Virtualization in Automotive Embedded Systems : an Outlook

Nicolas Navet, RTaW

Bertrand Delord, PSA Peugeot Citroën

Markus Baumeister, Freescale

Talk at RTS Embedded Systems 2010
Paris, 31/03/2010

Outline

1. Automotive E/E Systems: mastering complexity
2. Ecosystems of virtualization technologies
3. Automotive use-cases of virtualization
4. Limits of virtualization

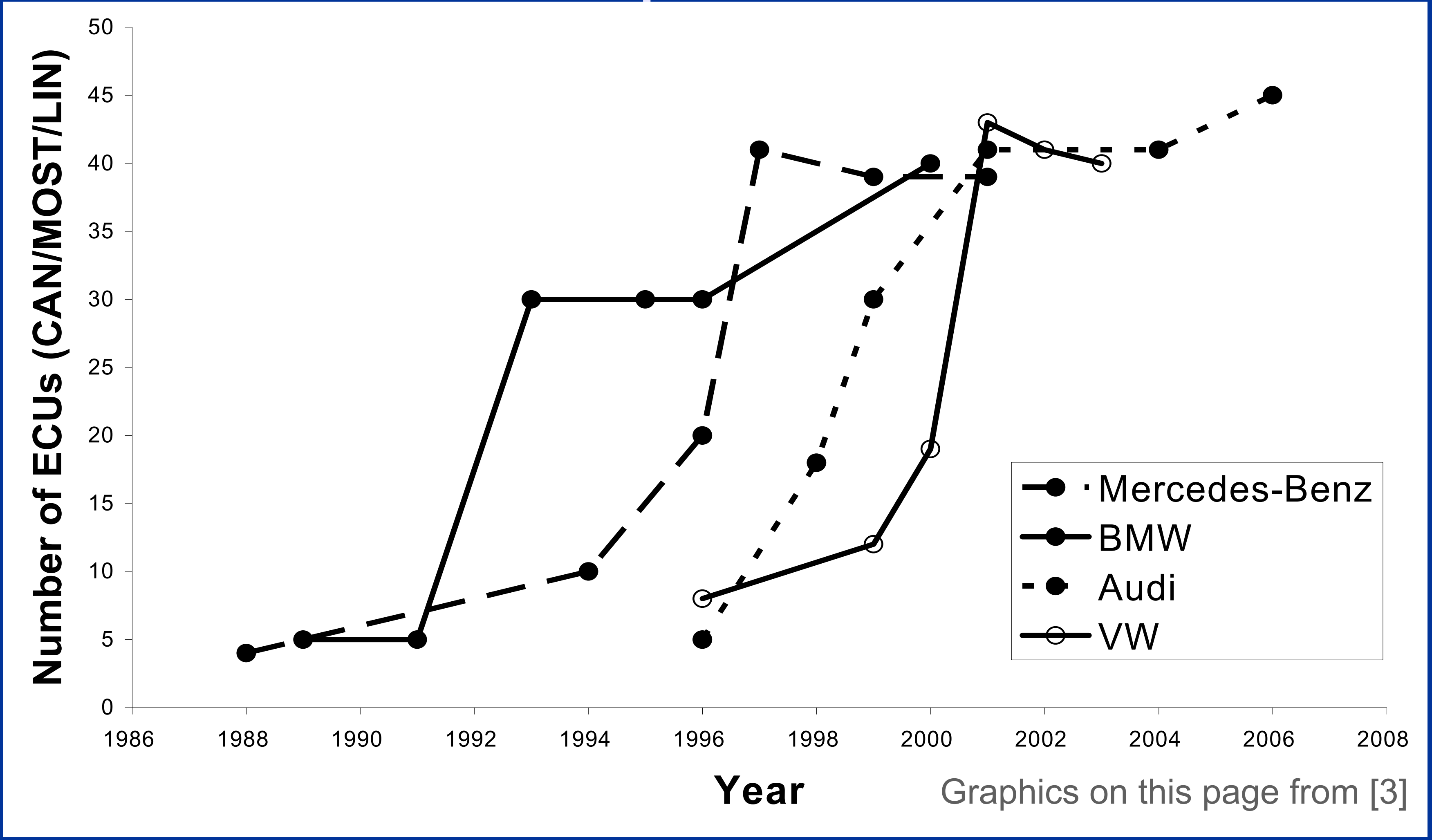
Mastering complexity of automotive Electrical and Electronics (E/E) Systems

Electronics is the driving force of innovation

- 90% of new functions use software
- Electronics: 40% of total costs
- Huge complexity: 80 ECUs, 2500 signals, 6 networks, multi-layered run-time environment (AUTOSAR), multi-source software, multi-core CPUs, etc

Strong costs, safety, reliability, time-to-market, reusability, legal constraints !

