KIA4SM - Kooperative Integrationsarchitektur für zukünftige Smart Mobility Lösungen

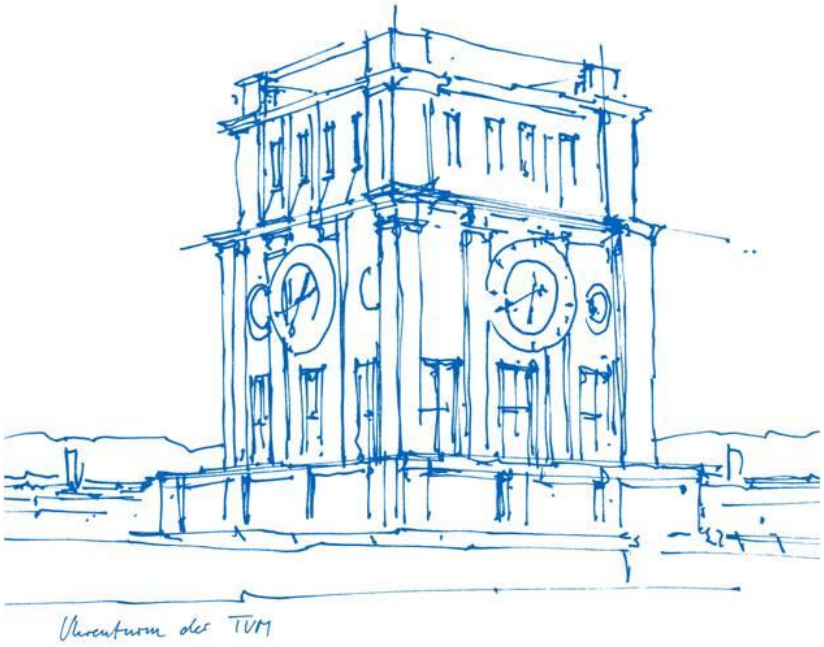
Daniel Krefft, Sebastian Eckl, Uwe Baumgarten

Technische Universität München

Fakultät für Informatik

Lehrstuhl/Fachgebiet für Betriebssysteme

Günzburg, 02./03. März 2017



Agenda

- About
- Vision
- Project: KIA4SM
- Dynamic Reconfiguration
  - Migration
  - Partitioning
- Test Setup
- Next Steps

Agenda

- **About**
- Vision
- Project: KIA4SM
- Dynamic Reconfiguration
  - Migration
  - Partitioning
- Test Setup
- Next Steps

About

- Cyber-physical systems, their components, mechanisms, and algorithms
- Automotive systems both inside future and possibly inter-connected vehicles
- Microkernel for dynamic load on multi-core platforms

About

Prof. Dr. Uwe Baumgarten
Head

Sebastian Eckl, M.Sc.
Research Associate

Daniel Krefft, M.Sc.
Research Associate

Agenda

- About
- **Vision**
- Project: KIA4SM
- Dynamic Reconfiguration
  - Migration
  - Partitioning
- Test Setup
- Next Steps

Vision

- Research topic: Distributed, Embedded, Real-Time Systems
- Example: Automotive Systems
- (Future) Environment/Settings: C-ITS & Autonomous Driving
  - Increasing amount of data & SW-based functions
  - More powerful hardware
- Approach: HW-Consolidation & SW-Virtualization
- Challenge: Mixed-Criticality Systems

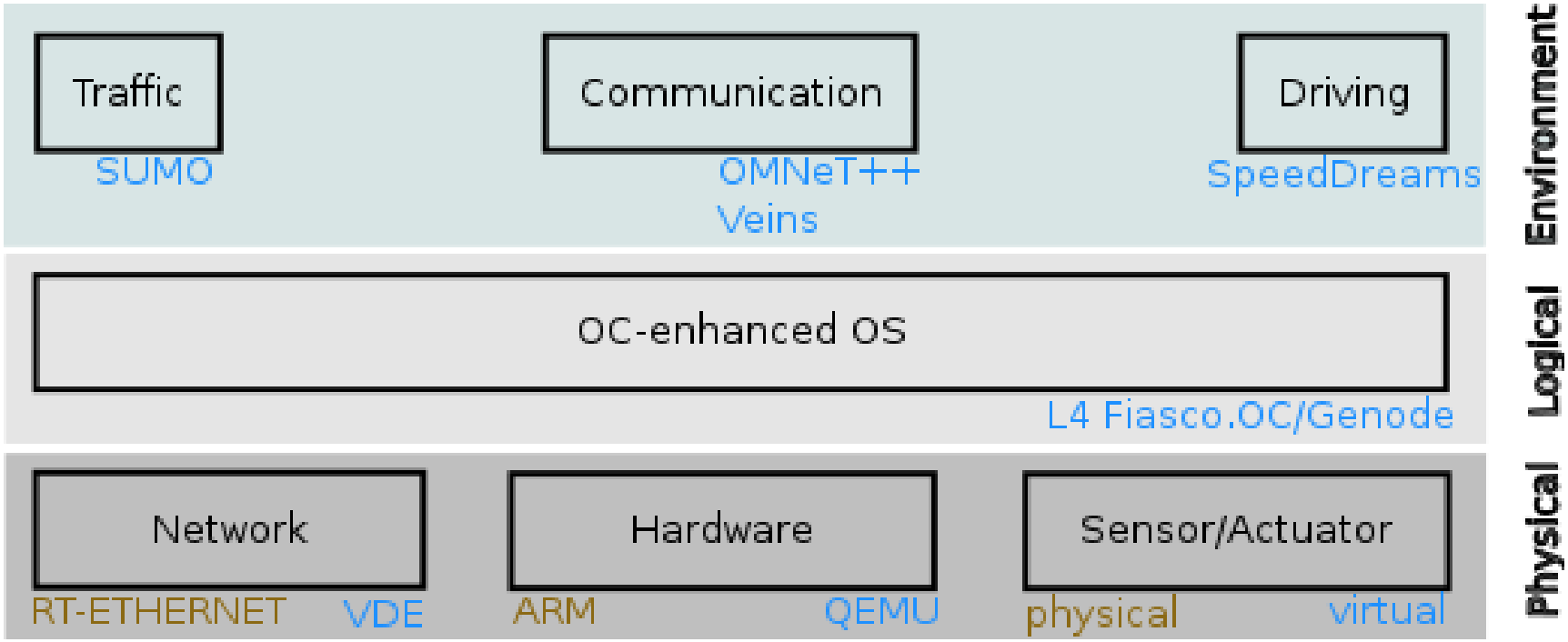
Vision

- Solution
  - State of Art: Static configuration
  - Goal: combination of partitioned dynamic and static parts
- Tradeoff: Flexibility vs. Reliability/Safety
- Idea: (Online) Dynamic Reconfiguration [Self-X]
  - Network (e.g. routing)
  - Hardware (e.g. FPGA)
  - Software (e.g. migration, replication)
- Choice: migration of SW-components at runtime

Agenda

- About
- Vision
- **Project: KIA4SM**
- Dynamic Reconfiguration
  - Migration
  - Partitioning
- Test Setup
- Next Steps

