middleware**
   - **Mechanisms for software/firmware updates**

2. Functionality required for industrial-grade applications is in many aspects converging to that offered by IT driven solutions[1)]. However, by today's software platforms **many non-functional requirements are not addressed** sufficiently:
   - **Functional Safety**
   - **Reliability**
   - **Maintainability, long term support**
   - **Security**
   - **Real-time support**

3. The **Internet-of-Things** connects previously stand-alone systems with open protocols to create **systems of systems**. This trend will substantially influence industrial system architectures.



stand alone



connected

1) Open Source Software / Linux

Vision

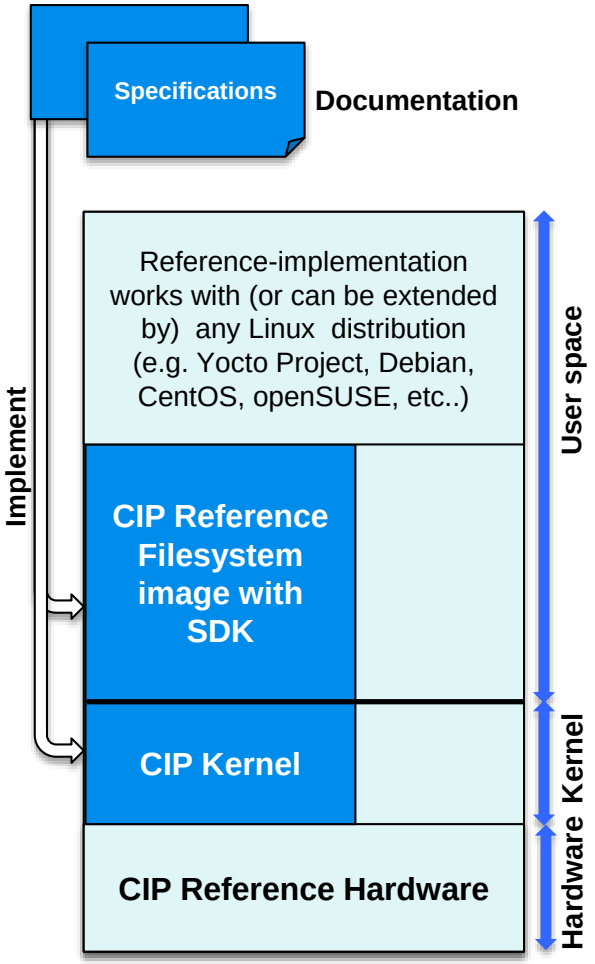
Jointly establish a scalable Open Source “base layer” of industrial grade software:

- Speed up implementation of civil infrastructure systems.
- Build upon existing open source foundations and expertise without reinventing non-domain specific technology.
- Establish (de facto) standards by providing a base layer reference implementation.
- Contribute & influence upstream projects regarding industrial needs.
- Motivate suppliers to actively support these platform / provide an implementation (e.g. silicon vendors).
- Ensure long term stability and maintainability.

A “base layer” – like the plain operating system – does not contribute to competitive innovation, but needs to be provided by every single vendor.

Goals

- Sharing development effort for development of industrial grade base systems.
- Fill the gap between capabilities of the existing Open Source Software and industrial requirements.
- Reference-implementation consisting of
  - Specification of on-device software stack and tools infrastructure
  - Linux kernel, file system, etc. selected reference hardware
  - Build environment and tools for companies to build their own distribution.
  - Test framework and test cases
  - SDK (e.g., poky based) and APIs (based on POSIX; compatibility layers for legacy APIs)
- Wide usage and acceptance in industry.
- Trigger development of an emerging ecosystem including tools and domain specific extensions.



