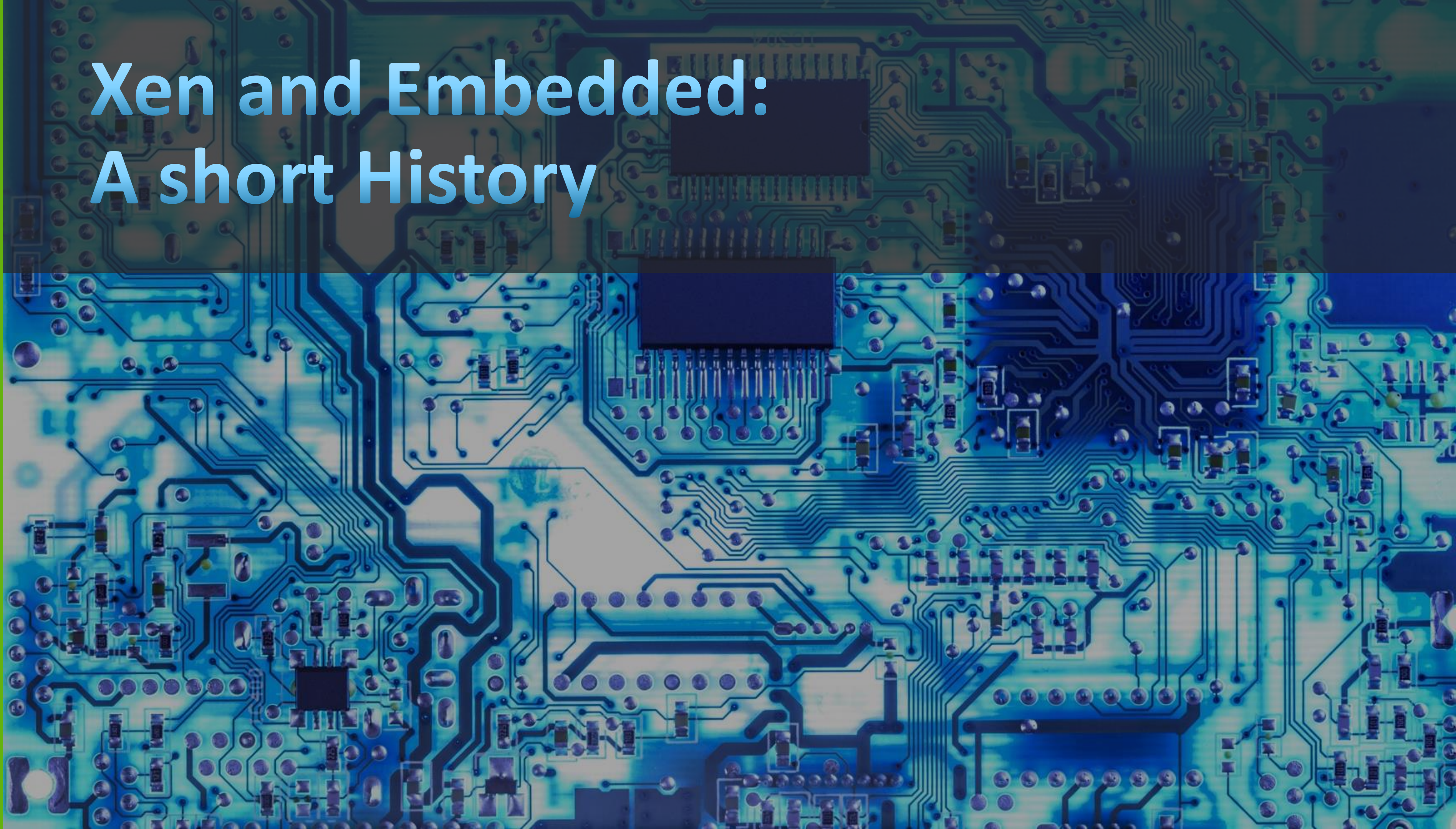
Xen Project
FuSa Overview

Elisa Workshop 2019, Cambridge UK

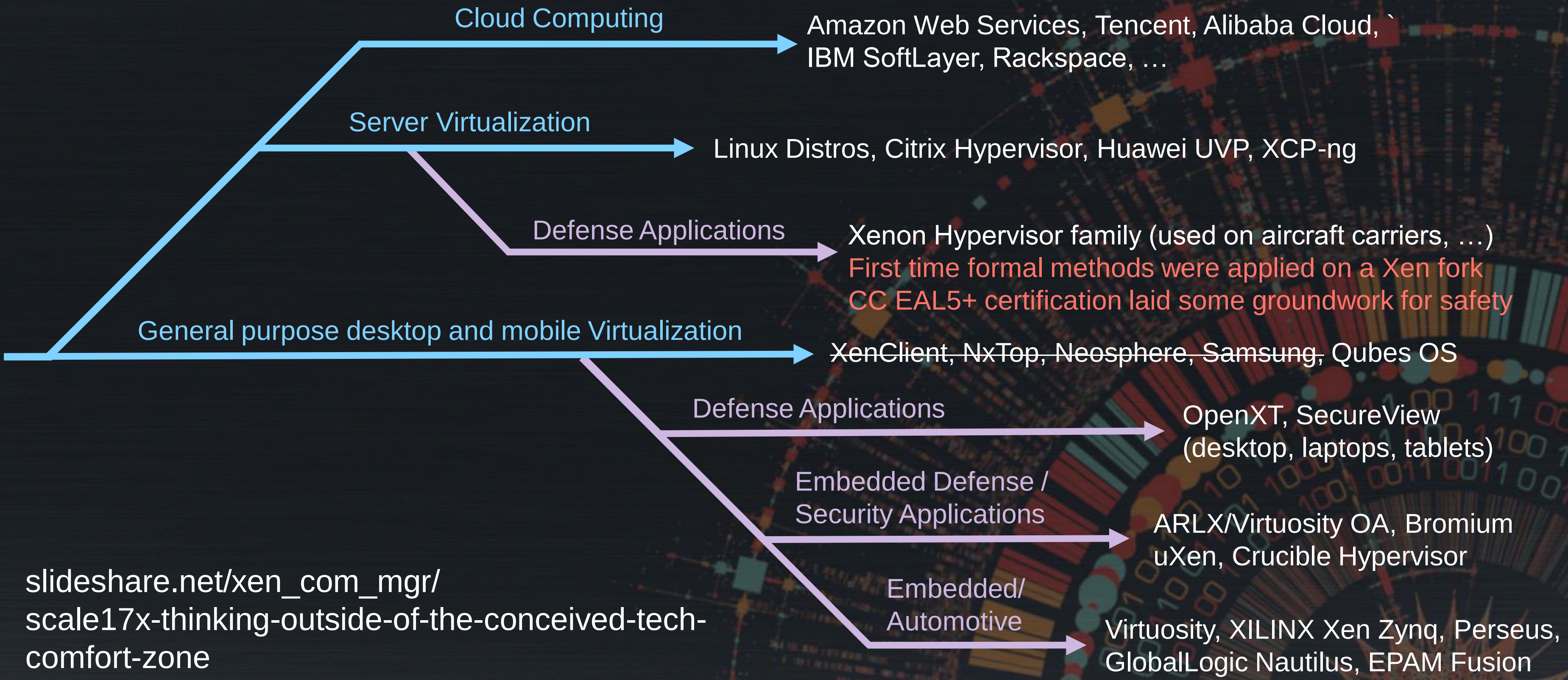
Lars Kurth
Community Manager, Xen Project
Chairman, Xen Project Advisory Board

IRC lars_kurth

Xen and Embedded:
A short History



Xen Ideas/Product Genealogy



Cloud Computing

Amazon Web Services, Tencent, Alibaba Cloud, `
IBM SoftLayer, Rackspace, …

Server Virtualization

Linux Distros, Citrix Hypervisor, Huawei UVP, XCP-ng

Defense Applications

Xenon Hypervisor family (used on aircraft carriers, …)