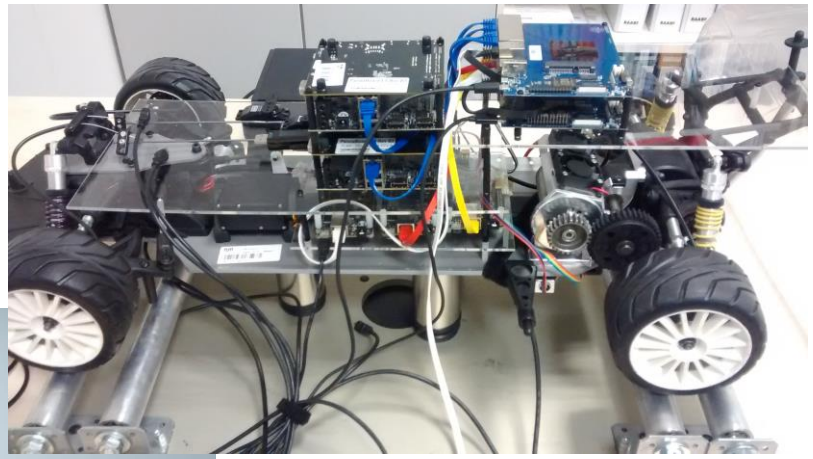
Project: KIA4SM

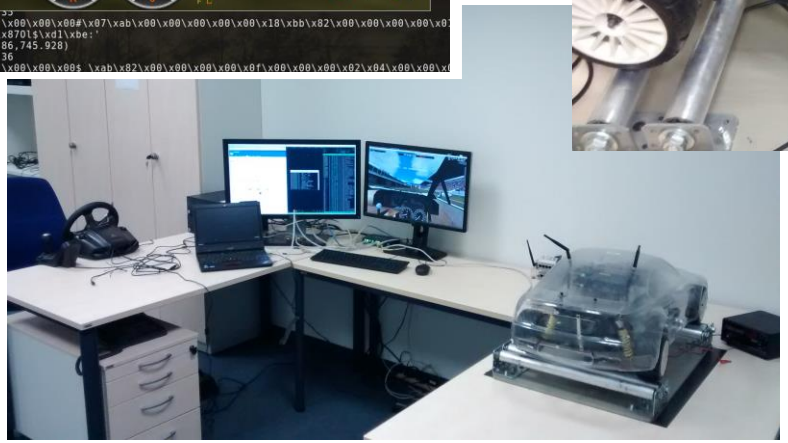


Three-layered architecture, driven by the idea of likewise combining virtuality and reality by allowing the execution of the OC-enhanced operating system on physical as well as virtual components.

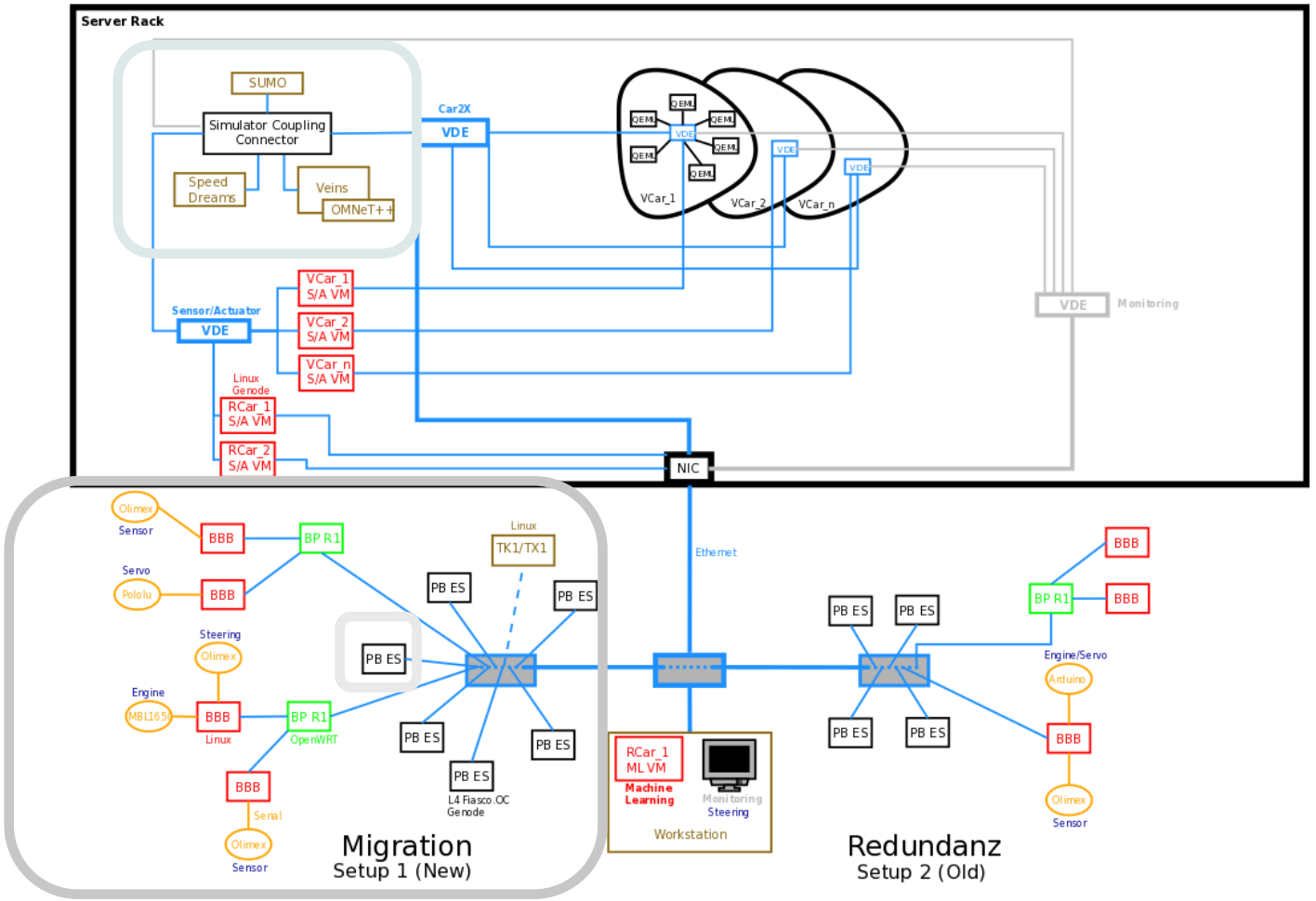
Setup


F/R Brake Rep.
SUMO: 1/1
Packet data: b'\x00\x00\x00$ \xab\x82\x00\x00\x00\x00\x0f\x00\x00\x00\x02\x04\x00\x00\x
Packet length: 36
tdpos is (1140.86,745.928)
Packet length: 36