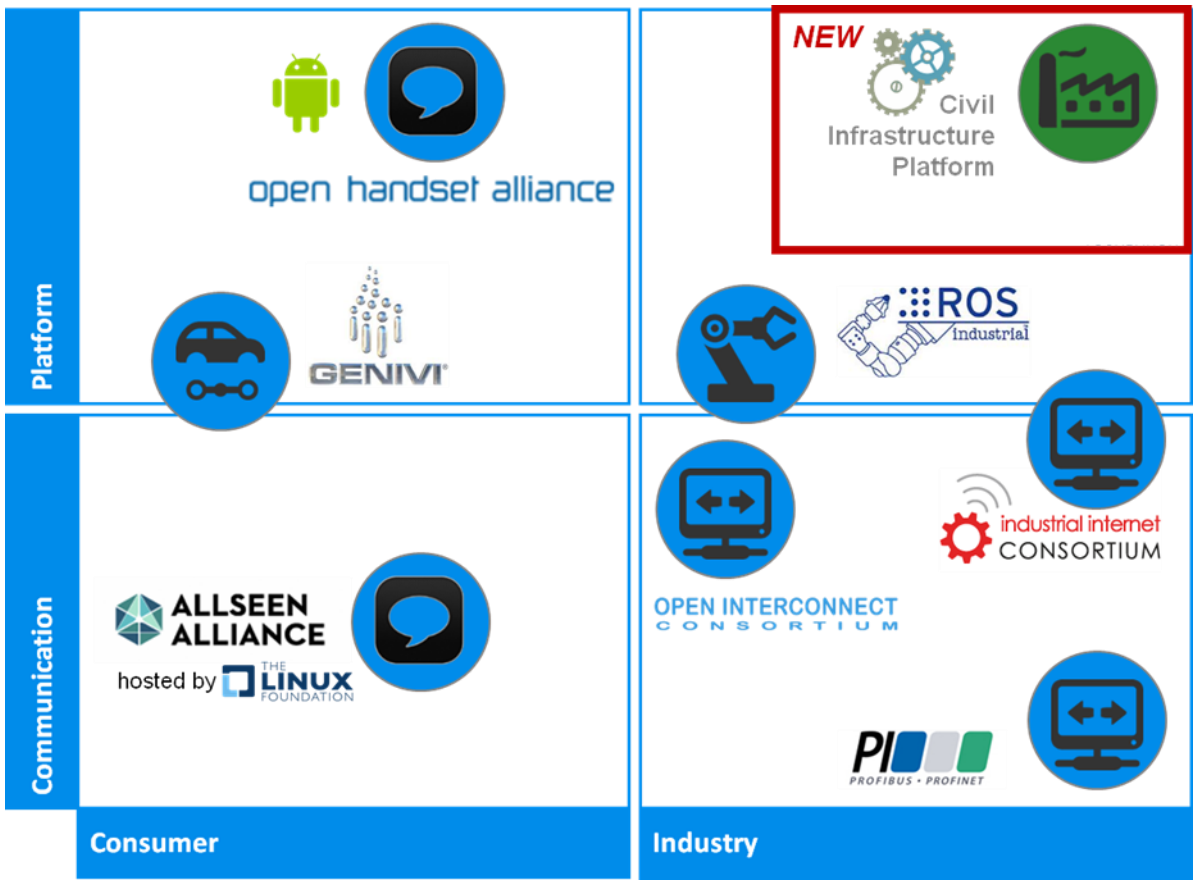
Outcome

CIP realizes an industrial grade, sustainable, standard software stack.

Reference implementation	Life-cycle management	Setting standards
Integrated reference platform implementation and build environment: ▪ Reference architecture of base platform and CIP specific extensions ▪ Selection/support of applicable upstream projects ▪ Tool chain set-up, platform implementation, integration ▪ Platform implementation for selected device classes and use cases	Processes for industrial use and sustainable long-term support: ▪ Test and validation: frameworks for first release and updates ▪ Maintenance strategy and long term support (LTS, LTSI) ▪ License clearing of used open source components ▪ Export control classification (ECC) ▪ License barrier architecture guidance	Harmonize base platform and fulfill certification standards: ▪ Standardize base platform components (select exis-ting standards and fill gaps with de-facto standards) ▪ Foster OSS acceptance for safety/security critical projects ▪ Provision of artefacts needed for certification (e.g. test reports) ▪ Development process assessment of relevant upstream projects

Comparison with existing Alliances

Other domains already benefit from collaborative development.



- **Development speed, shorter product cycles**
- **Software quality**
- **Establish a standard platform and enable ecosystems (e.g. for development tools, system extensions)**

Even competing companies as car manufacturers work in alliances already.

(Genivi, for example)

Target Systems

	Networked Node [1]	Target systems: Control Unit [2]	Embedded Computer [3]	Embedded Server [4]
Processor (example)	ARM M0/M0+/M3/M4	ARM M4/7,A9,R4/5/7,Atom	ARM A9/A15,R7,Core,PPC	ARM A53/A57,Xeon
Architecture, clock	8/16/32-bit,< 100 MHz	32-bit, <1 GHz	32/64-bit, <2 GHz	64-bit, >2 GHz
non-volatile storage	n MiB flash	n GiB flash	n GiB flash	n TiB flash/HDD
RAM	< 1 **M**iB	**1 M**iB - 1 GiB	256 MiB - 2 GiB	2 GiB - 768 GiB
HW ref. platform	Arduino class board	Raspberry Pi class board	SoC-FPGA, e.g.Zync	**i**ndustrial PC
application examples	**Sensor, field device**	**control systems**	**special purpose server based controllers**	
	PLC	**gateways**	**multi-purpose controllers**	

Excluded:

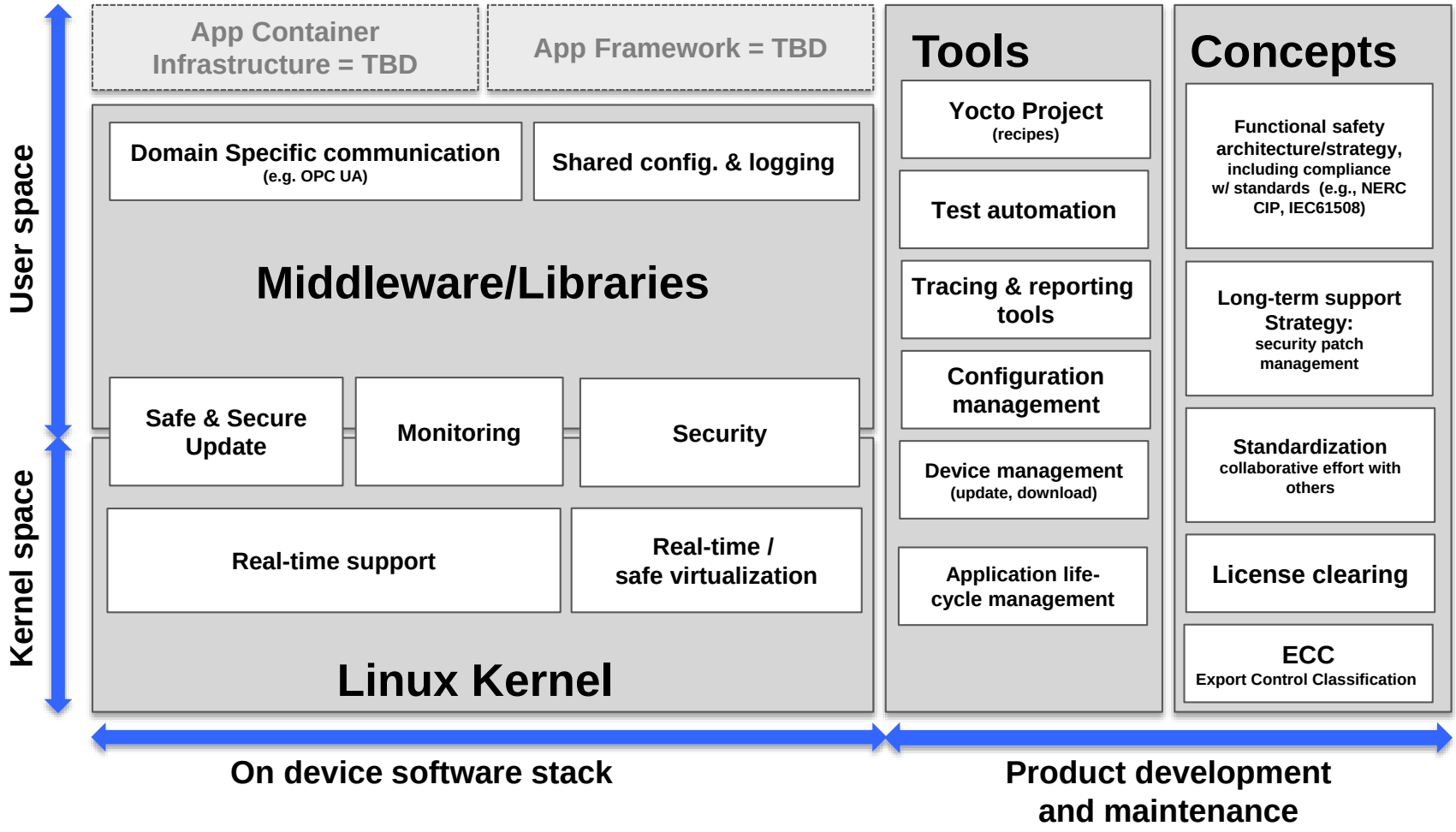
- Enterprise IT and cloud system platforms.