First time formal methods were applied on a Xen fork
CC EAL5+ certification laid some groundwork for safety

General purpose desktop and mobile Virtualization

~~XenClient, NxTop, Neosphere, Samsung,~~ Qubes OS

Defense Applications

OpenXT, SecureView
(desktop, laptops, tablets)

Embedded Defense /
Security Applications

ARLX/Virtuosity OA, Bromium
uXen, Crucible Hypervisor

Embedded/
Automotive

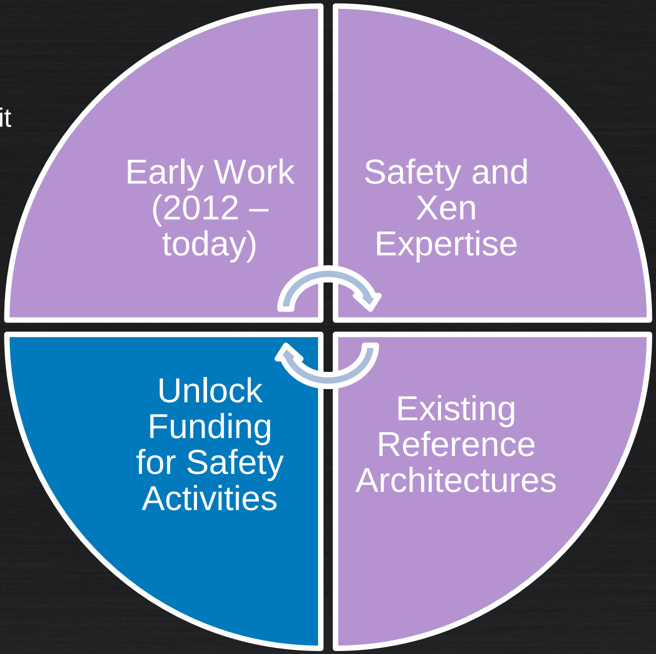
Virtuosity, XILINX Xen Zynq, Perseus,
GlobalLogic Nautilus, EPAM Fusion

slideshare.net/xen_com_mgr/scale17x-thinking-outside-of-the-conceived-tech-comfort-zone