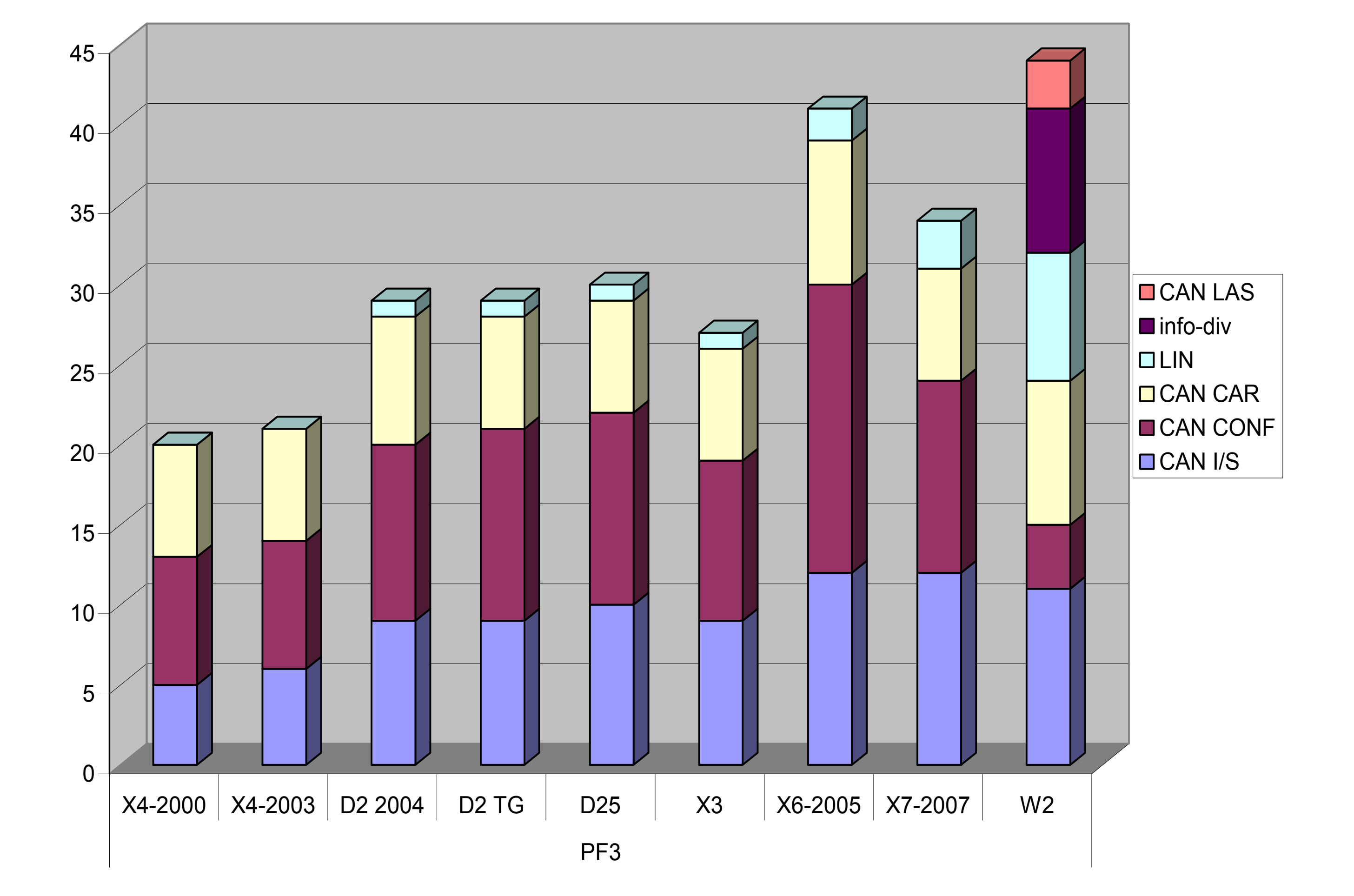
Proliferation of ECUs raises problems!