Proposed reference hardware for common software platform:

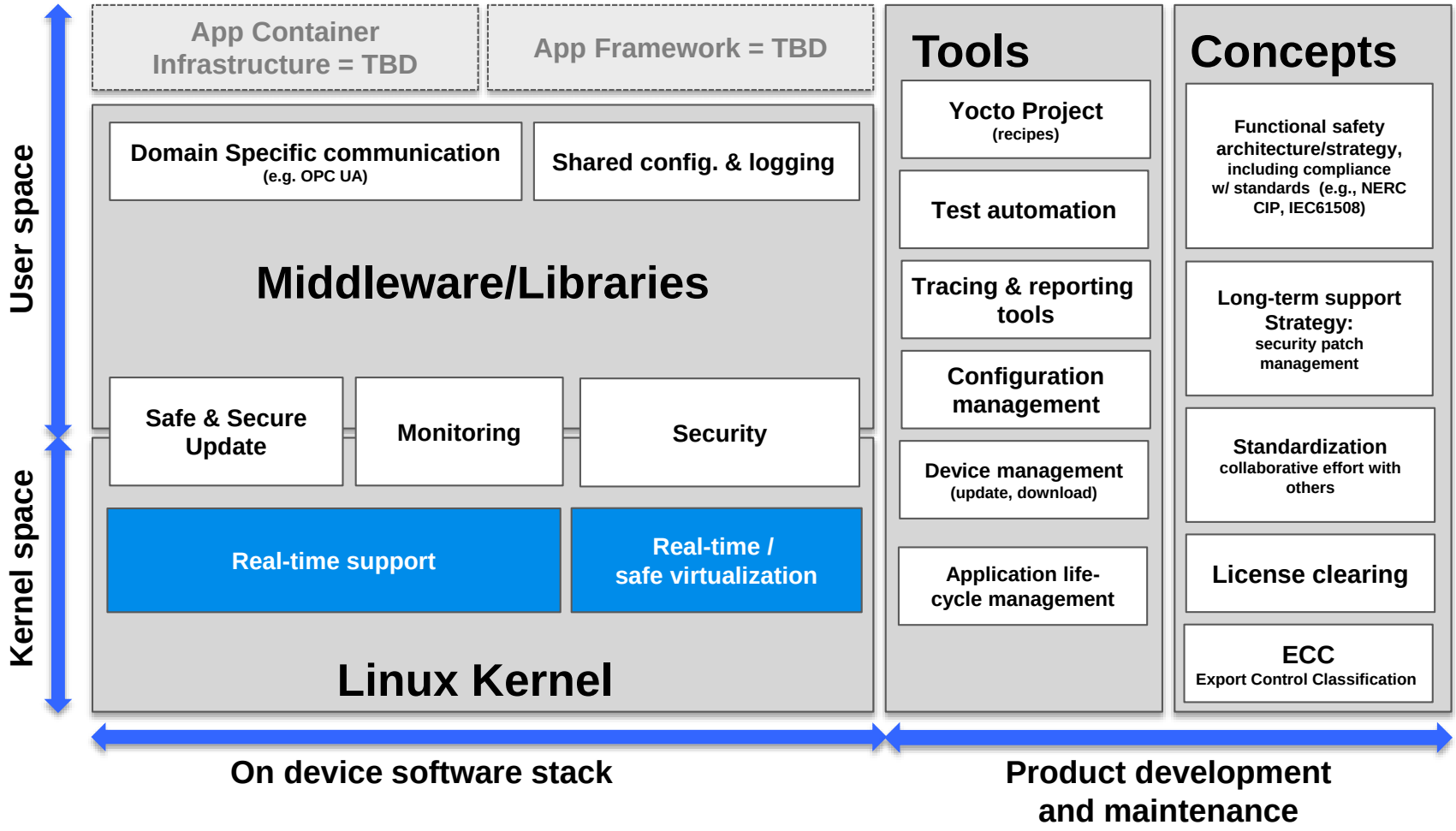
- Start from working the common HW platform, like a PC
- Later extend it to smaller/low power devices.