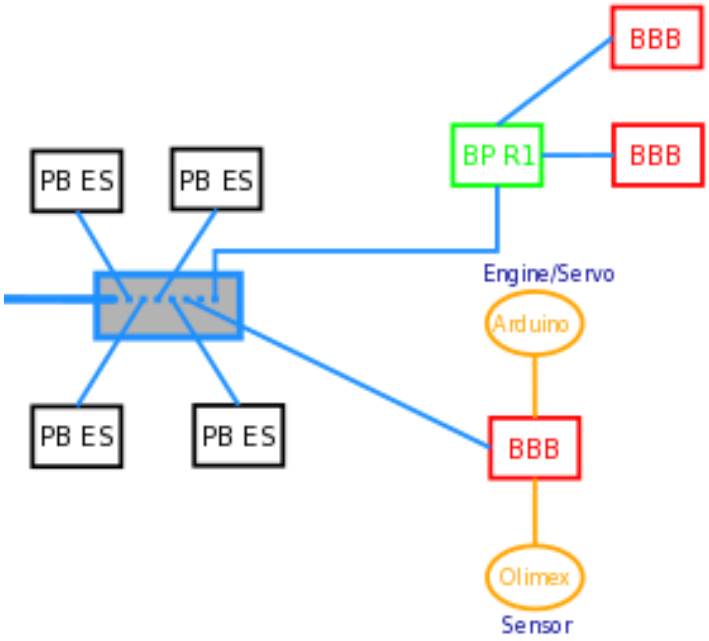




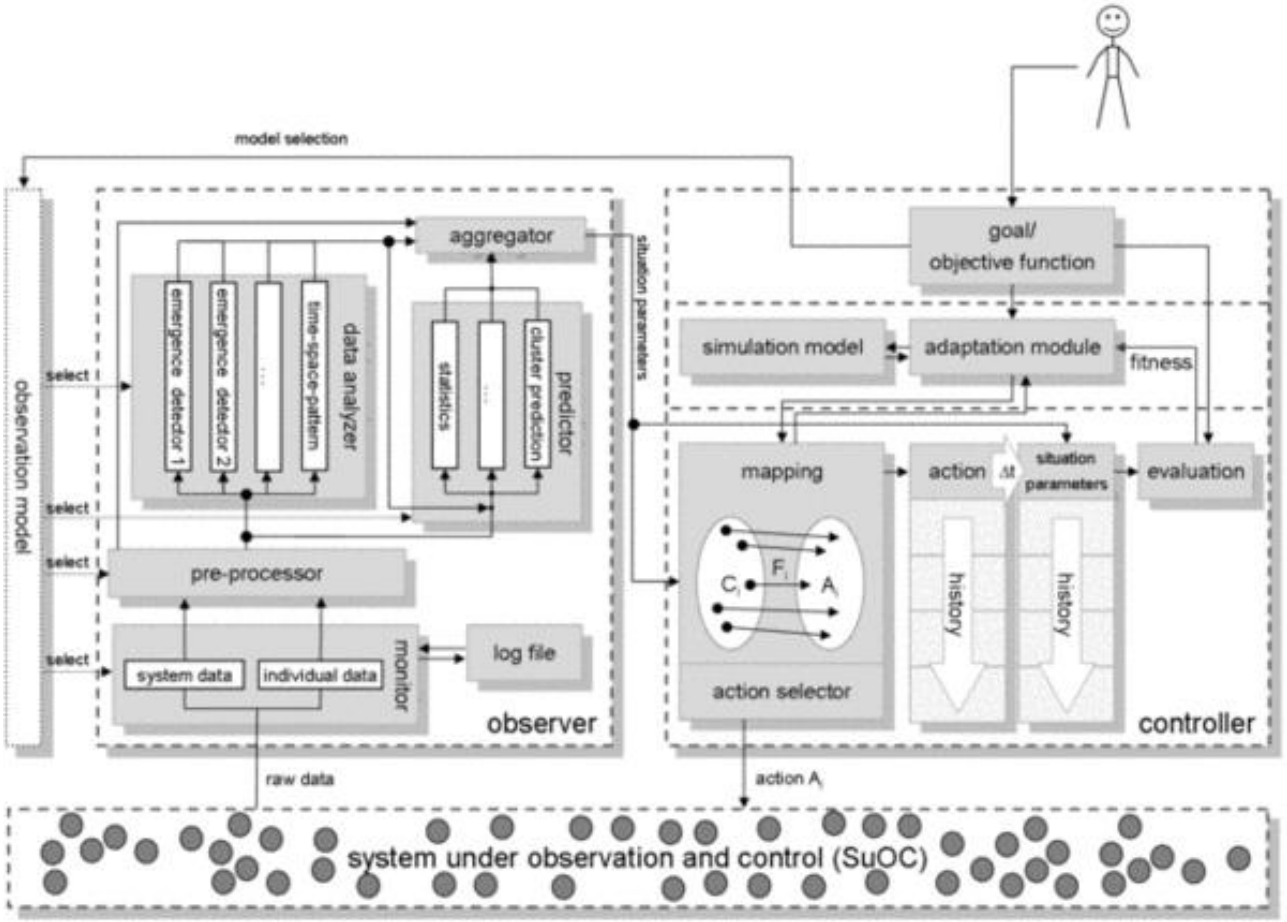
Hybrid Simulator Setup



ECU in detail

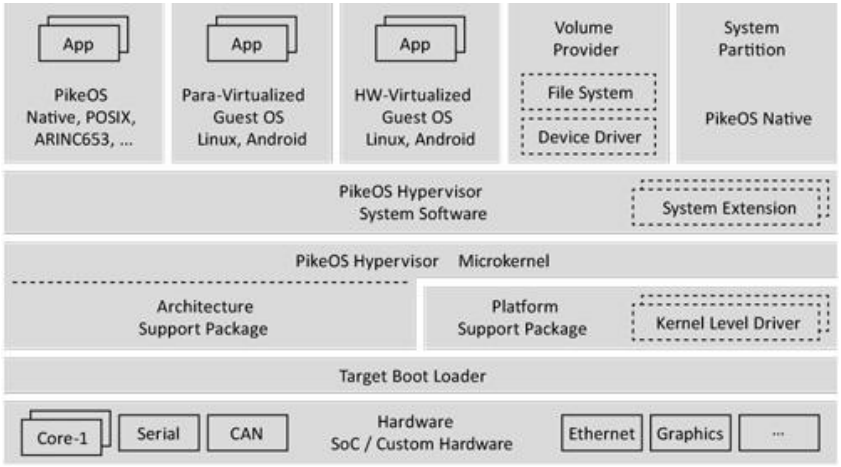


Background: Organic Computing (Flexibility)