Graphics on this page from [3]

Lexus LS430 has more than 100 ECUs [9]

The case of a "generalist" car manufacturer - PSA

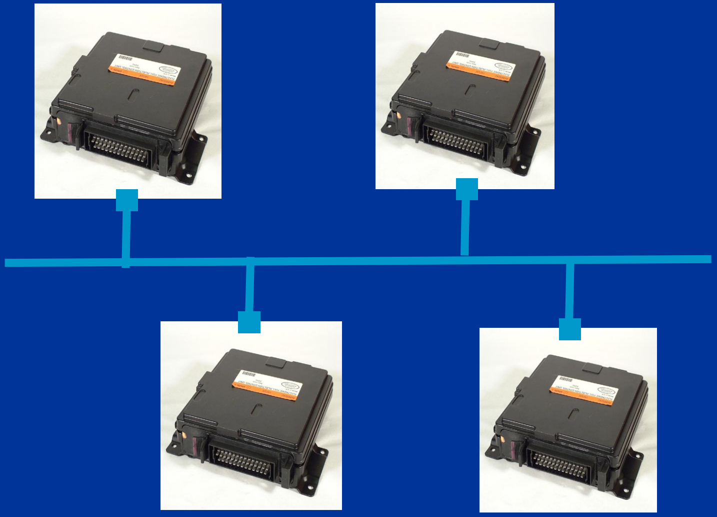


The number of ECUs has more than doubled in 10 years

Possible upcoming architectures in two car generations

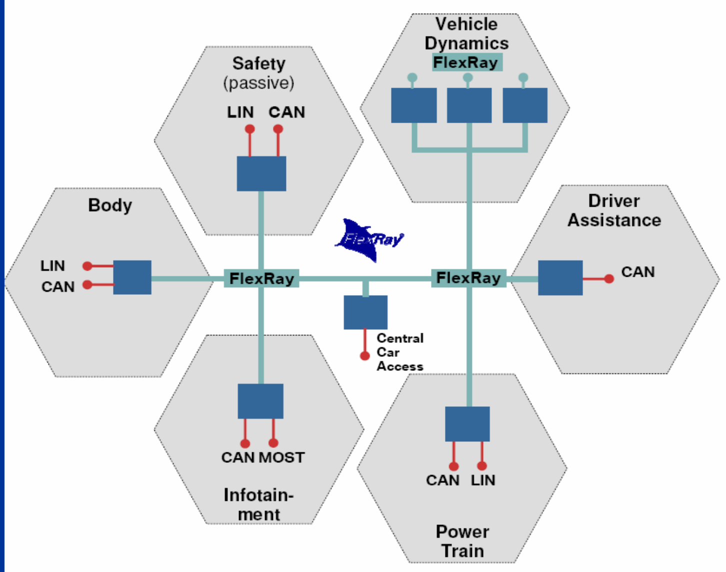
Fewer ECUs but more powerful

- Multi-core μ-controller
- Multi-source software
  - Autosar OS strong protection mechanisms
  - Virtualization ?
- ISO2626-2 dependability standard



Backbone :

- CAN 500Kbit/s with offsets
- FlexRay™ : 10 Mbit/s
- Ethernet ?



FlexRay™ as backbone at BWM in a few years [8]

How centralized is unsure because of carry-over ..

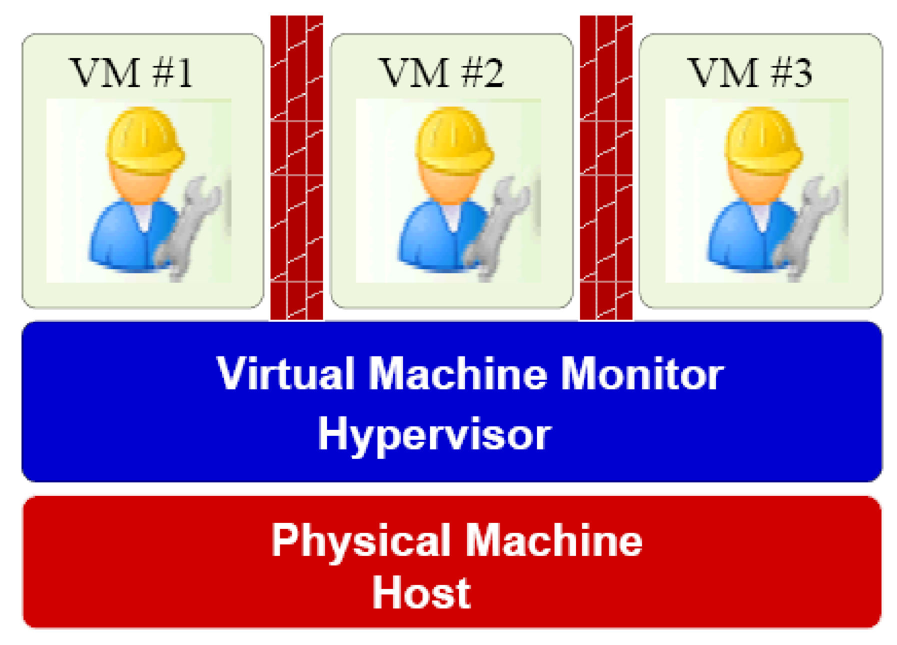
Ecosystem of virtualization technologies