[n] **Device class no.**

Platform Building Blocks



Platform Building Blocks



Requirements: Real-time performance

- **Typical Latency**
  - 100μsec - 1msec response time
  - 100msec network communication frequency
  - 5msec in control frequency
- **Number of I/Os**
  - Over 10 I/O cards, and 30K in/out-puts
- **Resource management**
  - CPU consumption
  - Memory consumption
  - Coupled with container technology

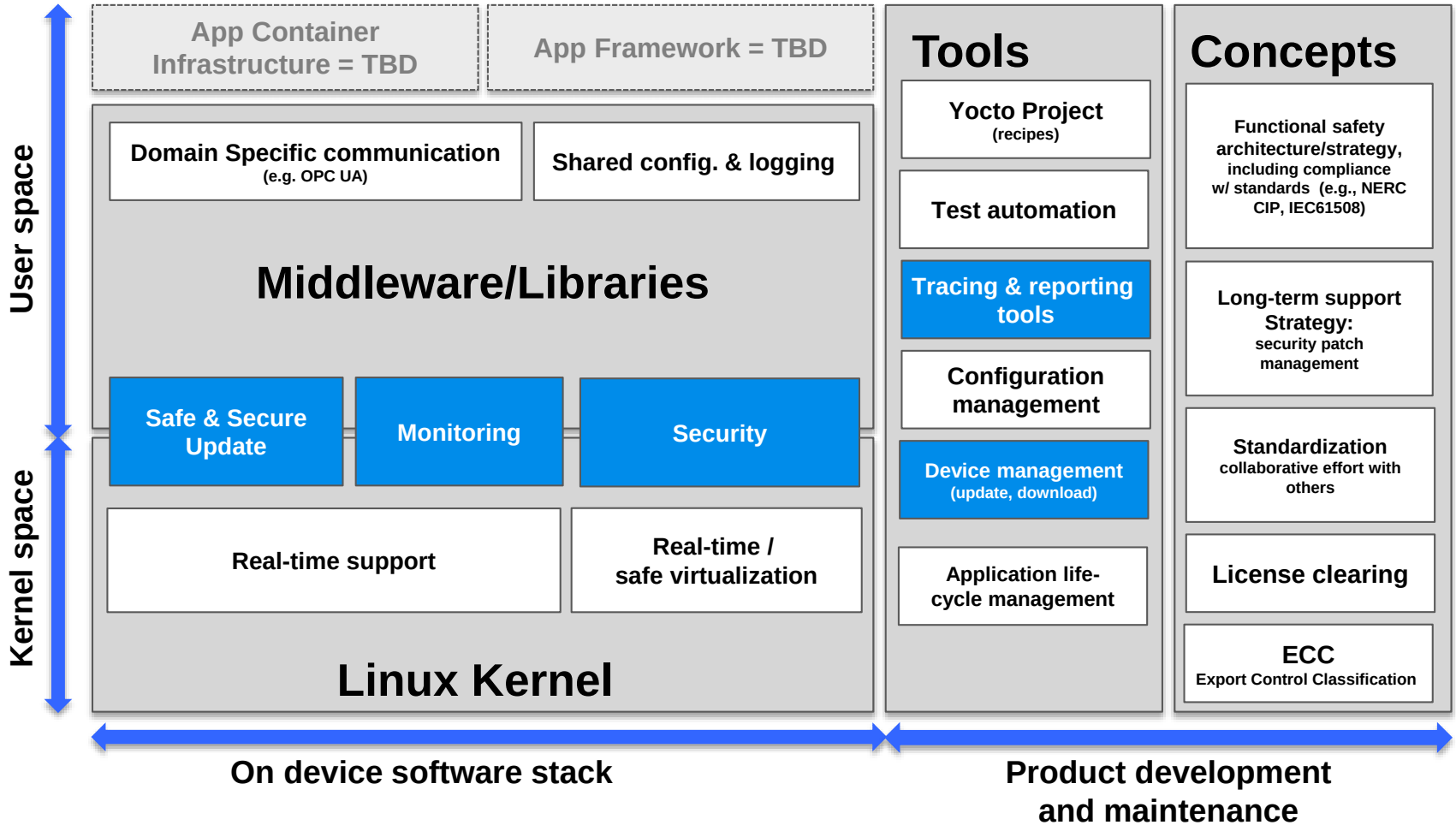
- **Related activities**
  - Preempt-RT

Requirements: Virtualization

- **Real-time safe virtualization**
  - Multi OS approach (Run with other RTOS beside the Linux)
    - E.g. Jailhouse, SafeG
  - Virtual machine
    - Real-time hypervisor enhancement (KVM)
- **Real-time OS API support**
  - E.g. Xenomai

- **Related Activities**
  - KVM
  - Jailhouse
  - SafeG by TOPPERS Project
  - Xenomai
  - V2lin

Platform Building Blocks



Requirements: Security

- **Access / execution control**
  - Access Management (SE Linux/SMACK)
  - Anomaly-based prevention systems
- **Network security**
  - Firewall technology
  - Untrusted activity detection
  - One-Way gate way (Date Diode)
  - Non-IP network
- **Pervasive Crypto**
  - Consistent standard cryptographic primitives for all core components
- **Trust authority with updated information**
  - Service that aggregates the security status (tractability) of nodes in the network and validates certificates
- **Test cases for certification**
  - E.g EDSA IEC62443