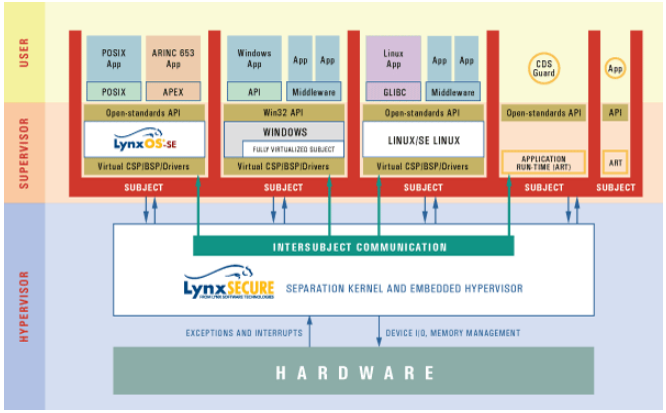


https://www.sra.uni-hannover.de/fileadmin/sra/pdf/Poster2005_QE_Mnif_v1.pdf

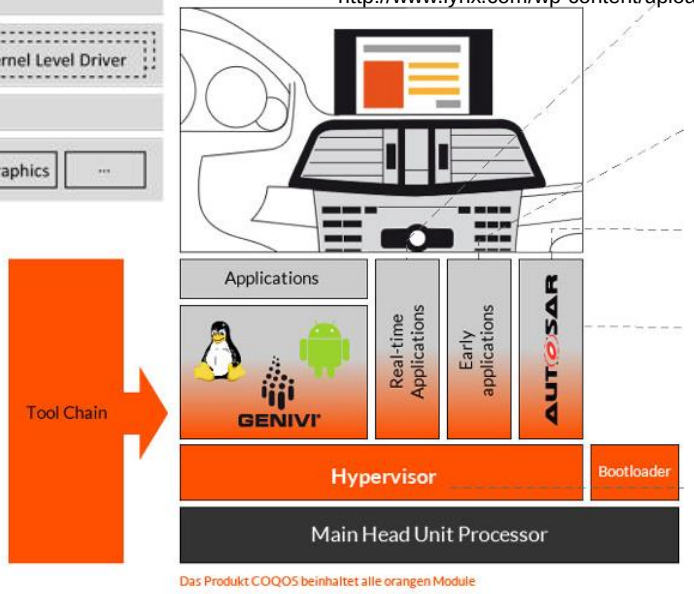
Background: Static deployment (Safety)



http://www.opensynergy.com/produkte/pikeos/

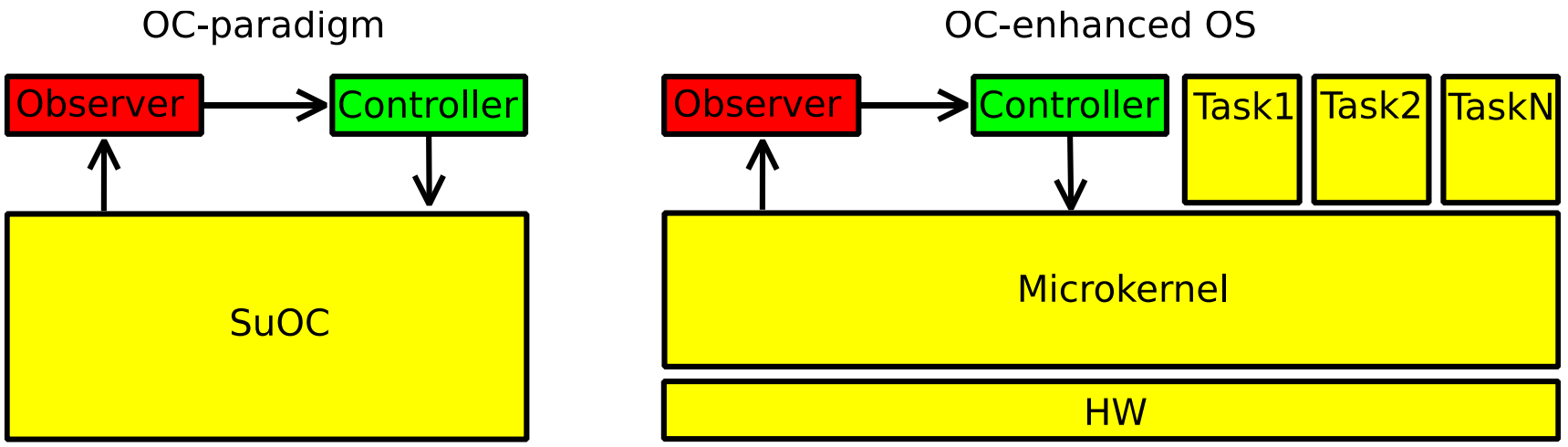


http://www.lynx.com/wp-content/uploads/2008/02/lynxsecure-2009-600px.png



http:///www.opensynergy.com/produkte/coqos/products/details

Application in KIA4SM

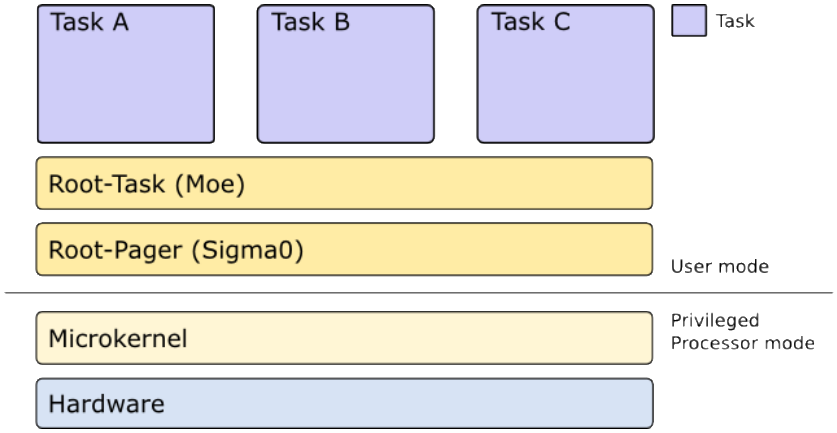


1. Organic Computing: Observer/Controller pattern
2. Small Trusted Computing Base (Microkernel)
3. Data Centric Communication: Publish/Subscribe mechanism

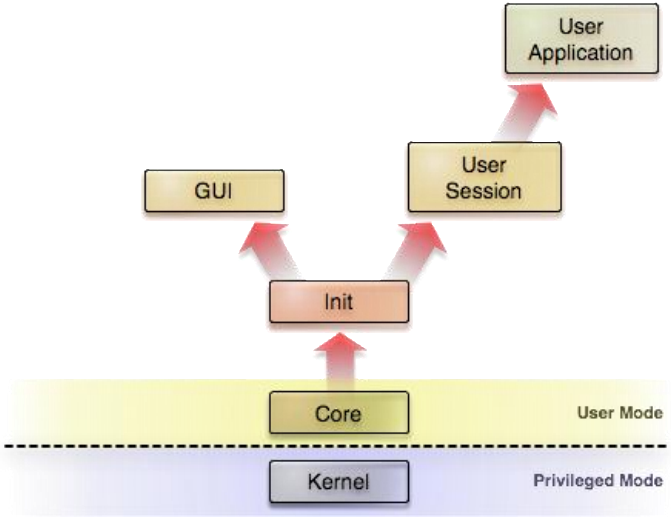
Research Areas

- Communication
- Migration
  - Checkpoint/Restore
  - Flexible Thread Management (variable task-set)
    - Admission
    - Synchronisation
    - Scheduling
  - Analysis/Planning (local/global)

