SELECTION AND EVALUATION OF AN EMBEDDED HYPERVISOR: APPLICATION TO AN AUTOMOTIVE PLATFORM

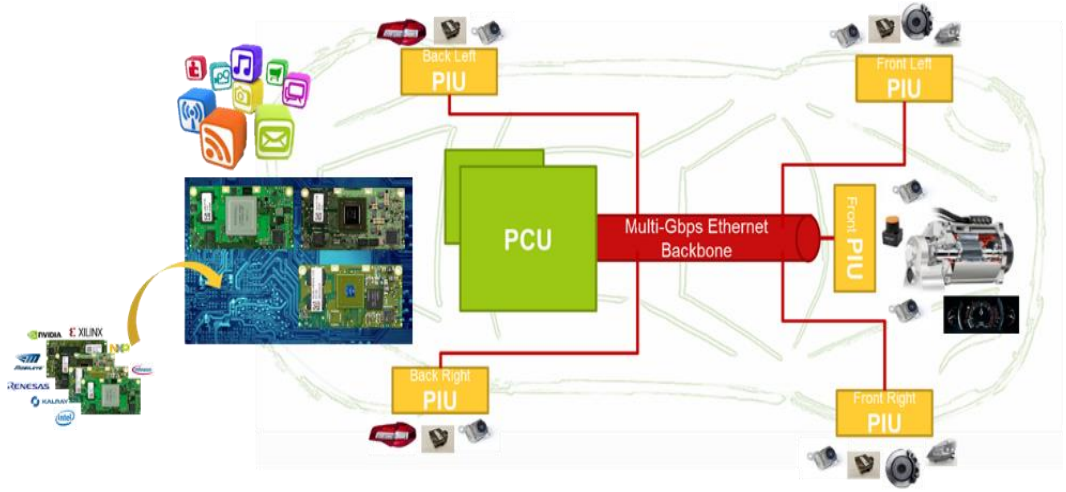
Etienne HAMELIN, Moha AIT HMID, Amine NAJI, Yves MOUAFO-TCHINDA | ERTS2020 | 30/01/2020

This work was partly funded by the Renault-Nissan alliance under the FACE & OPTEEM projects

SUMMARY

CONTEXT & MOTIVATION

- **More SW functions on fewer, high-perf SoCs**

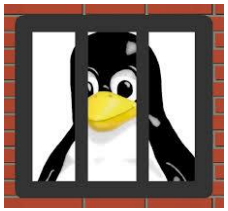


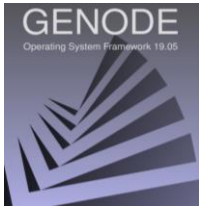
- **Centralized computing platform**
  - Common motherboard
  - Modularity via general-purpose or specialized SoCs daughter boards
  - Mother- & daughterboards host heterogeneous SW payloads from various SW suppliers

SW payload	Computation	Safety-relevant	Security-relevant
Command & control	–	+	~
ADAS	+	+	~
Multimedia, infotainment	+	–	+

CONTEXT & MOTIVATION

- OK, so we need an embedded hypervisor… but which one?





HOW TO CHOOSE?

- **So many criteria…**

	Technical/ non-technical	Quantitative / qualitaive	Objective / subjective	Required / nice-to-have	Relative evaluation effort
Hypervisor type (type I, type II, µK-based)	1	0	1	0	1
Supported CPU architectures (x86, ARM, …)	1	0	1	1	1
Supported OS (full-/para-virtualized), exposed task API	1	0	1	0	1
Memory, peripherals management scheme	1	0	1	0	2
Scheduling scheme, real-time	1	0	1	0,5	2
Performances & overheads	1	1	1	0,5	3
Supports to safety, security, lifecycle	1	0	1	0,5	2
Signs of industrial maturity: prototype or field success stories	0	0	0	1	1
Safety/security certification or qualification packages	0	1	1	1	2
Usability (incl. tools, user guidance, examples)	0	0	0	1	3
Licensing, partnership, support, business model	0	0	0,5	1	2
Price scheme	0	1	1	1	2

A RATIONAL SELECTION PROCESS

- **Scientific selection process: optimal choice**
  - For each criteria, define evaluation method, relative weight: w_j
  - For each hypervisor & criteria, evaluate $c_{i,j}$
  - Select best hypervisor: $s^* = \arg\max \sum_j w_j \cdot c_{i,j}$