Virtualization basics

Executing software on virtual machines decoupled from the real HW

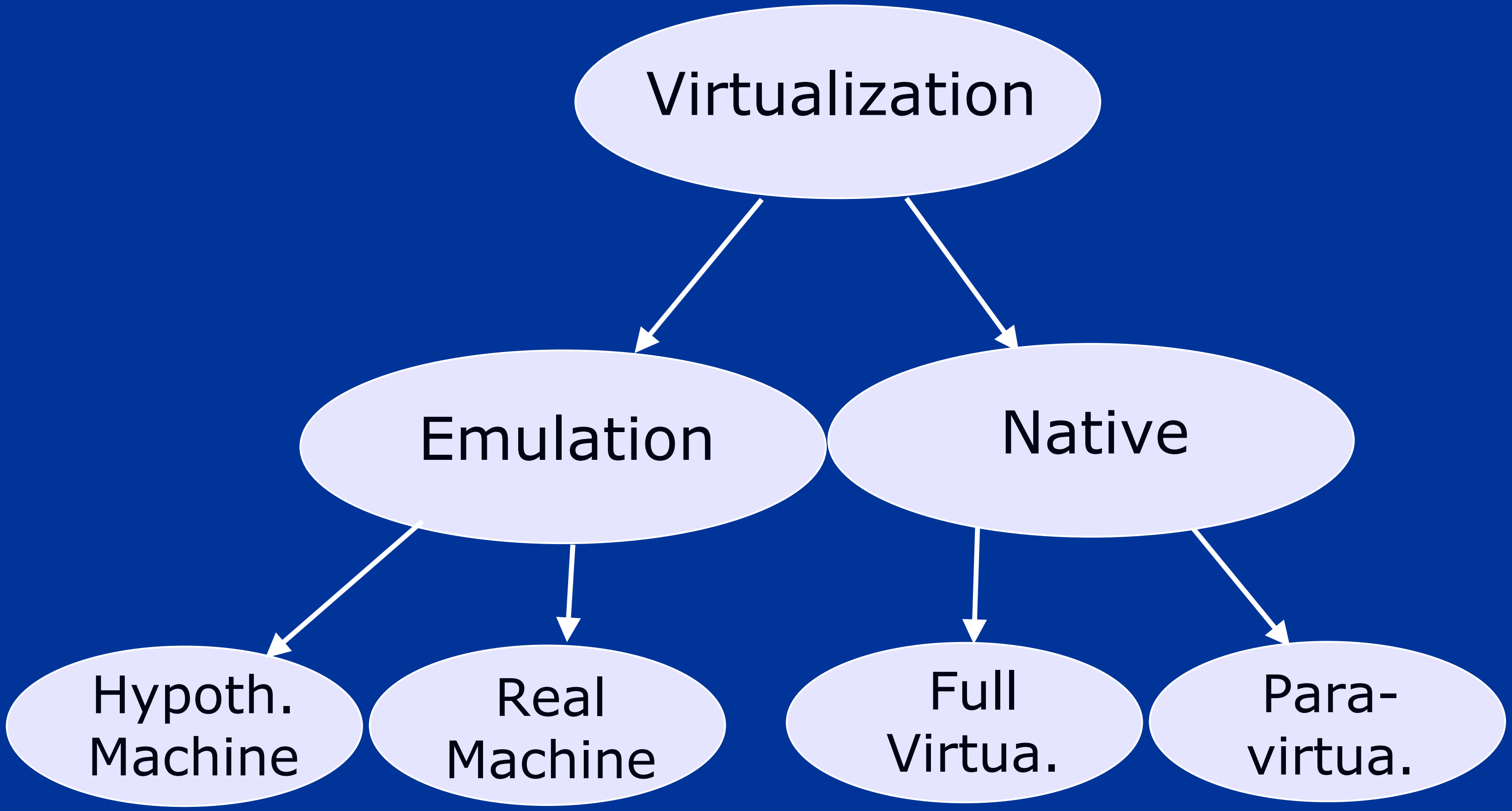
- Virtual Machine: software that executes software like a physical machine
- (System) VM contains an OS
- HW resources can be shared between VMs : role of hypervisor