Background: L4 Fiasco.OC & Genode

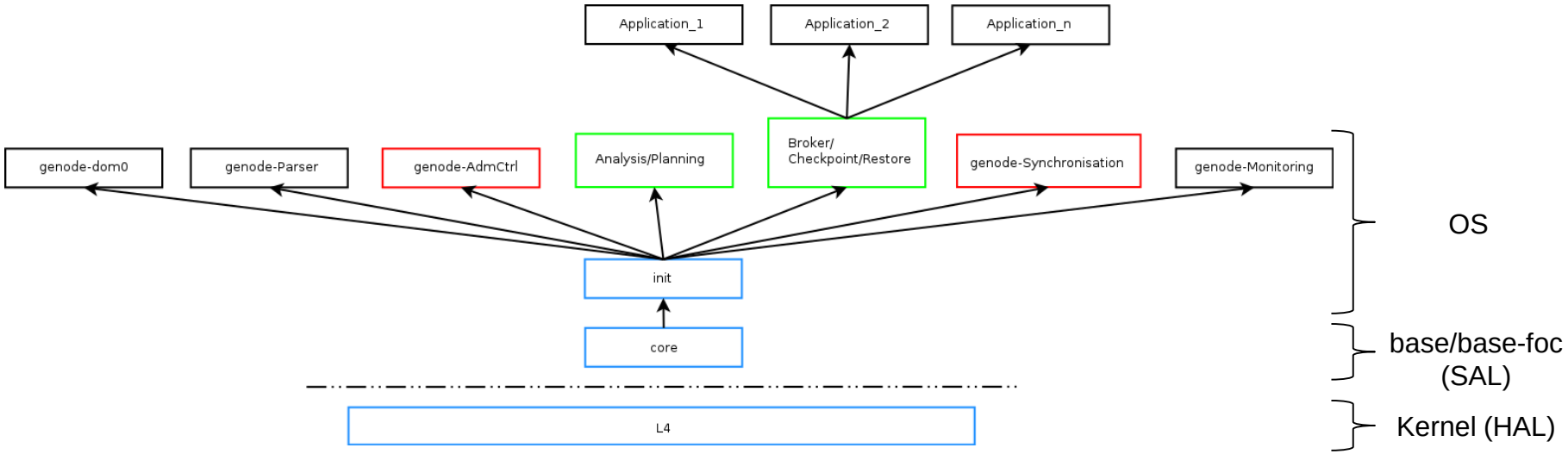


https://l4re.org/doc/l4re_intro.html



http://genode.org/documentation/general-overview/index

Hierarchical component structure



Agenda

- About
- Vision
- Project: KIA4SM
- Dynamic Reconfiguration
  - **Migration**
    - Checkpoint/Restore
    - Flexible Thread Management
  - Partitioning
- Test Setup
- Next Steps

Migration: OS components

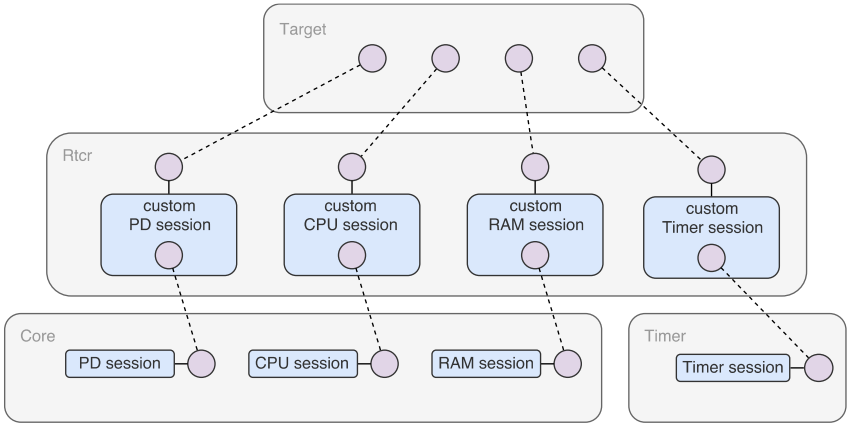
- Core OS (migration execution)
  - **Snapshot (Checkpoint/Restore, De/Serialization)**
  - **(Real-Time) Scheduling**
  - Storage (tmpfs)
  - Networking (dom0)
  - (User-)Data exchange (Broker, Publish/Subscribe)
- Observer/Controller (migration planning)
  - Runtime Monitoring
  - Analysis/Planning (Machine Learning)
  - Synchronisation
- Working components
  - Sensors/Actuators
  - Processing (e.g. ACC)
  - Dummy (e.g. load producing applications)

Core OS – Checkpoint/Restore

- Granularity
  - Full snapshot (binary + mutual data)
  - Partial/Diff snapshot (mutual data)
  - Incremental (mutual data -> dump vs. single pages)
- Overhead/Resource consumption
  - Executed on demand: start/stop (user input, system event/interrupt)
  - **Executed after scheduler preempt**
    - Executed periodically/time-triggered (flag tasks/memory area)
    - Executed occasionally/randomly (flag tasks/memory area)
- Timing
  - **After interruption/termination of task/thread (result/data)**
  - After termination of single instruction within task/thread
  - At any point in time
  - ...

Core OS – Checkpoint/Restore

- (Non) Real-Time / (Non) Determinism
  - Separate Memory [Halt process + get access rights + copy memory]
  - **Shared Memory** [May halt process + copy memory changes]
    - Start/Stop
    - Copy-on-write
  - **Redundant Memory** [May halt process + no copy of memory]
    - Write into three different memory regions in parallel during runtime
    - No need to copy data



Core OS – Checkpoint/Restore

- HW-support
  - **Multi-Core** / Co-Processor
  - FPGA / dedicated HW (C/R in HW)
- Snapshot location
  - Storage at original ECU (transfer at runtime)
  - Storage at dedicated ECU/repository (transfer at runtime)
  - Storage decentralized at different ECUs (possible transfer at runtime)
  - **Storage at (potential) target ECU (pre-planned)**
- Implementation Layer (Kernel vs. **Userland**)

Core OS – Flexible Thread Management

- Load Situations (Migration, Overload)
  - Variable task sets
- Three Phases
  - Admission (local planning)
  - Synchronization (deployment)
  - Scheduling (enforcement)
- Combination of best-effort/real-time tasks (Mixed-criticality)
  - Information about criticality level (safety)
  - Thread and Cores accordingly classified

Core OS – Flexible Thread Management

- Admission
  - Aggregation of adaptation strategies (reactive & knowledge-based)
  - Local planning component
- Synchronization
  - Concept
    - Deployment of new system state without (hazardous) side effects
    - Keep the system up and running (on-line)
  - Implementation
    - Basic idea: synchronize in idle mode
    - Enhanced idea: synchronize by utilizing slack time
- Scheduling
  - Focus on standard scheduling policies
  - Simple implementation to keep up with the idea of TCB

Core OS – Flexible Thread Management

- Implementation Layer

Alternative	Kernel-Space	User-Space
1)	Scheduler	Logic
2)	Scheduler/Logic	-
3)	-	Scheduler/Logic

- Overhead & Resource consumption

Migration Process (global)

- Runtime Monitoring
- Migration Planning (Machine Learning)
  - Analysis
  - Planning
- Migration
  - Checkpointing (save all necessary process data)
  - Serialisation (prepare data for transport)
  - Transfer (send data to target platform)
  - De-Serialisation (unpack data on target platform)
  - Restore (restart process on target platform)
  - OPTIONAL: Reorganisation (restore communication paths)