    Easy, right?
- **But**
  - n hypervisors, m criteria $\Rightarrow n \times m$ evaluations!
  - weights tuning very subjective

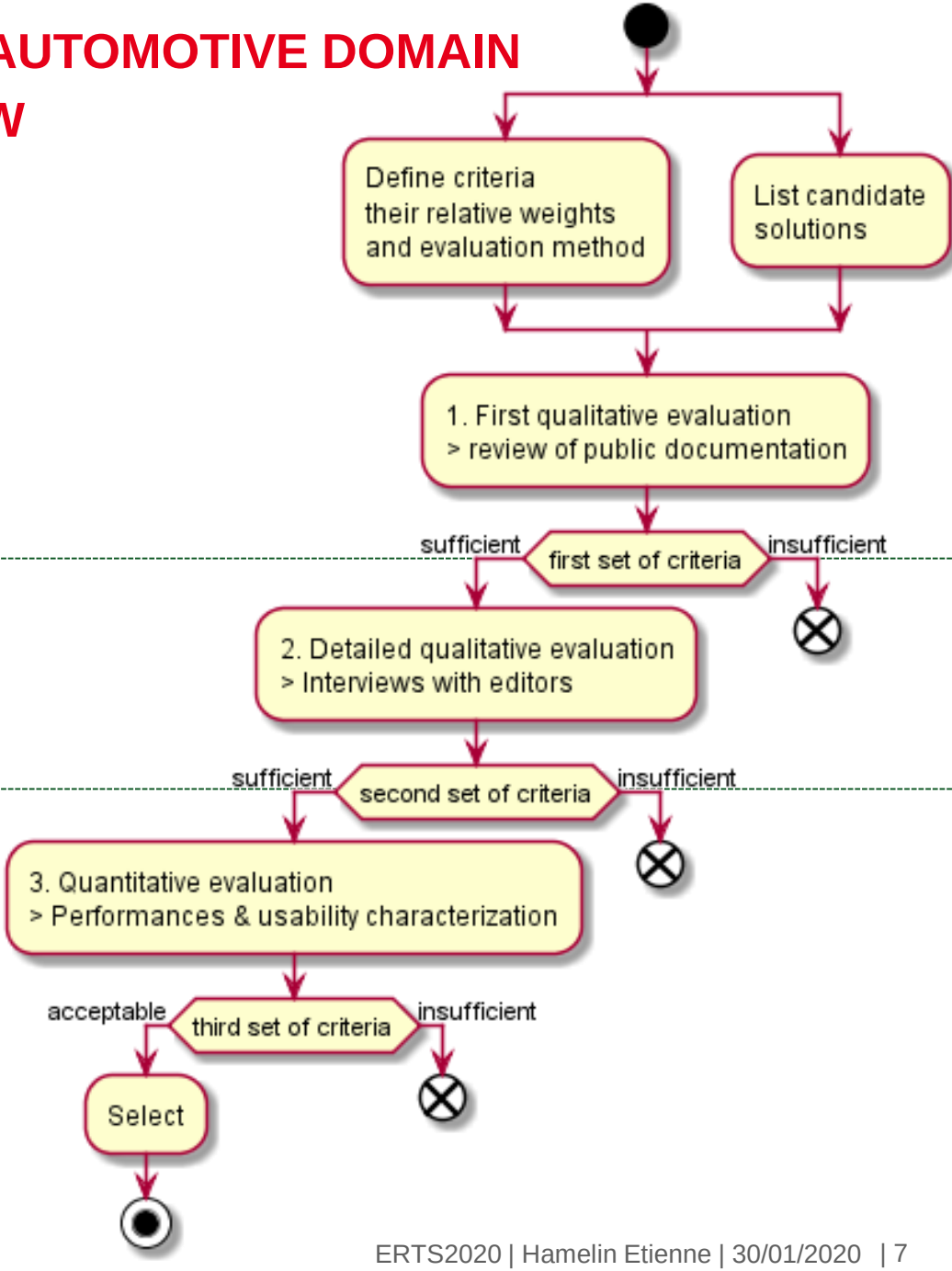
- **Empirical approach: multi-step filtering**
  - Assess criteria most easily evaluated, and most discriminative
  - Filter out solutions below threshold
  - Repeat until 1 solution