Enablers for a Xen Safety Story

- Study by DornerWorks to establish feasibility of whether Xen on Arm could be certified to DO 178b Level A ➜ Cost matrix & Product family (ARLX, Virtuosity OA)
- Study by HORIBA MIRA to assess whether it is possible to safety certify a subset of the Xen Project ➜ EPAM ref platform
- Fill functional gaps (RT, reduce code size, configurability, …) ➜ Reference platforms

- NASA funds Dornerworks to integrate the Xen Project Hypervisor into NASA's new High Performance Space Computing Platform (HPSC)
- Significant funding from a group of vendors to re-write Xen on Arm port for embedded likely (originally designed for servers)
- Side channel attacks ➜ Re-architect Xen core (AWS), use of TLA+ (Citrix)
- Other funding routes being considered (e.g. HORIZON 2020, US grants, …)


Early Work (2012 – today)
Safety and Xen Expertise
Unlock Funding for Safety Activities
Existing Reference Architectures

- Multiple consultancies which know the Xen codebase and various safety standards (DornerWorks, StarLabs.io and EPAM which is nascent)
- All have experience in upstreaming functionality to Xen
- Today: DO 178 centric

- **DornerWorks:** OpenGroup FACE certified Virtuosity OA (military)
- **XILINX:** generic embedded stack
- **EPAM:** automotive stack
- But: all open source, but not all is up streamed
- Some use in production:
  In a non-safety context
  In safety contexts where safety can be isolated outside of Xen

Feature Examples specific to Embedded

Schedulers: ARINC, RTDS, Null and other real-time support

Laid the foundation for embedded use-cases and use of Xen as a partitioning HV

Low latency and real-time support

A minimal Xen on Arm Configuration

< 50 KSLOC of code for a specific HW environment

PV drivers (and in future virtio drivers) and GPU mediation for rich IO

Available in various upstreams

OP-TEE virtualization support

Both in Xen and in OP-TEE

Dom0less Xen

For now: allows booting VM's without interaction with Dom0, but Dom0 still exists

2020: an architecture without a Dom0 and/or an RTOS as Dom0

Feature Examples specific to Embedded

Schedulers: ARINC, RTDS, Null and other real-time support

Laid the foundation for embedded use-cases and use of Xen as a partitioning HV

Low

A r

< 5

PV … O

Ava

OP

Bot

Do

For now: allows booting VM's without interaction with Dom0, but Dom0 still exists

2020: an architecture without a Dom0 and/or an RTOS as Dom0

Key Point:

Xen on Arm, turned out to be a great open source hypervisor for embedded and mixed-criticality use-cases in theory

Despite having been designed for servers!

Safety Certification
The beginning of the journey

FOSS SW and Functional Safety

Requires major changes to the software

Requires tools, infrastructure and expertise

Funding ↔ Confidence

Requires changes in how FOSS projects work

Until recently: assumption was that the two worlds cannot work together

Community Challenges ↔ Trust & Confidence

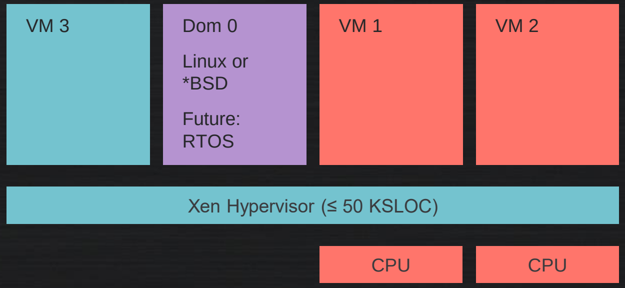
Tooling has a huge impact on Community Challenges

We need tools (ideally FOSS tools) that fit into our Git and CI workflow

Tools Challenges ↔ Funding

Mixed Criticality case

Dom0less VMs (today)



VM 3

Dom 0

Linux or *BSD

Future: RTOS

VM 1

VM 2

Xen Hypervisor (≤ 50 KSLOC)

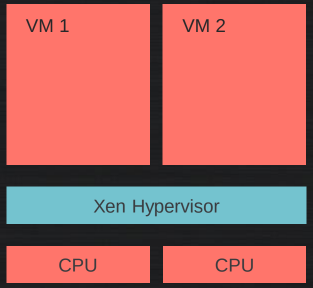
CPU

CPU

Dom0less VMs loaded by uBoot and booted by Xen (not Dom0), pinned to a CPU via the Null scheduler and I/O handled by device assignment

Dom0 completes boot after VM 1 and VM 2. Static set-up

True Dom0less (2019/20)



VM 1

VM 2

Xen Hypervisor

CPU

CPU

Ongoing work to fully implement true Dom0less for small systems

- Shared memory and interrupts for VM-to-VM communications
- PV frontends/backends drivers for Dom0-less VMs