Machine Learning

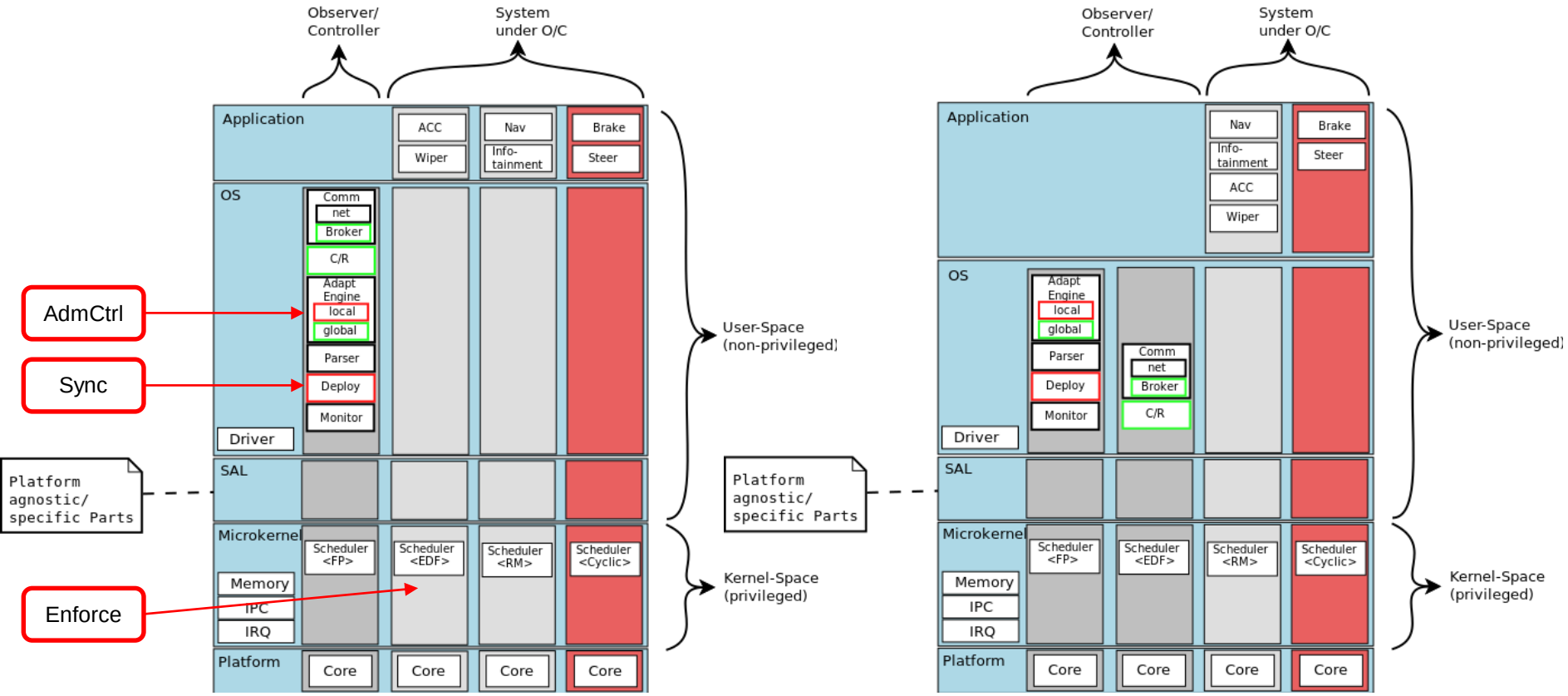
- Define different sorts of tasks (independent vs. dependent, dummy load, …)
- Define necessary task attributes (deadline, WCET, criticality level, type of scheduler e.g. FP/RR or EDF)
- Build suitable task sets (e.g. same task type, combine varying task types, task order…)

- Goal
  - Offline: Identify possible connection between task attributes and resource mapping (schedulability analysis)
  - Online: Create migration decisions based on identified relationships

Agenda

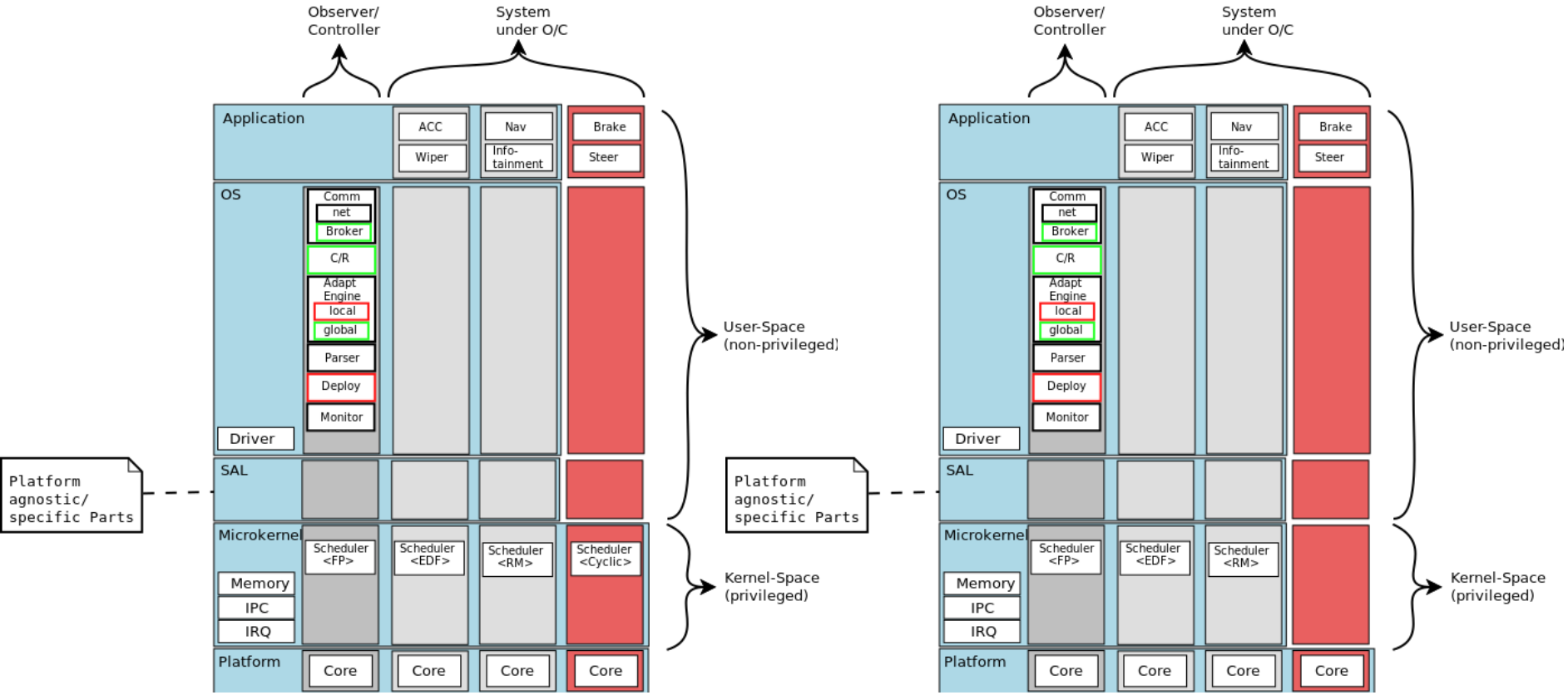
- About
- Vision
- Project: KIA4SM
- Dynamic Reconfiguration
  - Migration
    - Checkpoint/Restore
    - Flexible Thread Management
  - **Partitioning**
- Test Setup
- Next Steps