APPLICATION IN THE AUTOMOTIVE DOMAIN
SELECTION OVERVIEW

~ days of effort

~ weeks

~ months



APPLICATION IN THE AUTOMOTIVE DOMAIN
SELECTION OVERVIEW

23 solutions evaluated

CPU arch., periph. support
Real-time
Industrial maturity
Safety- / Security qualification

5 editors interviewed

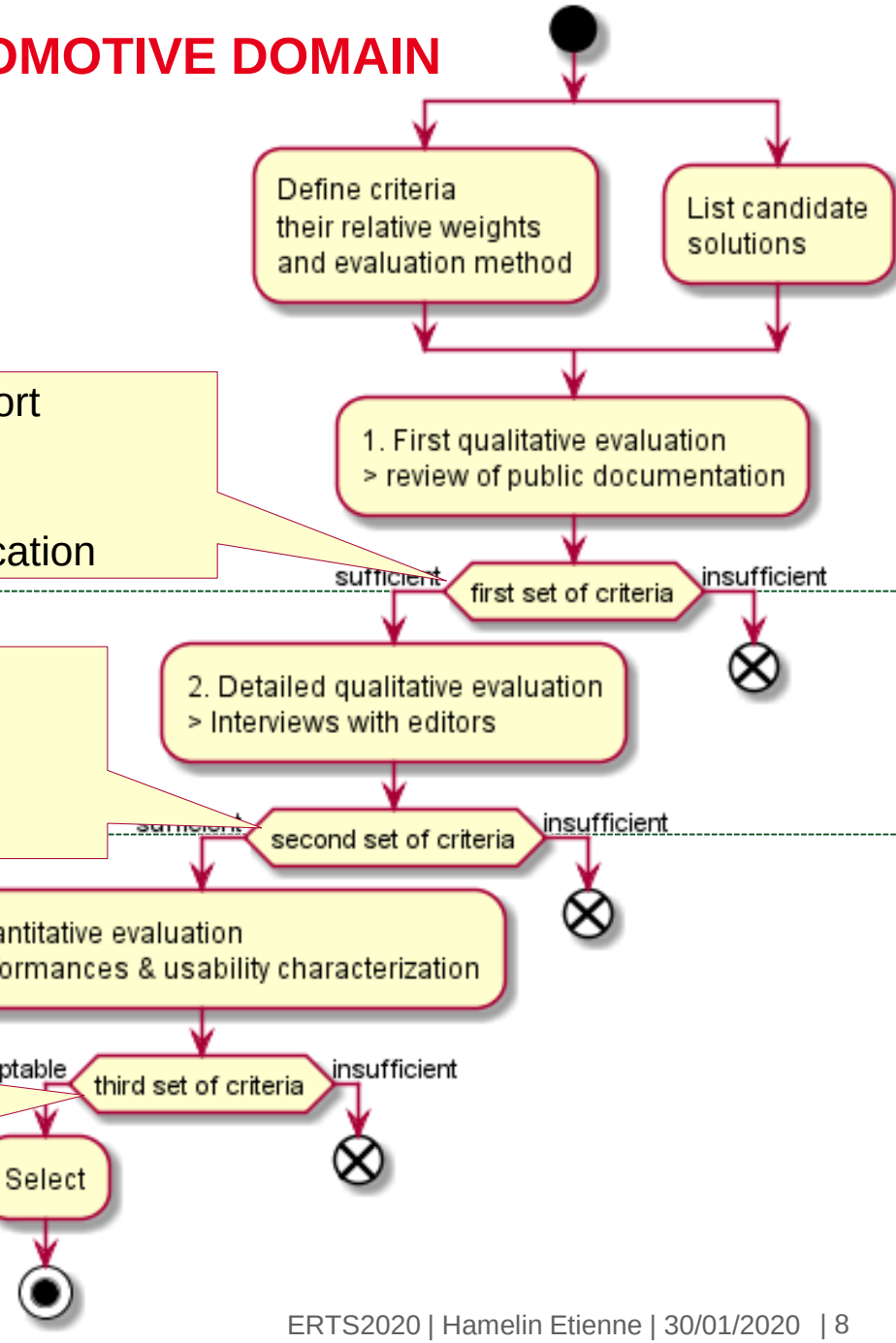
Red Bend Software, Mentor Graphics, Green Hills Software, QNX Hypervisor 2.0, SYSGO Embedding Innovations, Multivisor