- **Related activities**
  - Linux security module
  - EDSA

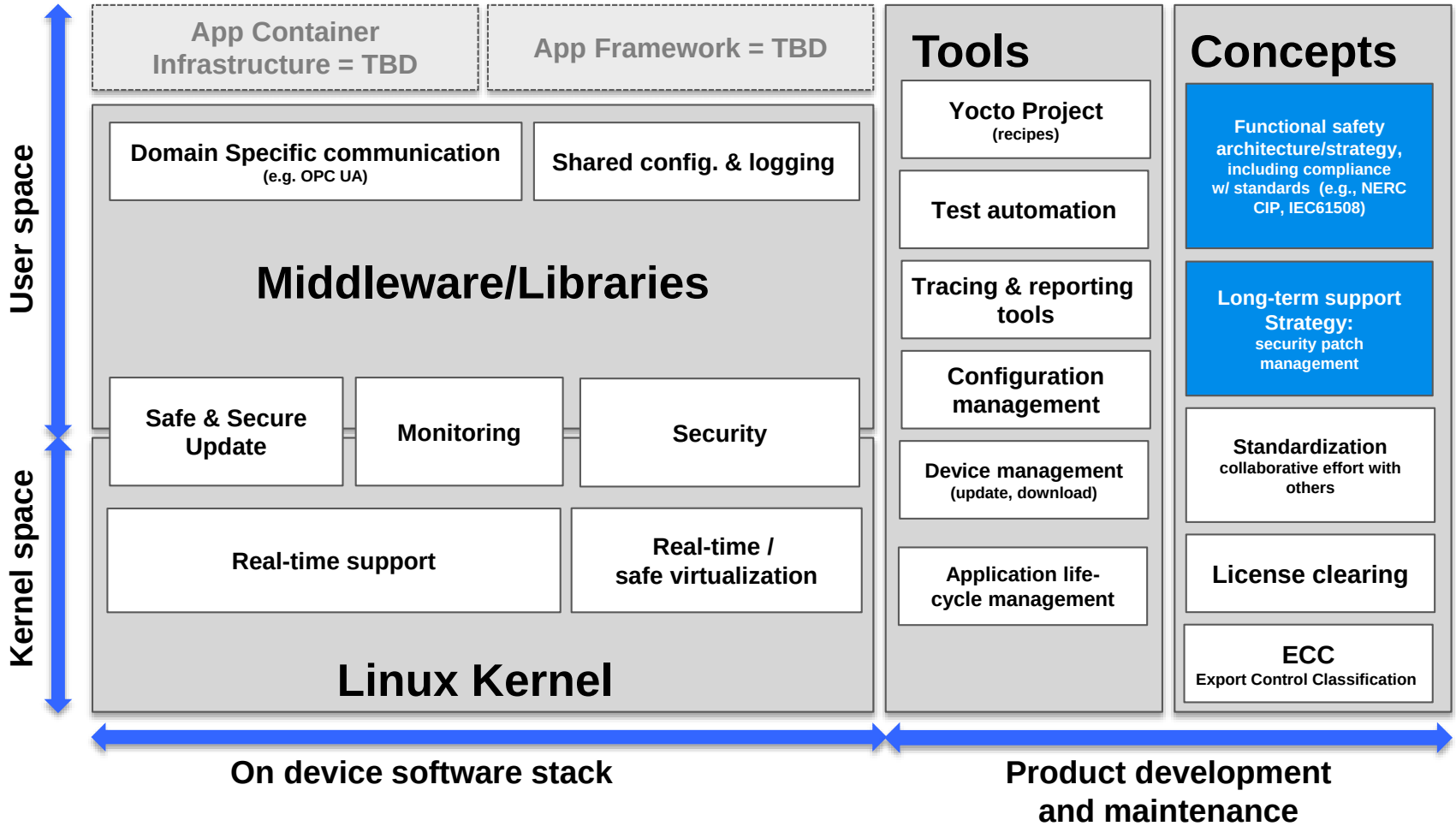
Requirements: Reliability enhancements

- **High availability**
  - 24/7 operation support
  - Failover in less than 5msec
  - Live patching with deterministic behavior
  - System health monitoring
- **Framework for failure detection and recovery**
  - Hardware error detection
    - Error detection (CPU/Memory/BUS etc)
    - Error record (trace/Panic Log/Crash dump)
  - Degeneration operation support
- **Verification test cases**

Requirements: Update / Deployment

- **Hardware update mechanism**
  - E.g. I/O card hot swap
- **Software deployment**
  - Application deployment and update mechanism (device part)
  - Firmware update
  - Device management, server side backend
- **Related activities**
  - Livepatch