Dom0less initial safety certification scope

slideshare.net/xen_com_mgr/elc2019-static-partitioning-made-simple

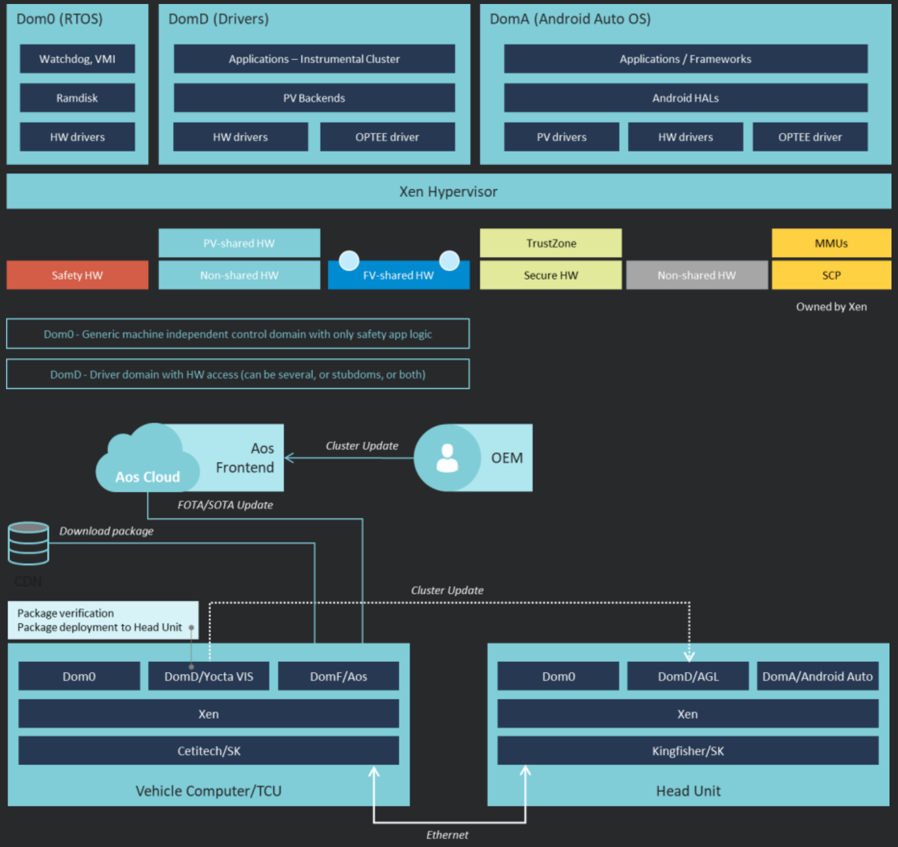
Automotive Case

epam

AUTOMOTIVE

Mix Safety Digital Cockpit

In-Vehicle Computer



Dom0 (RTOS)
- Watchdog, VMI
- Ramdisk
- HW drivers

DomD (Drivers)
- Applications – Instrumental Cluster
- PV Backends
- HW drivers
- OPTEE driver

DomA (Android Auto OS)
- Applications / Frameworks
- Android HALs
- PV drivers
- HW drivers
- OPTEE driver

Xen Hypervisor

Safety HW

PV-shared HW

Non-shared HW

FV-shared HW

TrustZone

Secure HW

Non-shared HW

MMUs

SCP

Owned by Xen

Dom0 - Generic machine independent control domain with only safety app logic

DomD - Driver domain with HW access (can be several, or stubdoms, or both)

Aos Cloud

Aos Frontend

Cluster Update

OEM

FOTA/SOTA Update

Download package

CDN

Package verification
Package deployment to Head Unit

Cluster Update

Vehicle Computer/TCU
- Dom0
- DomD/Yocta VIS
- DomF/Aos
- Xen
- Cetitech/SK

Head Unit
- Dom0
- DomD/AGL
- DomA/Android Auto
- Xen
- Kingfisher/SK

Ethernet

AUTOMOTIVE GRADE LINUX

Xen Project

androidauto

FuSa SIG with Workstreams

Subgroups meet at least every other week. Partly resourced

Community Reps

Lars Kurth (chair and project mgmt)
George Dunlap (committers)

Assessors

RESILTECH
exida
TÜV SÜD ISO 9001
A HORIBA COMPANY MIRA

Stream Owners and Implementers

Lars Kurth
THE LINUX FOUNDATION
epam
XILINX
RESILTECH
arm
RENESAS

Other Members

ADIT

2-day workshop in March 2019

Create a understanding between the community and industry

Terminology, Concepts, etc.
How safety certification works: look at different standards, routes, requirements
Explain assets and processes

Establish community “red lines”

Principles the community can agree to or would object to
What level of change would be acceptable
Identify potential obstacles

High Level Agreements

Split development model with an open and a closed part

Everything that is valuable to the wider community **ideally** in the open part,
e.g. documentation, **some** tests, traceability, automation and infrastructure,….