Strong isolation between VMs : security and fault-confinement are the primary motivations



Picture from [2]

Classification of virtualization schemes [3]



eg. JVM

eg. Bochs

eg. Z/VM

eg. Xen, Sysgo, Wind River

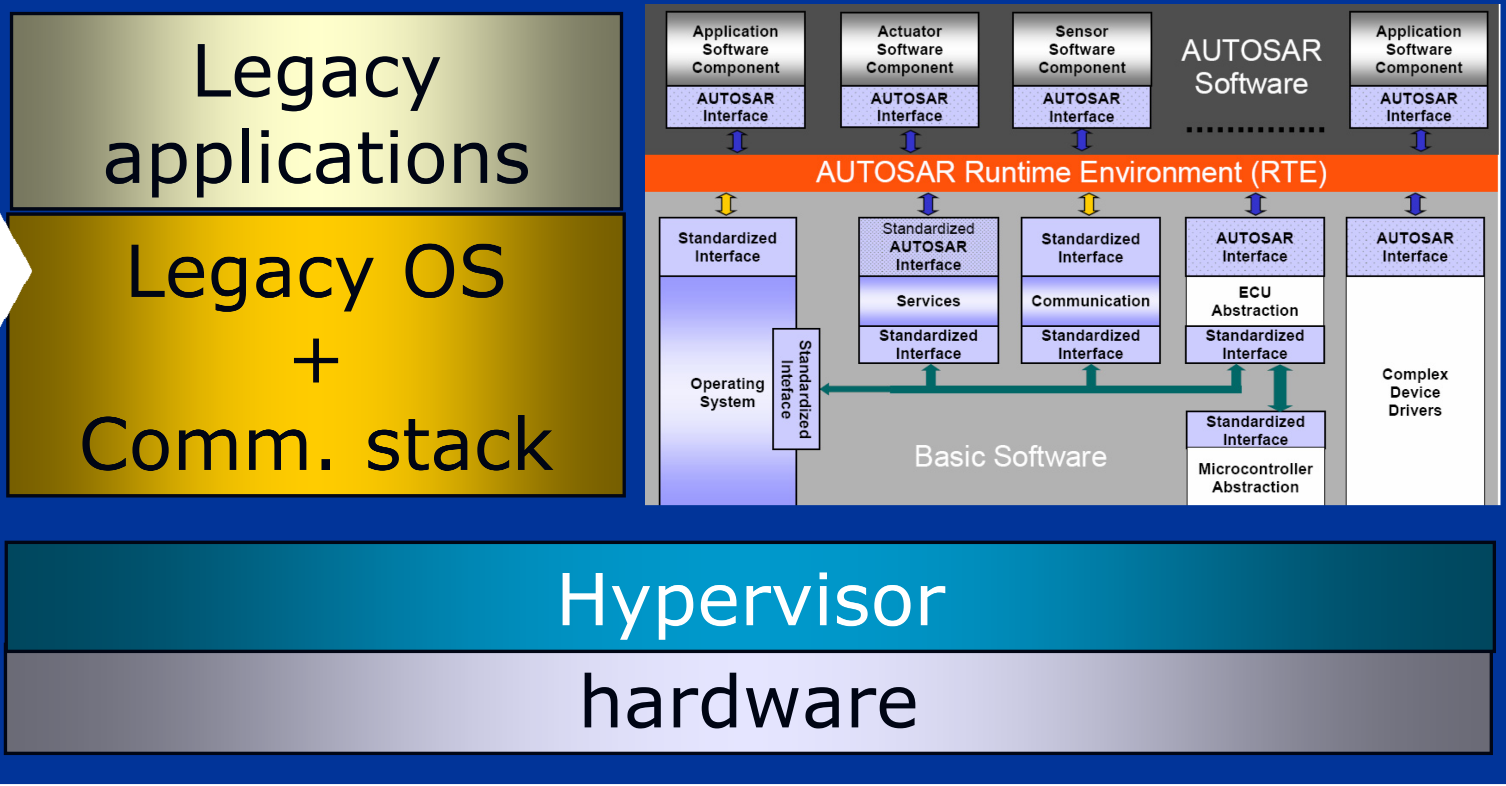
Use-cases of virtualization

Heterogeneous operating system environments (1/2)