Partitioning – Concept I



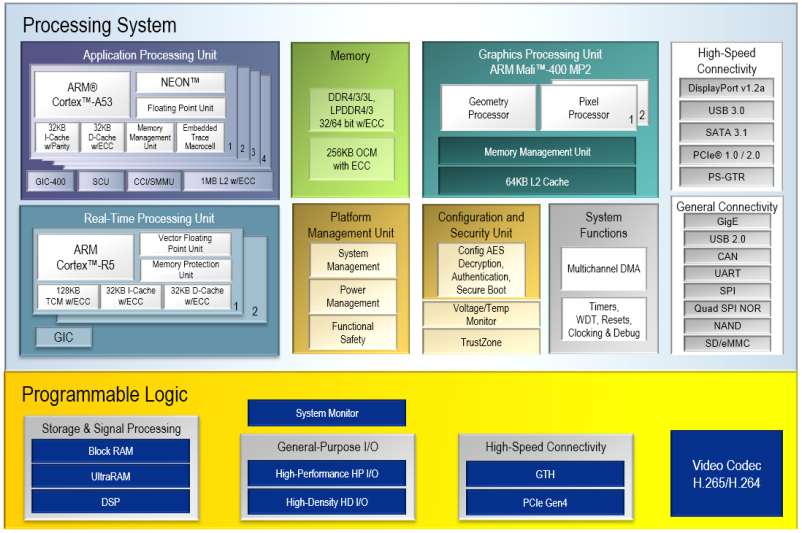
Combined management of best-effort, soft and hard real-time tasks.

Partitioning – Concept II

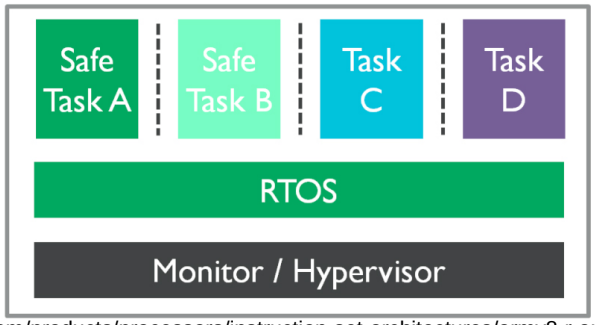


Virtually separated management of best-effort, soft and hard real-time tasks.

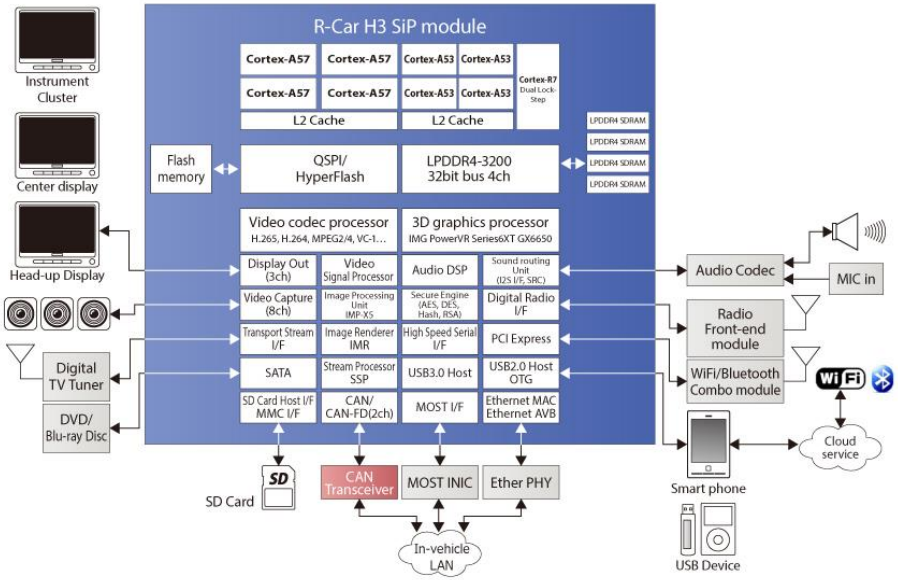
Background: Future Trends



https://www.xilinx.com/products/silicon-devices/soc/zynq-ultrascale-mpsoc.html

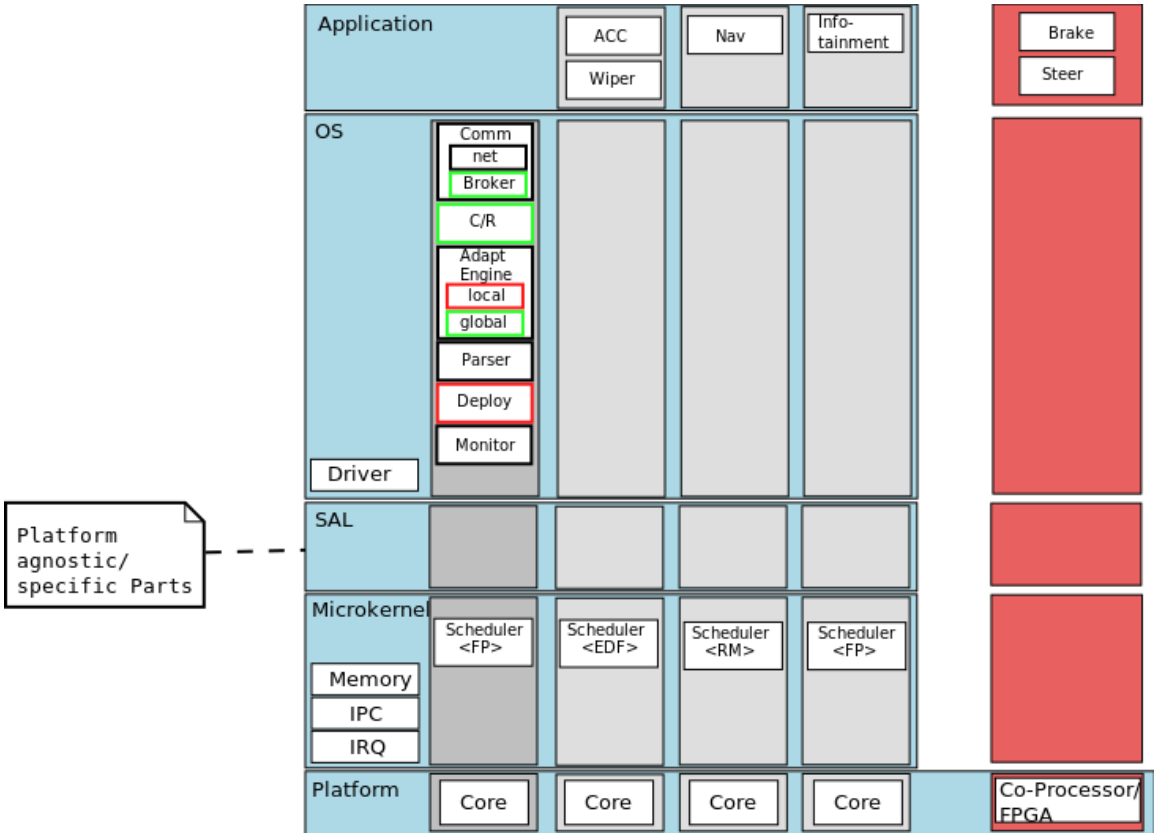


https://www.arm.com/products/processors/instruction-set-architectures/armv8-r-architecture.php