Everything that creates code churn if it wasn't open as much as possible:
e.g. coding standards (MISRA)

Changes to the development workflow have to be kept minimal

There must be a benefit the community
Otherwise the community wont carry

There are long-term implications for the community

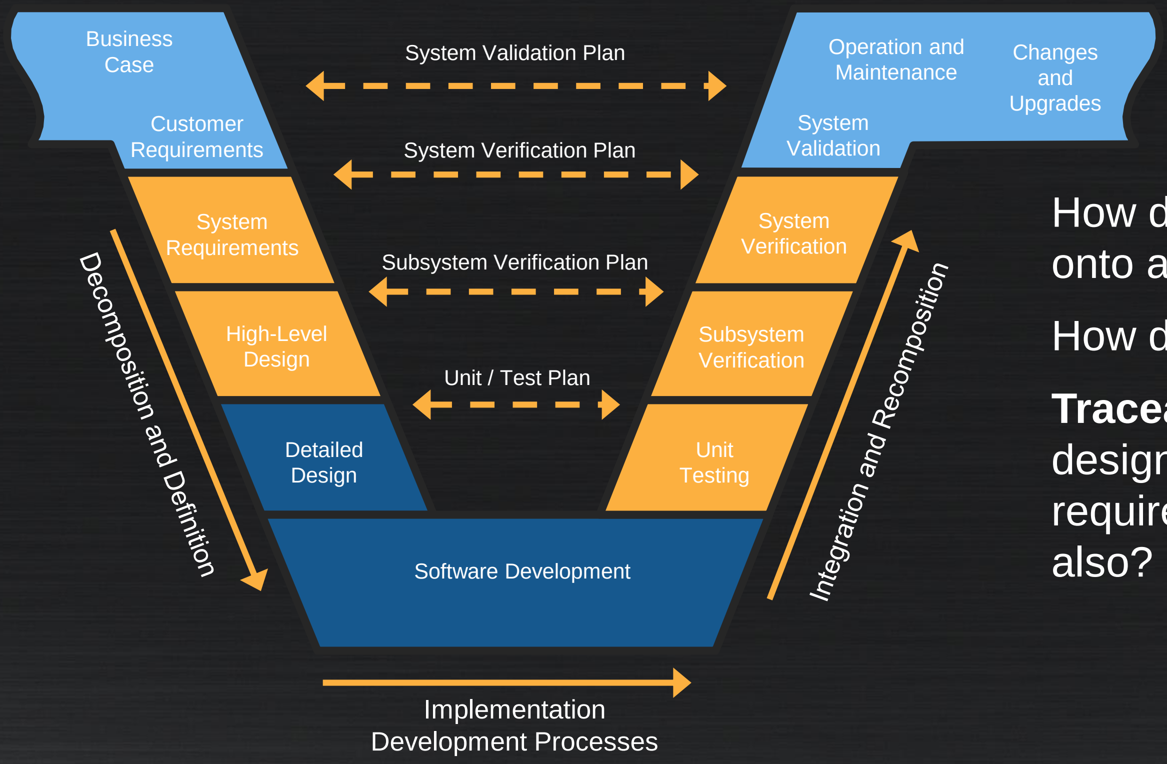
Make-up, scalability, decision making, conflicts – need to be managed
No major new barriers for contributors can be introduced