Platform Building Blocks



Requirements: Long-term support

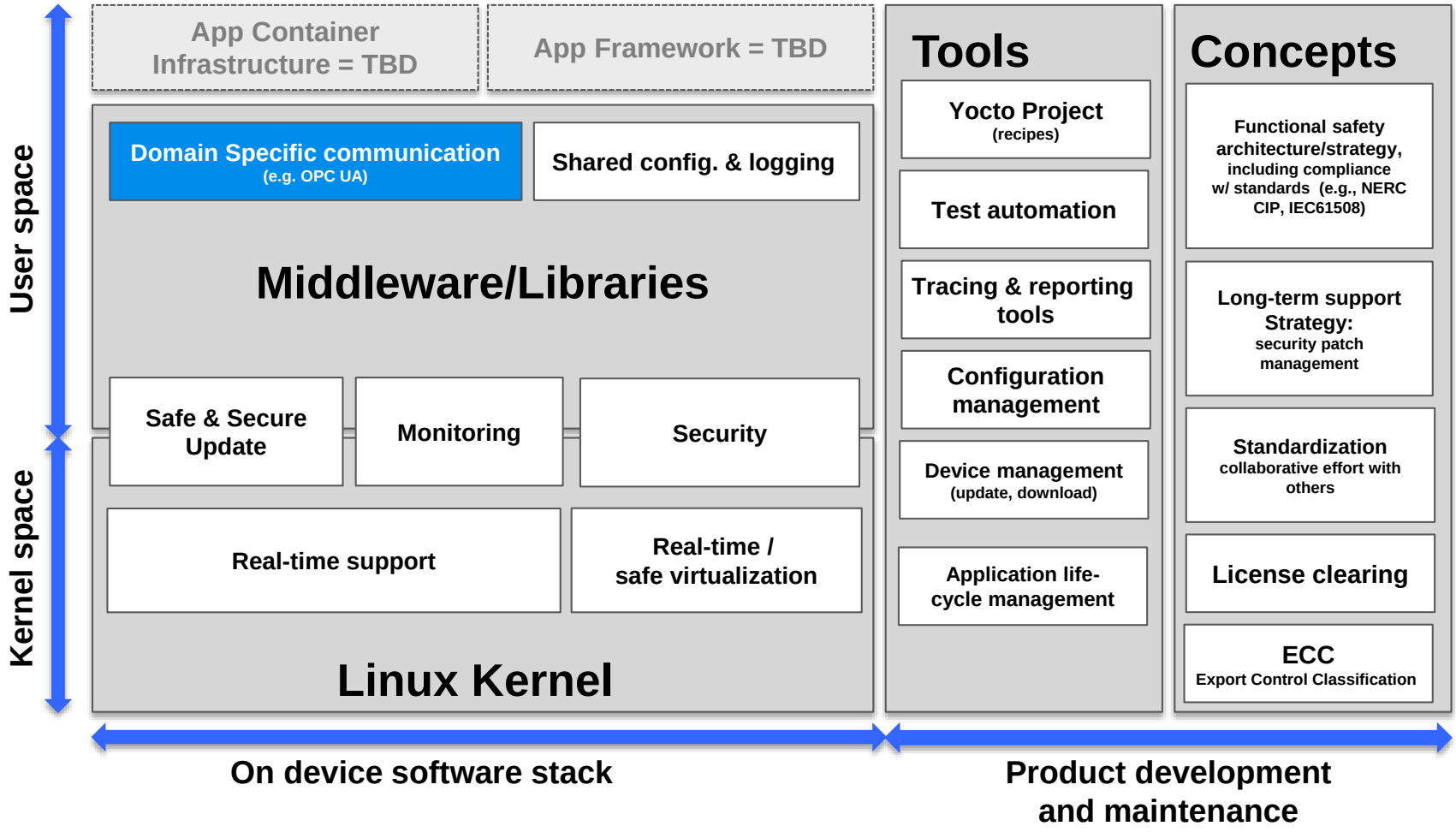
- **Very long term support (e.g. more than 15 years)**
  - Patch management tools
  - Mainly focus on security fixes
- **Migration support**
  - Enable old Linux drivers
  - Compatibility evaluation between current and new environment
  - Test cases required to ensure it

- **Related activities**
  - Long Term Support Initiative (LTSI)
  - LTSI Testing Project
  - Driver backport

Requirements: Functional safety

- **IEC61508**
  - Development process
  - SILx Linux kernel (e.g SIL2, SIL3, SIL4)
  - SILx VM
- **Monitoring Support**
  - Non-intrusive system health monitoring
- **Related activities**
  - SIL2LinuxMP
  - Jailhouse

Platform Building Blocks



Requirements: Communication stacks

- **IoT middleware**
  - AllJoyn
  - IoTivity
  - OM2M
- **Domain specific communication**
  - ZigBee
  - AVnu
  - ECHONET (might be ECHONET Lite)
  - Other industrial standard protocols
    - E.g. Real-time Ethernet

Covered topics and related projects

To be specified / implemented by CIP

Integration / cooperation

Middleware / Tools

Communication Stacks for IoT
- AllJoyn
- IoTivity
- OM2M
- …

Domain specific communication
- ZigBee
- Avnu
- ECHONET
- Industrial specific protocols

Application support
- App Framework
- HMI Framework

Deploy and update mechanism
- FW update
- App deploy
- Device manager

Safety
- Health monitor

Toolchain
- CIP TCK tests
- Yocto Project

Configuration/Device management
- Self-config
- Auto config

Testing
- LTP
- kselftest
- LTSI test
- CIP TCK tests
- Integration tests

Linux Kernel

Update mechanism
- Live patching
- Safe FW update

Real-time support
- Integration with non-RT apps
- Real-time capable GPGPU
- FPGA enhanced real-time
- PREEMPT-RT
- SIL2LinuxMP (OSADL)

Security
- LSM
- Anomaly-based prevention
- SELinux

Isolation mechanism
- LXC
- Cgroups

Funcional Safety
- SIL3 support

Monitoring / Tracing
- Ftrace
- ktap
- Error detection
- RAS

Heterogeneous Computing
- Jailhouse
- SoC FPGA

RTOS

(Out of scope)