- Re-use of a complete legacy ECU : eg. parking assistance

Benefits

- Time-to-market,
- Cost reduction
- Validation done
- Way to deal with discontinued hardware

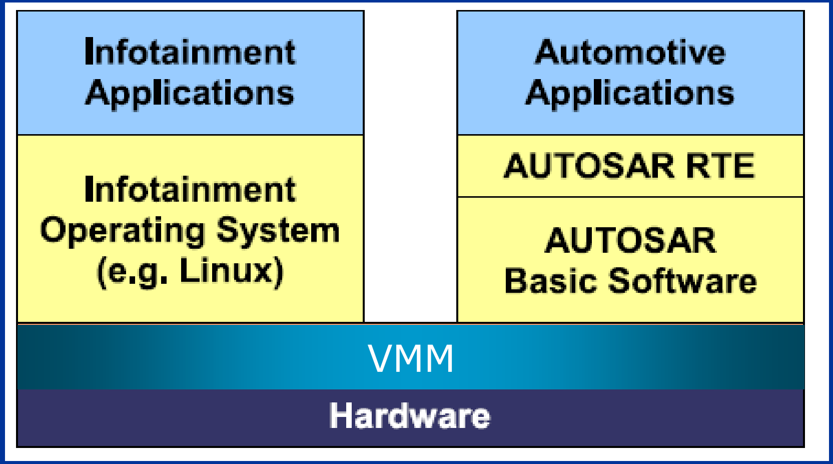


Heterogeneous operating system environments (2/2)

- Using the best execution platform : eg. Body gateway with both an Autosar and an infotainment VM (eg., linux, android)

Benefits

- Performances
- Availability of manpower / applications
- Time-to-market
- **Security despite open systems**
- Segregation in "vehicle domains"
- Etc


Infotainment Applications
Infotainment Operating System (e.g. Linux)
Automotive Applications
AUTOSAR RTE
AUTOSAR Basic Software
VMM
Hardware

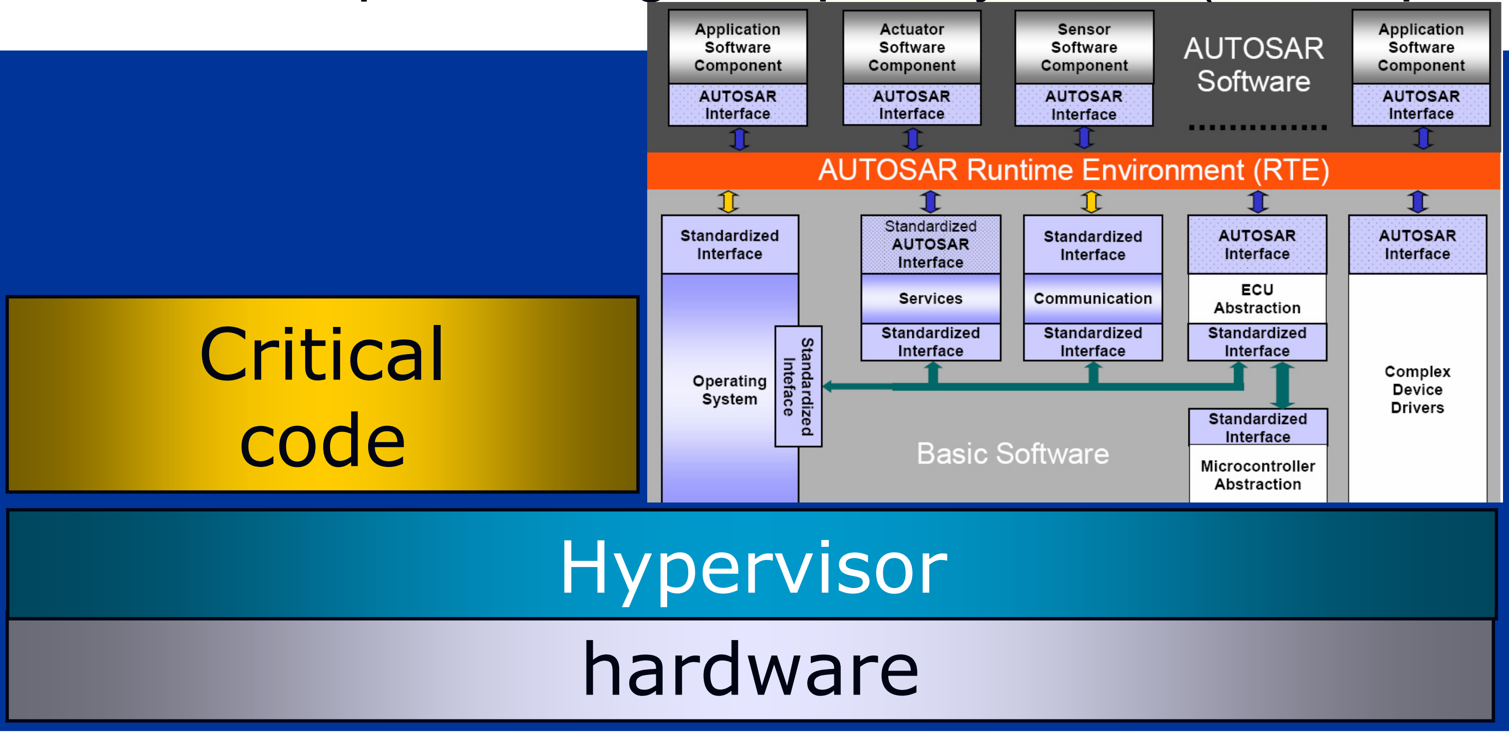
Picture from [2]