Tool support
Partnership model
Applicable regulation

1 solution characterized

SYSGO Embedding Innovations

Performances overheads, predictability
Overall usability



APPLICATION IN THE AUTOMOTIVE DOMAIN
CHARACTERIZATION

- **Characterization**
  - Performance overheads
    - virtualized vs. bare-metal
  - Inter-VM interferences
    - disturbed vs. undisturbed

- **Quantitative metrics**
  - Boot time overhead
  - Memory overhead
  - Context switch overhead
  - Scheduling and interferences

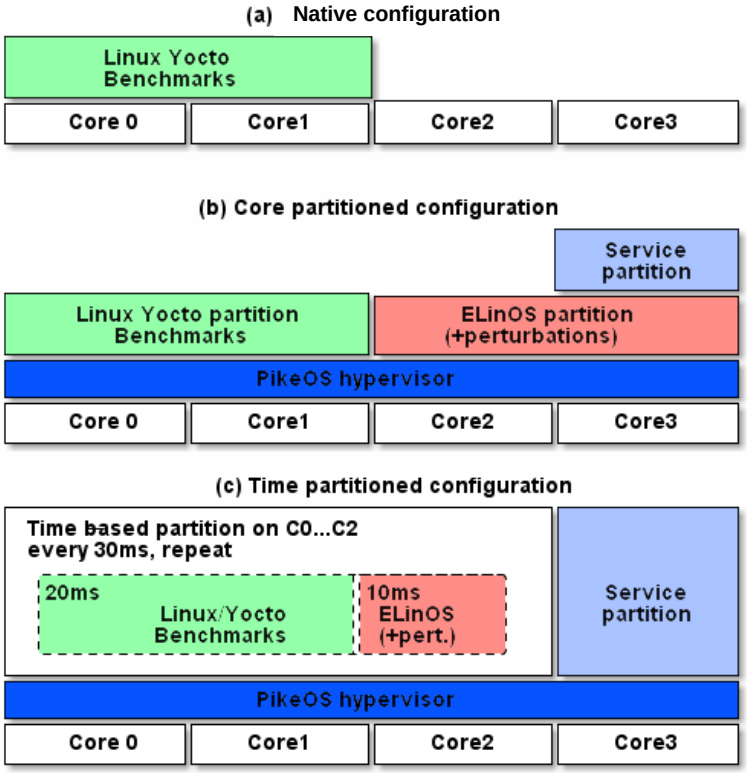
- **Environment**
  - Renesas RCar-H3: heterogeneous ARMv8A SoC
    - 4 × ARM Cortex-A57 (32kB L1I, 48kB L1D cache)
      - 2MB shared L2
    - 4 × Cortex-A53 (32kB L1I, 32kB L1D cache)
      - 512kB shared L2
    - 2 × Cortex-R7 Dual lockstep (32kB L1I, 32kB L1D cache)