Goal:
significantly reduce the cost for users to safety certify Xen derivates

Share as much burden as possible by collaborating upstream

Examples of Challenges that need to be overcome

Development Process and Traceability

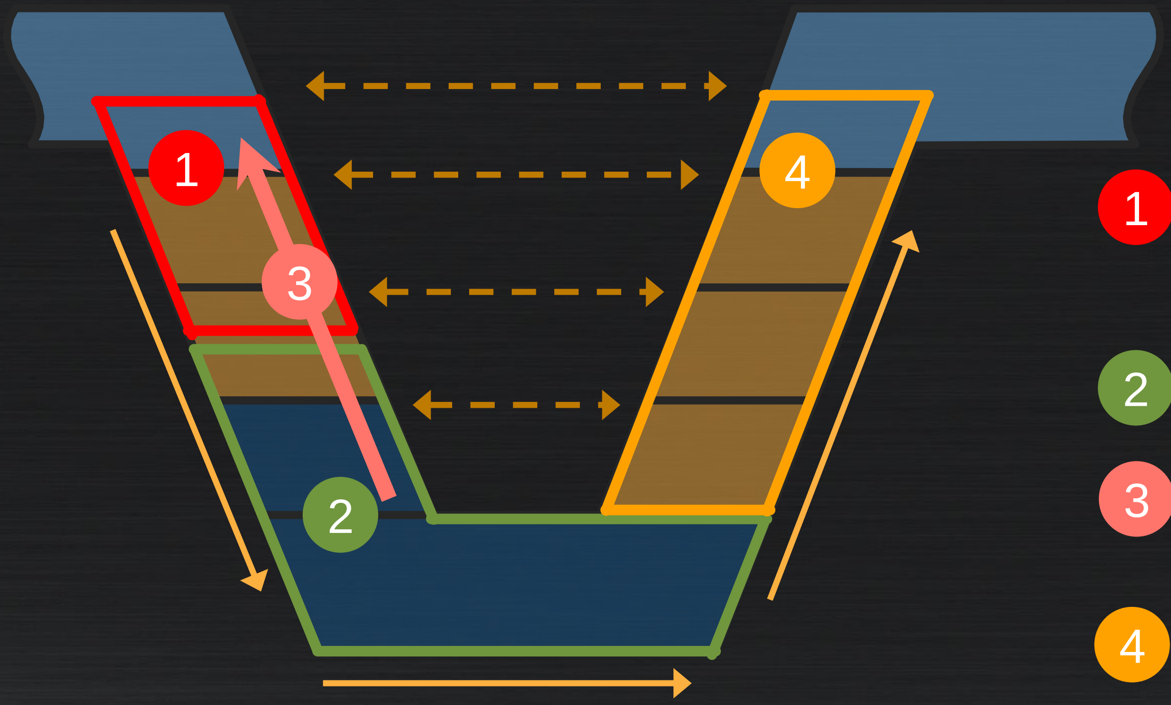


How do you map this onto a FOSS development process?

How do you get community buy-in?

Traceability: how do you prove that design and architecture satisfies requirements and tests verify these also?

What you normally have in FOSS is …



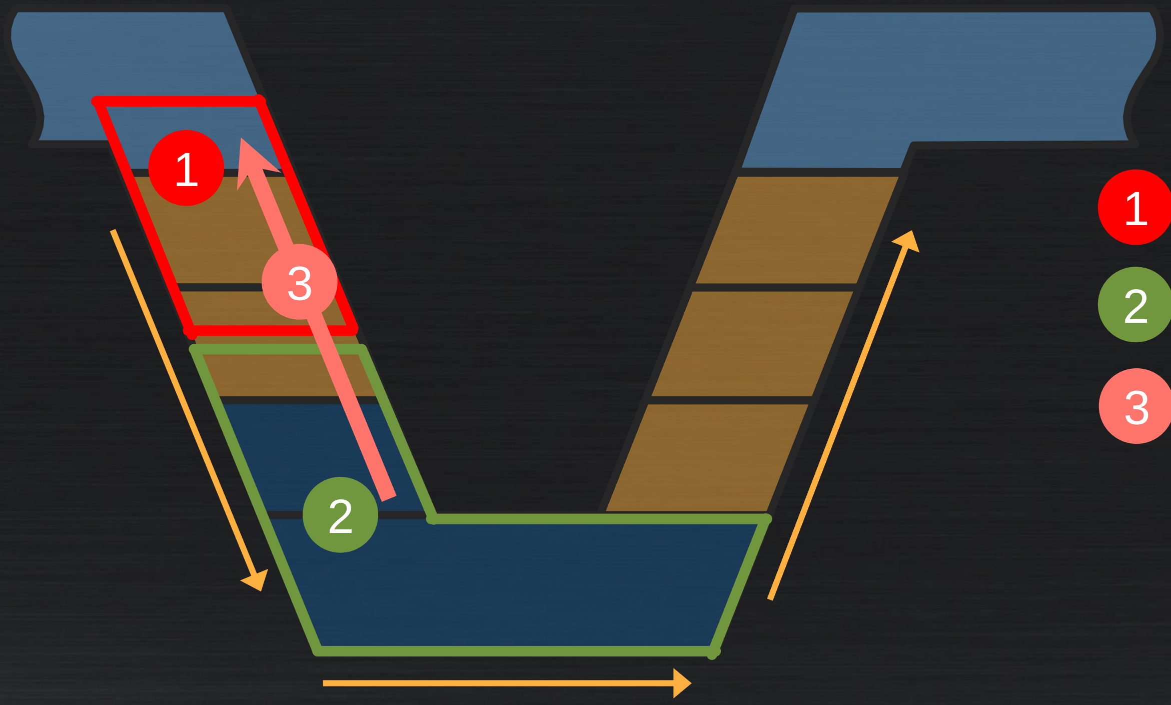
1 Not at at all, or outside
Not a huge effort to retrofit
Valuable for developers & users
Does not change often for a Hypervisor

2 Frequently as good or better
than proprietary. Process discipline

3 Not at all. Difficult to maintain
manually. Should not change that
often

4 A subset of this usually exists, but
typically tests **code, not requirements/specifications**.
That’s the most expensive part to
address.