The most obvious and likely use-case in a first step

Virtualization for security-critical sub-systems

Benefits:

- Critical code can run on bare hardware
- Sufficiently small for formal methods
- “Brick-wall” partitioning for open systems (OTA update)



Critical code

Hypervisor

hardware

Virtualization for safety-critical sub-systems

Short term benefits:

- **Memory, CPU, IO protection mechanisms**
- **Redundant execution with diversity** reduces common faults, possible to go one step farther with OS and com. stack diversity
- **Monitoring / watchdog on the same multi-core chip** (ideally with some HW diversity at the core level)

Medium term goal:

- **Virtual lockstep execution without dedicated HW**

Not the same scope of protection as Autosar OS

Autosar OS : OS application, OS task, ISR

Virtualization : VM (usually with an OS)

AUTOSAR OS protection mechanism - a recap (see [7])

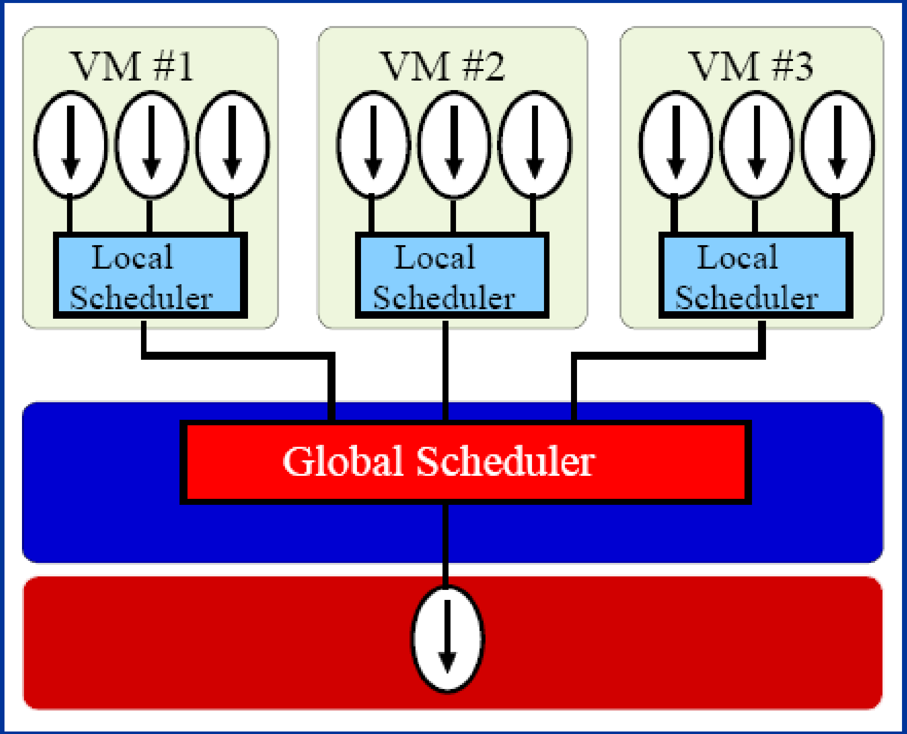
- Issues : resource confiscation (CPU, memory, drivers), non authorized access / calls, fault-propagation
- 5 types of mechanisms
  - Memory protection
  - Temporal protection
  - OS service protection
  - HW resource protection
  - trusted / non-trusted code
- 4 scalability classes