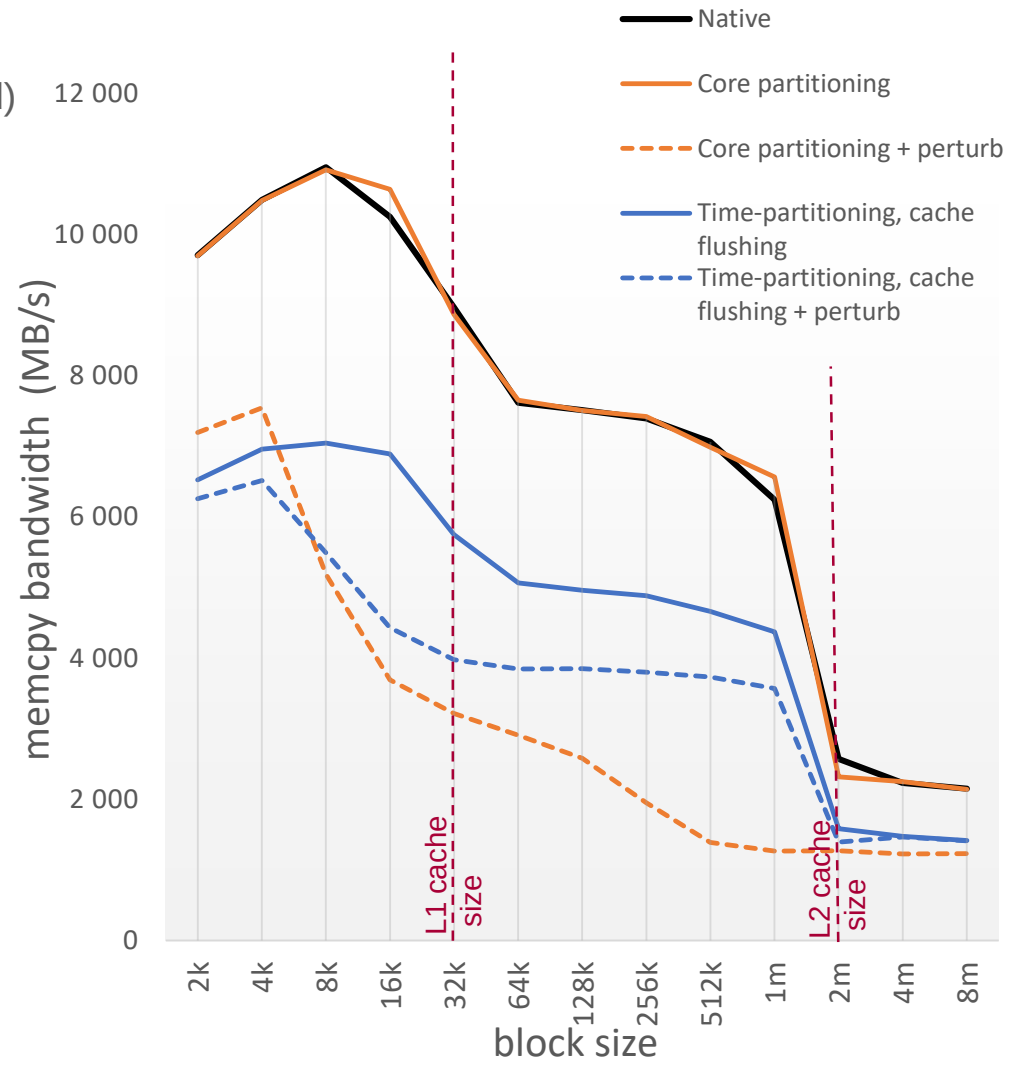
CHARACTERIZATION: A FEW FIGURES

- **Boot time**
  - From last U-Boot instruction to first VM instruction
  - Depends on VM size
  - Measured ~16ms + 31ms per GB
- **Context switch time**
  - Ping-pong message between 2 VMs
  - Measured ~ 8µs to 17µs (warm/cold caches)
- **Memory overhead**
  - Hypervisor footprint + VMM memory per VM
  - Measured from 8MB (hello world app) to 28MB (full Linux VM)
- **Computational overhead**
  - MiBench basicmath compute-bound tasks
  - Measured ~4%

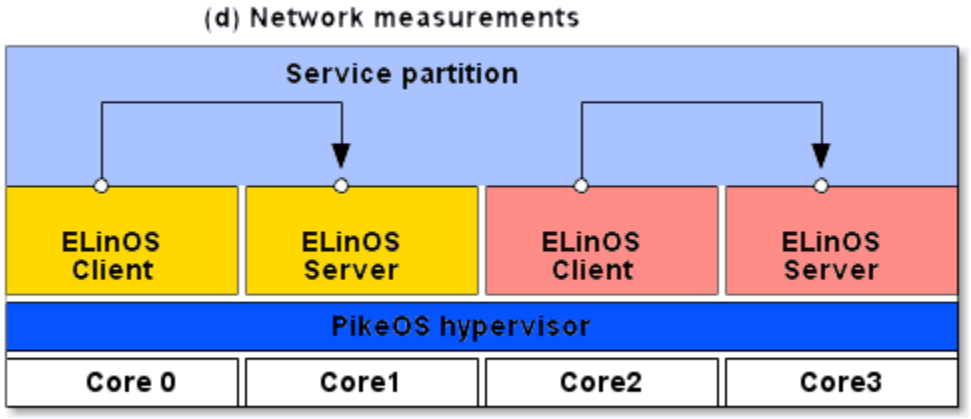
CHARACTERIZATION
MEMORY BANDWIDTH & INTERFERENCES