What must be upstream: all key inputs …



1 **Documented Requirements**

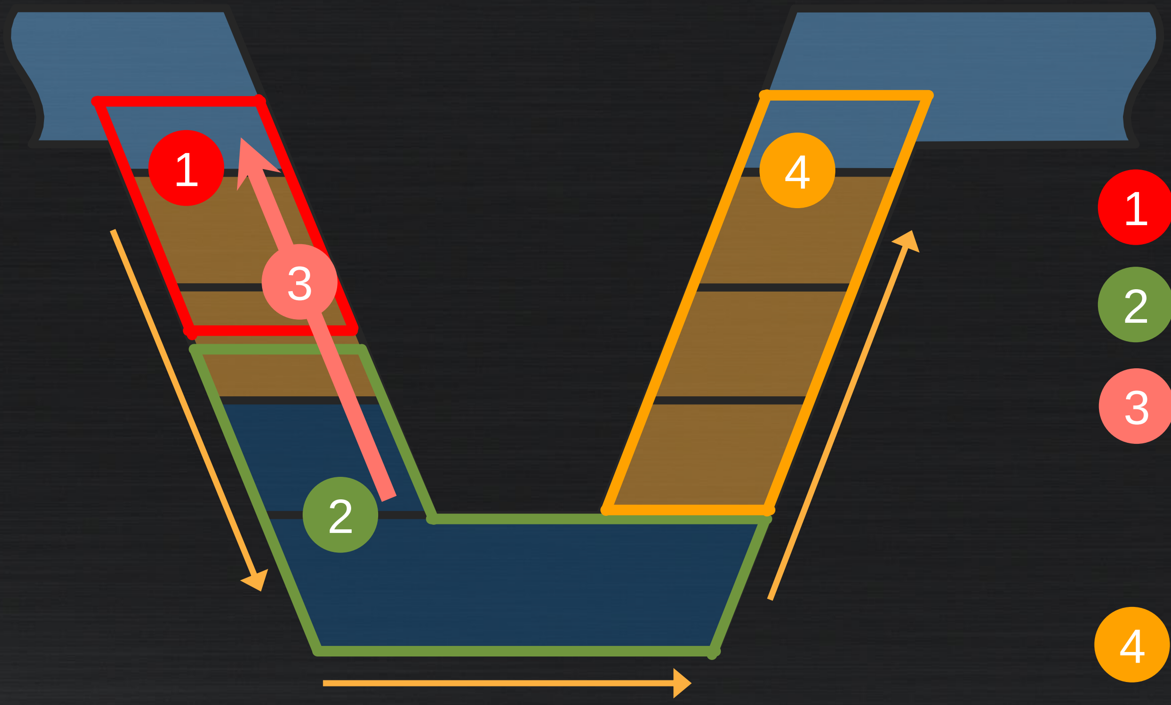
2 **Design, Architectural and API documentation**

3 **Traceability info:**
Between requirements
Between requirements and other docs
Between requirements and code

With appropriate tooling and Information Architecture this can be done in a git-workflow

Candidate tool: DOORSTOP

What must be upstream: all key inputs …



1 **Documented Requirements**

2 **Design, Architectural and API documentation**

3 **Traceability info:**
Between requirements
Between requirements and other docs
Between requirements and code

4 **Validation:**
Can be outside of upstream
Needs a feedback loop to deal with breakage – like OpenStack 3rd party CI

Community Challenges: MISRA C

Picked MISRA C as an example, because …

it is representative of the hardest type of community problems that you should expect if you look at safety certification

Picked hardest and controversial rules to see what would happen!

We did not expect to succeed !

We got stuck early on

MISRA C spec is proprietary

Rule text cannot be copied into a posted patch series →
lack of clarity, lack of rationale: leading to unnecessary debate

Interactions w compilers, HW, assembly code problematic

Ended up with 11 iterations and man weeks of review effort

Bike shedding and strong opinions

Some rules will create a flame-war if there is a single argumentative maintainer

E.g. MISRA C:2012, 15.7
"if ... else if" constructs should end with "else" clause

```
if (x == 0) {
  doSomething();
} else if (x == 1) {
  doSomethingElse();
} else {
  error();
  /* or justification why no action is taken */
}
```

Deviations and Scalability

Possibility of MISRA C Deviations encourage arguments

Deviations: justification of a class or instance of non-compliance
Deviation Permits: previously approved deviations for a use-case

An expert (assessor) is needed to advise the project on a case-by-case basis
Probably needs funding

Community Scalability

Code review process encourages too much discussion, if there is no up-front plan on how to approach a disruptive set of changes

Fix: A priori agreed strategy and plan on how to approach this

Safety Certification
Creating a credible plan …

Low customization route

Candidates: IEC 61508 or ISO 26262

Build Confidence and
Unlock Funding / solve Community problems iteratively

Chicken and egg problems

Focus on left side of V model first

While refreshing the Xen on Arm port at the same time

- Effort to identify key APIs and improve documentation (started)
- Code review map (started)

Need docs & traceability tooling story:
Ideally a cross-project standard using tools and Information Architecture
Make it easy to keep artefacts up-to-date

Does ELISA have a role in this?

CI Loop changes

Front-load CI: do as much as possible **before** code review (in progress)

Use bots and automation (in progress)

More tests in “simulated environments” – capacity problem

3rd party CI loop hooks

Coding Standards

Need more experiments: initially keep clear of MISRA

Need a process to prioritize rule implementation

Compliance tooling and reporting that fits into CI (issue: © of MISRA)

Goal: Minimize unnecessary discussion

Areas which are not yet clear

Testing and Validation

Safety management system that can coexist with generic Xen mainline development

…

Xen and Linux

Similar Development Process and Culture

Some differences in areas such as Release Management, CI Infrastructure, Vulnerability Management, Leadership team vs Dictator

Code Size and Community Size

Linux is 1-2 orders of magnitude larger

Community Make-up

Linux: dominated by cloud and server vendors

Xen: has areas which are exclusively driven by embedded vendors (aka Xen Arm) with some common code affecting all users. While x86 is cloud, server and security applications