As of Autosar R4, there are multi-core extensions enabling CPU core partitioning

Limits of virtualization

Real-time performances

Virtualization implies a hierarchical two-level scheduling that is inherently less predictable and more complex to handle



Picture from [2]

Actually, three-level scheduling since runnables are scheduled within OS tasks!

✓ Static core allocation (to VMs) is probably the way to go ..

Technical issues

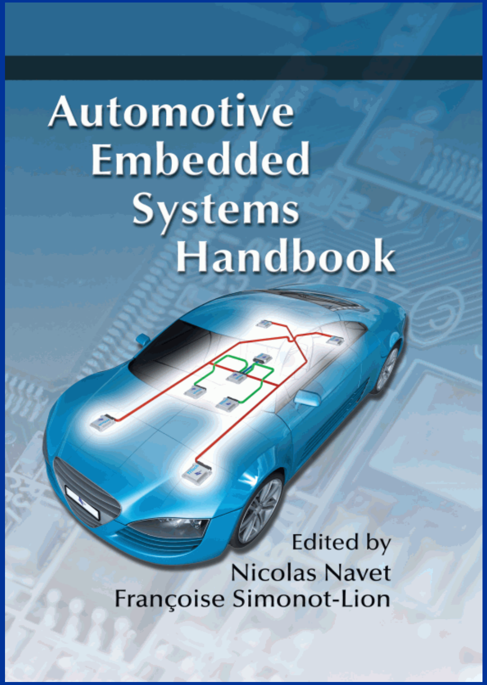
- Memory:
  - VMM footprint: < 64KB
  - Possibly several OSs !
- CPU:
  - Limited hardware support in embedded CPU [6]
  - Preemption, L2 cache flush, locked cache
- Resource sharing is tricky: ISR, IOs, com. controllers
  - Real-time performances (eg. LIN)
  - peripheral virtualization is complex (eg. CAN)
- VMM must be kept small to be secure (more than guest OSs) and ideally bug free … otherwise responsibility sharing is impossible

Conclusion

- Virtualization is a mature technology, industrial risk is limited
- Automotive can benefit from both aerospace / military and consumer electronic experiences: Products, certification, deployment tools, etc
- The overlap between virtualization and Autosar OS seems small
- There are meaningful use-cases but real-time behavior of the virtualized systems should be (formally) verified.

References

References



[1] N. Navet, F. Simonot-Lion, editors, The Automotive Embedded Systems Handbook, Industrial Information Technology series, CRC Press / Taylor and Francis, ISBN 978-0849380266, December 2008.

[2] R. Kaiser, D. Zöbel, Quantitative Analysis and Systematic Parametrization of a Two-Level Real-Time Scheduler, paper and slides at IEEE ETFA'2009.

[3] T. Nolte, Hierarchical Scheduling of Complex Embedded Real-Time Systems, slides presented at the Summer School on Real-Time Systems (ETR'09), Paris, 2009.

[4] G. Heiser, The role of virtualization in embedded systems, Proceedings of the 1st workshop on Isolation and integration in embedded systems, 2008.

[5] D. Baldin, T. Kerstan, Proteus, a Hybrid Virtualization Platform for Embedded Systems, IFIP Advances in Information and Communication Technology, 978-3-642-04283-6, 2009.

[6] F. Behmann, Virtualization for embedded Power Architecture CPUs, Electronic Products, September 2009.

[7] N. Navet, A. Monot, B. Bavoux, F. Simonot-Lion, Multi-source and multicore automotive ECUs - OS protection mechanisms and scheduling, to appear in IEEE ISIE, 2010.

[8] A. Schedl, Goals and Architecture of FlexRay at BMW, slides presented at the Vector FlexRay Symposium, March 2007.

[9] R. Schreffler, Japanese OEMs, Suppliers, Strive to Curb ECU Proliferation, Wardsauto.com, March 6, 2006.

Questions / feedback ?

Please get in touch at :
nicolas.navet@realtimeatwork.com
bertrand.delord@mpsa.com
B17517@freescale.com