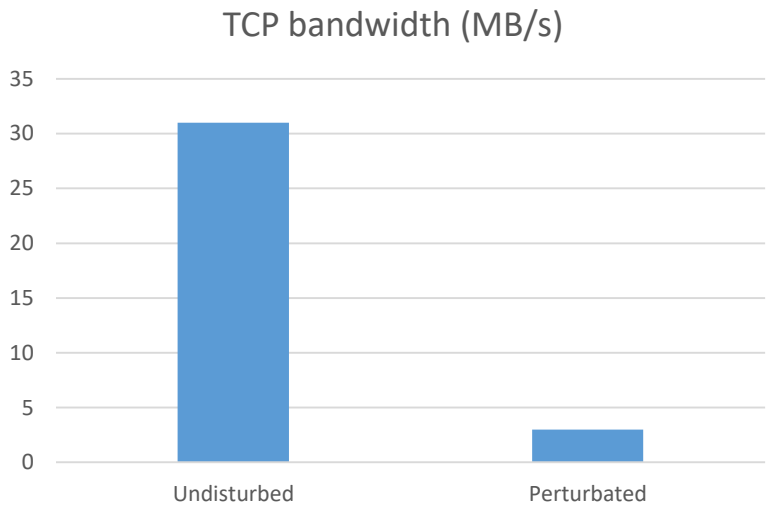
VM1: Linux + HBench-OS mem benchmark
VM2: optional perturbation (Hbench-OS based)





CHARACTERIZATION
SHARED SERVICES, E.G. NETWORK

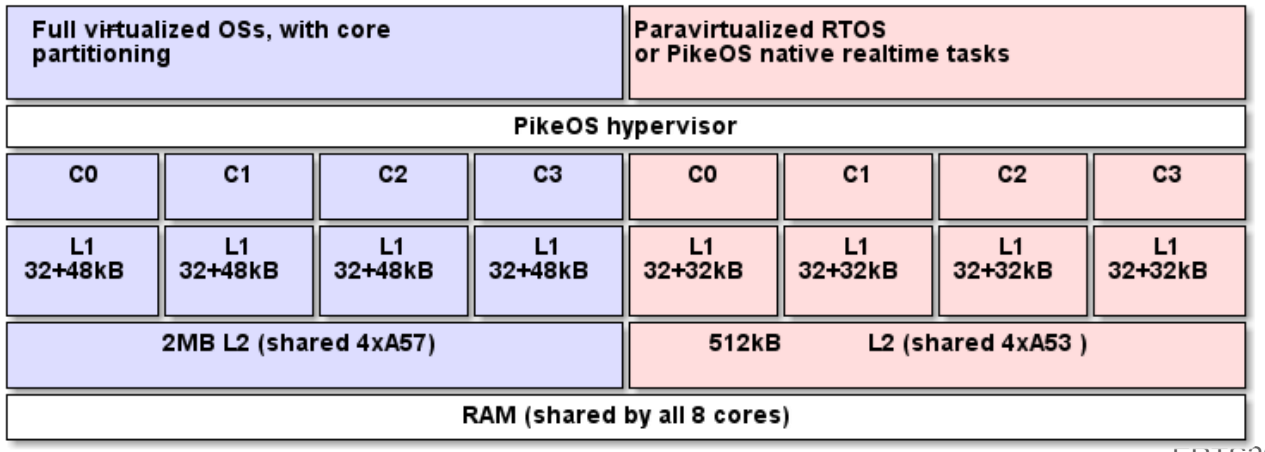




- **Inter-VM shared services: virtual network**
  - Hbench-OS TCP bandwidth benchmark, server/client configuration (bench) + server/client (disturbance)
  - From ~31MB/s (undisturbed) down to ~3MB/s: 90% bandwidth loss
- **2 simultaneous sources of interference**
  - Shared software service used by both pairs
  - Service handler (partition) can preempt user application

FEEDBACK FOR THE AUTOMOTIVE DOMAIN

- **Feedback**
  - Perf. overheads limited (boot time, CPU time, context switch, mem)
  - Impact of inter-VM interferences on predictability
    - shared HW (e.g. caches, TLB)
    - SW services (e.g. shared Eth.)
    - even in time-partitioning with L1 cache & TLB flush
- **Usage recommendation**
  - Mitigate interference through hardware
    - Reduce resource sharing between real-time & best-effort worlds
    - Leverage L2 cache separation between clusters
  - Mitigate shared services-induced interference
    - Software monitoring / rate-control usage of shared services



OVERALL CONCLUSION

- **Selecting a software platform is a strategic choice…**
  - High technical stakes
    - compatibility, performance, features
    - many issues can often be dealt with usage restrictions or additional developments
  - Non-technical stakes sometimes even higher
    - partnership & licensing,
    - business-model,
    - regulation