Virtualization / Dual kernel

Real-time support
- Xenomai

Real-time Safe virtualization
- Jailhouse
- SafeG

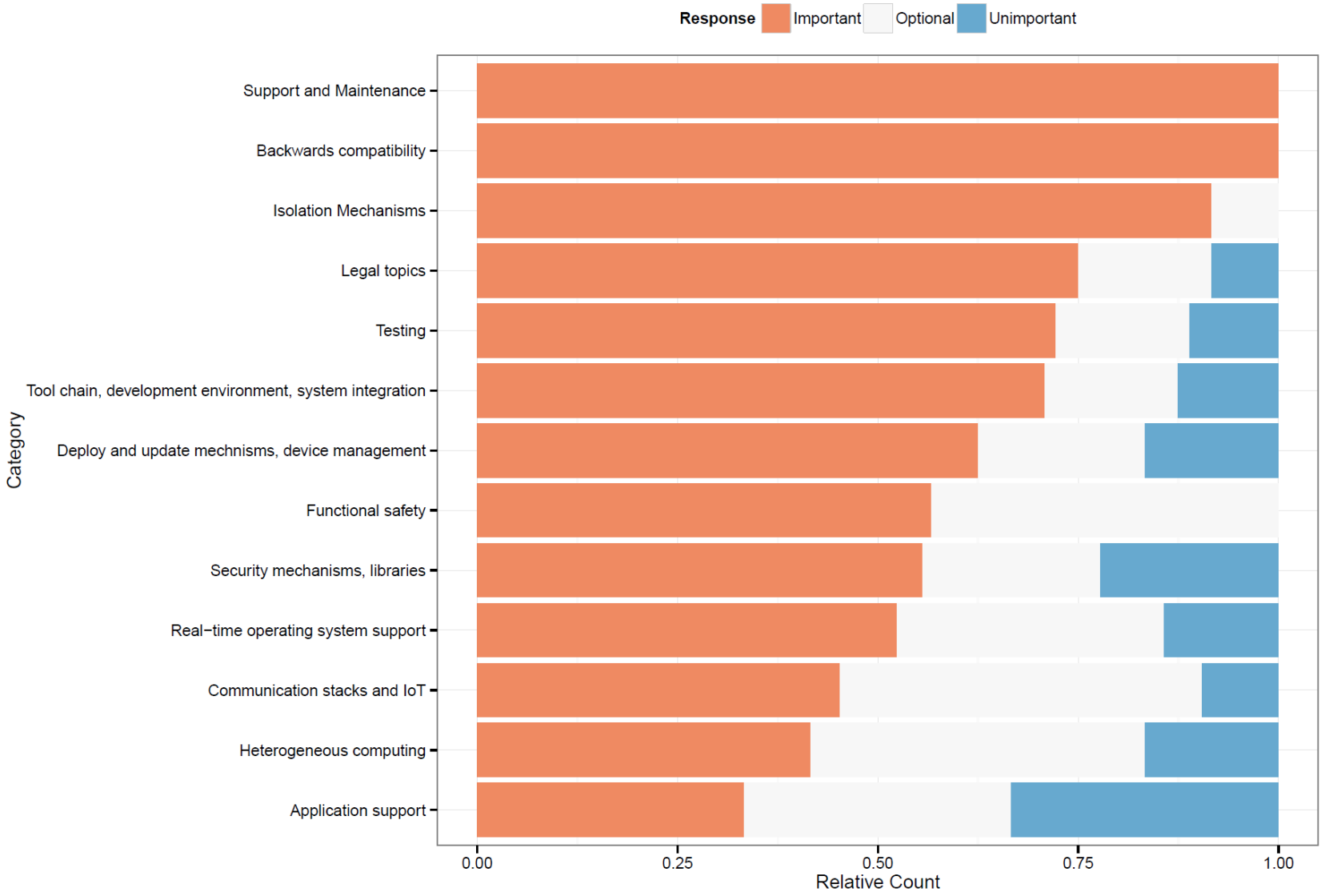
General topics

Development process
- SIL2 support
- SIL3 support

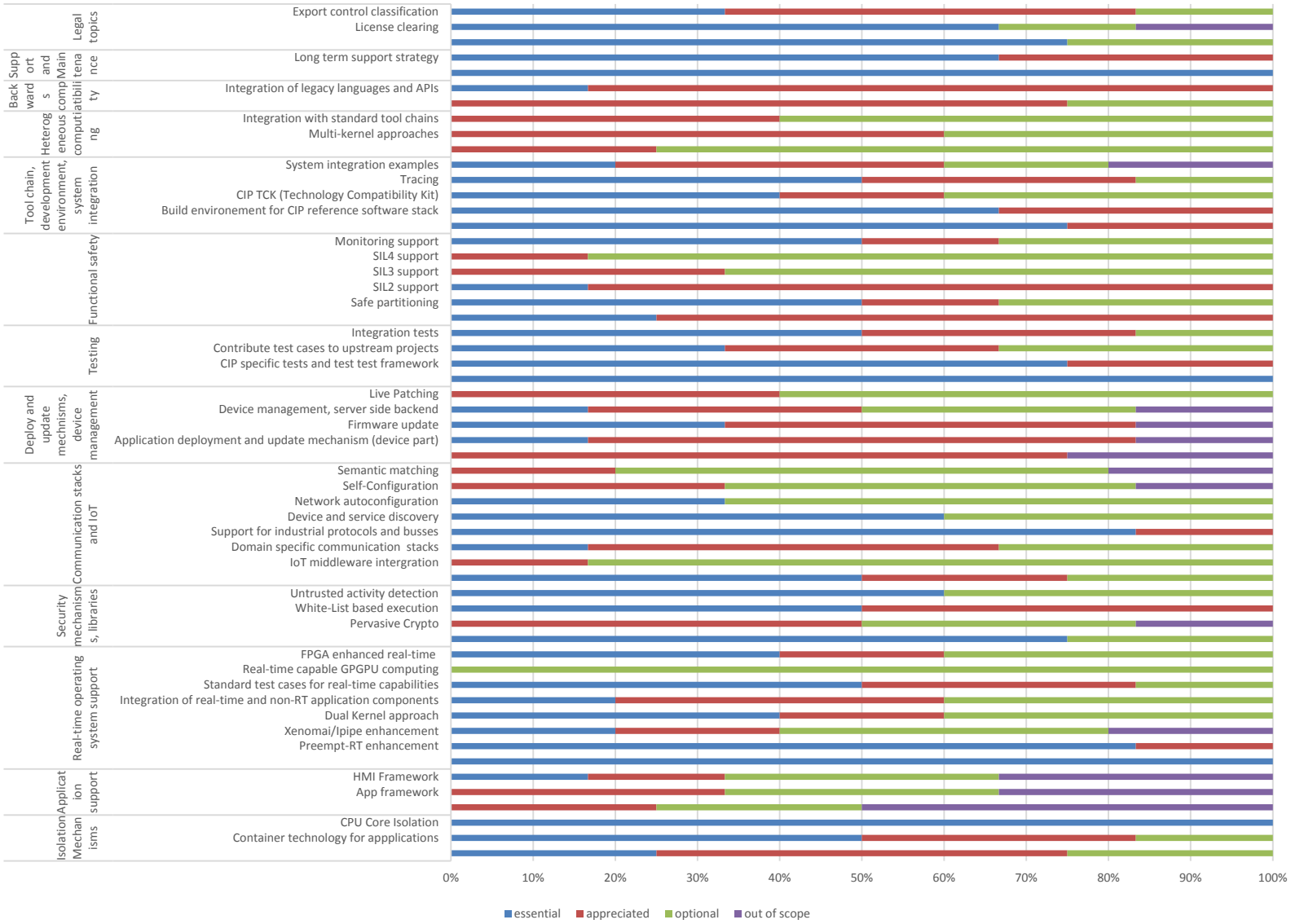
Support
- VLTS
- Backwards compatibility