Are there common challenges where collaborating makes sense

Backup stuff

Certification Costs: Example DO-178b

Level	Requirements	Application	Cost with Experience
DAL E	The software must exist	**Infotainment** Failure is a minor inconvenience	0.11 hour / SLOC
DAL D	High-Level Docs/Tests	**Instruments** Failure can be mitigated by operator	0.13 hour / SLOC
DAL C	Low-Level Docs/Unit Tests, Statement Coverage, and Code/Data Coupling Analysis		0.20 hour / SLOC
DAL B	Branch Coverage	**Engine Control** Failure could kill someone without warning	0.40 hour / SLOC
DAL A	Source to Object Analysis and MC/DC Coverage		0.67 hour / SLOC

Credit/Source: Dornerworks / XPDS14 - Xen and the Art of Certification.pdf

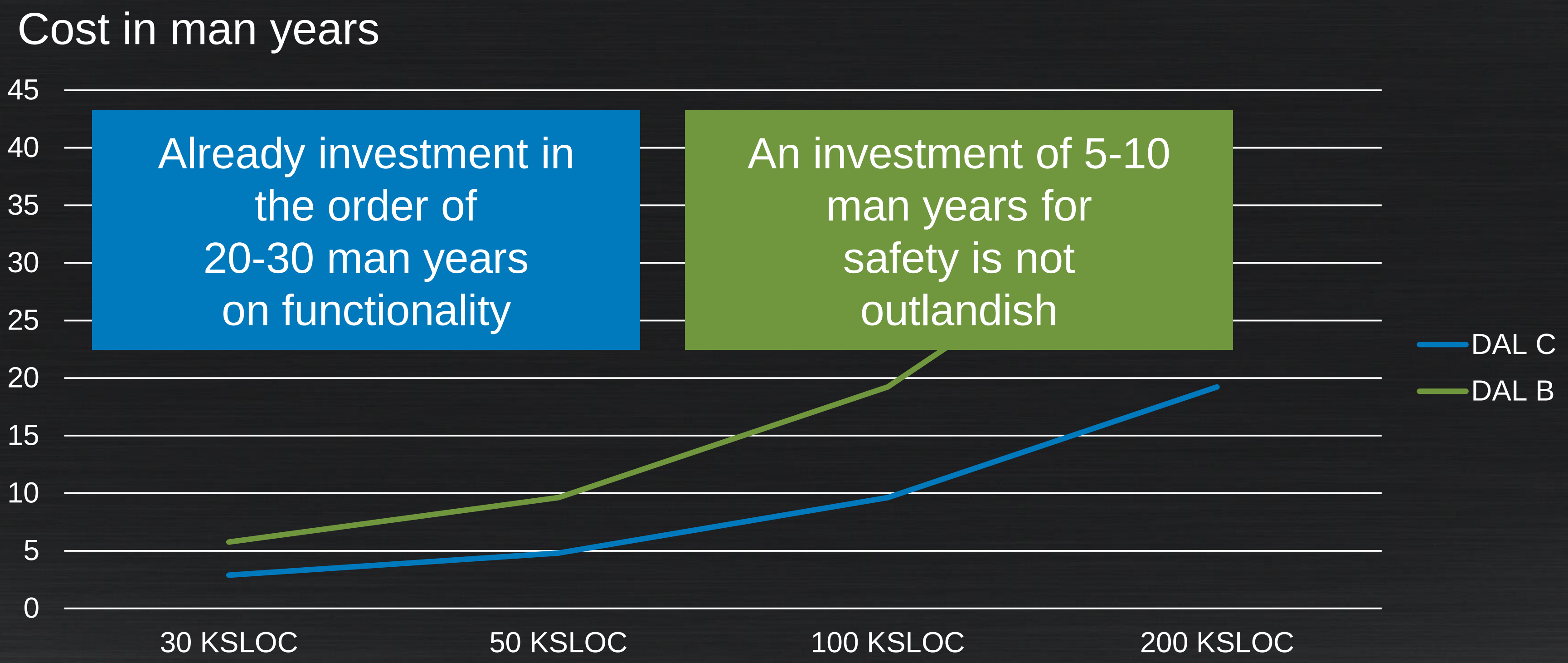
Certification Costs: Example DO-178b

Level	Requirements	Application	Cost with Experience
DAL E	The software must exist	**Infotainment** Failure is a minor inconvenience	0.11 hour / SLOC
DAL D	High-Level Docs/Tests	**Instruments** Failure can be mitigated by operator	0.13 hour / SLOC
DAL C	Low-Level Docs/Unit Tests, Statement Coverage, and Code/Data Coupling Analysis		0.20 hour / SLOC
DAL B	Branch Coverage	**Engine Control** Failure could kill someone without warning	0.40 hour / SLOC
DAL A	Source to Object Analysis and MC/DC Coverage		0.67 hour / SLOC

3-4 times as much without experience

Credit/Source: Dornerworks / XPDS14 - Xen and the Art of Certification.pdf

Certification Costs: Example DO-178



Already investment in the order of 20-30 man years on functionality

An investment of 5-10 man years for safety is not outlandish