Legal topics
- SPDX
- FOSSology
- License Clearing
- Export Control

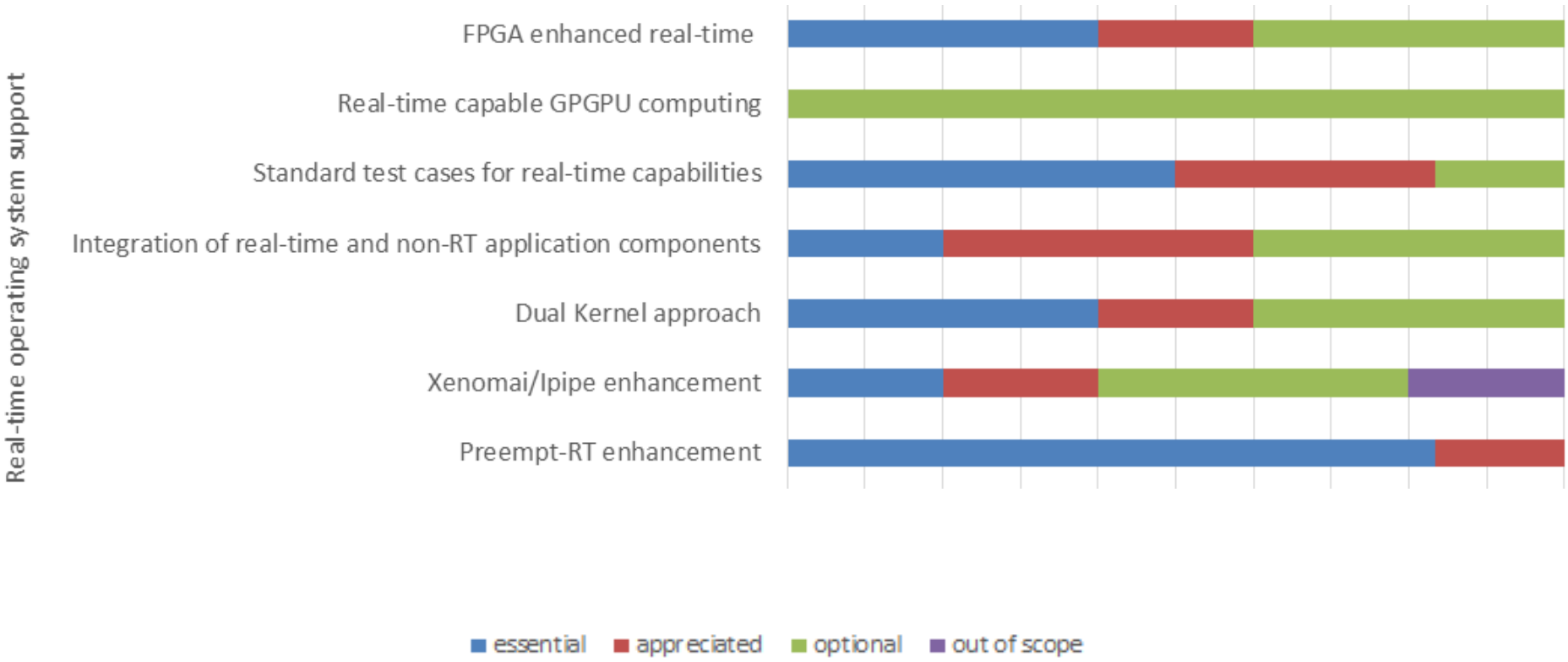
An example of topic prioritization



Detailed prioritization



Detailed prioritization of Real-time support



What's next?

- **Our current activities**
  - Collecting topics for civil infrastructure
  - Topic prioritization
  - Discussing with the Linux Foundation regarding organization
  - Have conference calls with Linux Foundation and companies
  - Have F2F meetings at Linux Foundation's conference

- **Looking for more participating companies**
  - Civil Infrastructure related vendors
  - Silicon vendors
  - Tool vendors
  - …

Please join!

- **Any comments and suggestions are welcome**

- **Contact information**
  - To get the latest information, please send an email to the following address:
    - Noriaki Fukuyasu fukuyasu@linuxfoundation.org
    - Urs Gleim urs.gleim@siemens.com
    - Yoshitake Kobayashi yoshitake.kobayashi@toshiba.co.jp
    - Satoshi Oshima satoshi.oshima.fk@hitachi.com

Questions?

Thank you