https://www.renesas.com/en-eu/solutions/automotive/products/rcar-h3.html

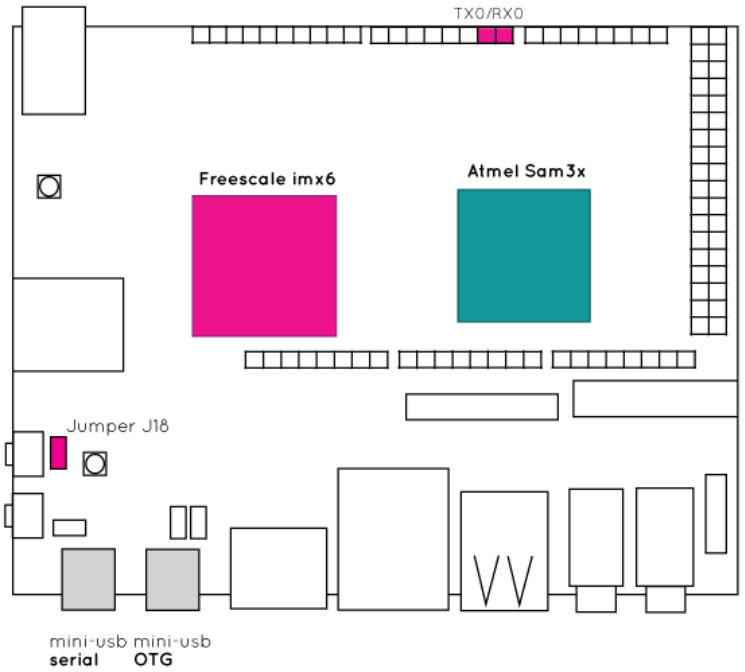
Partitioning – Concept III



Physically separated management of best-effort, soft and hard real-time tasks.

Partitioning – Prototypical Development

- UDOO Dual/Quad (NXP i.MX6 Cortex-A9 + Arduino Due-compatible ATMEL Cortex-M3)

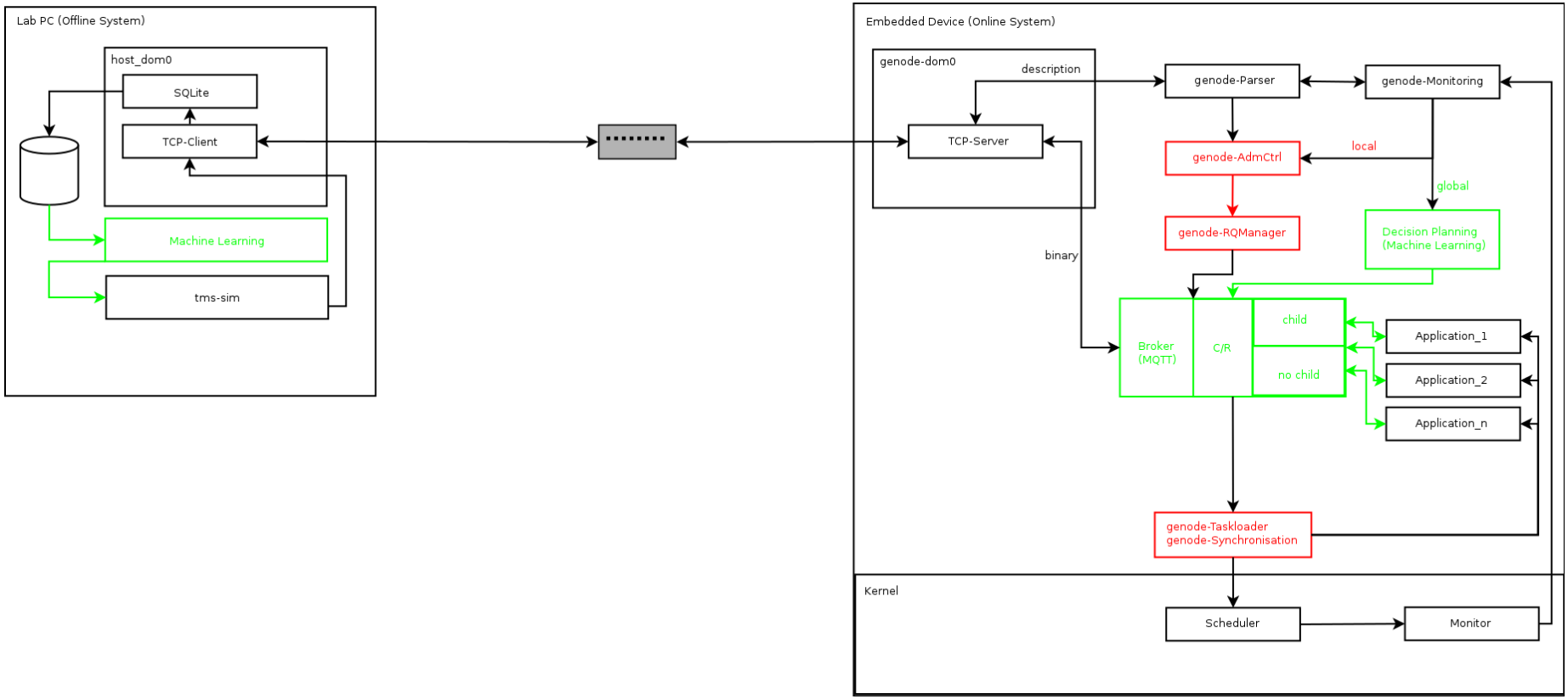


http://www.udoo.org/docs/Hardware_&_Accessories/IMX6_And_Sam3X_Communication.html

Agenda

- About
- Vision
- Project: KIA4SM
- Dynamic Reconfiguration
  - Migration
  - Partitioning
- **Test Setup**
- Next Steps

OS Workflow: Offline & Online systems

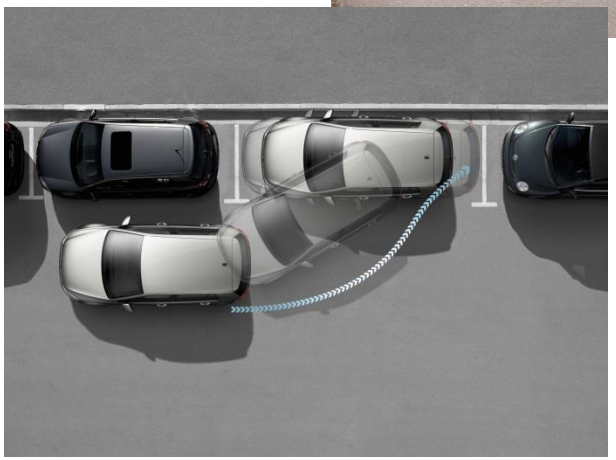


Test Case: Autonomous Driving

- Driving (in convoy)
  -> fail-operational

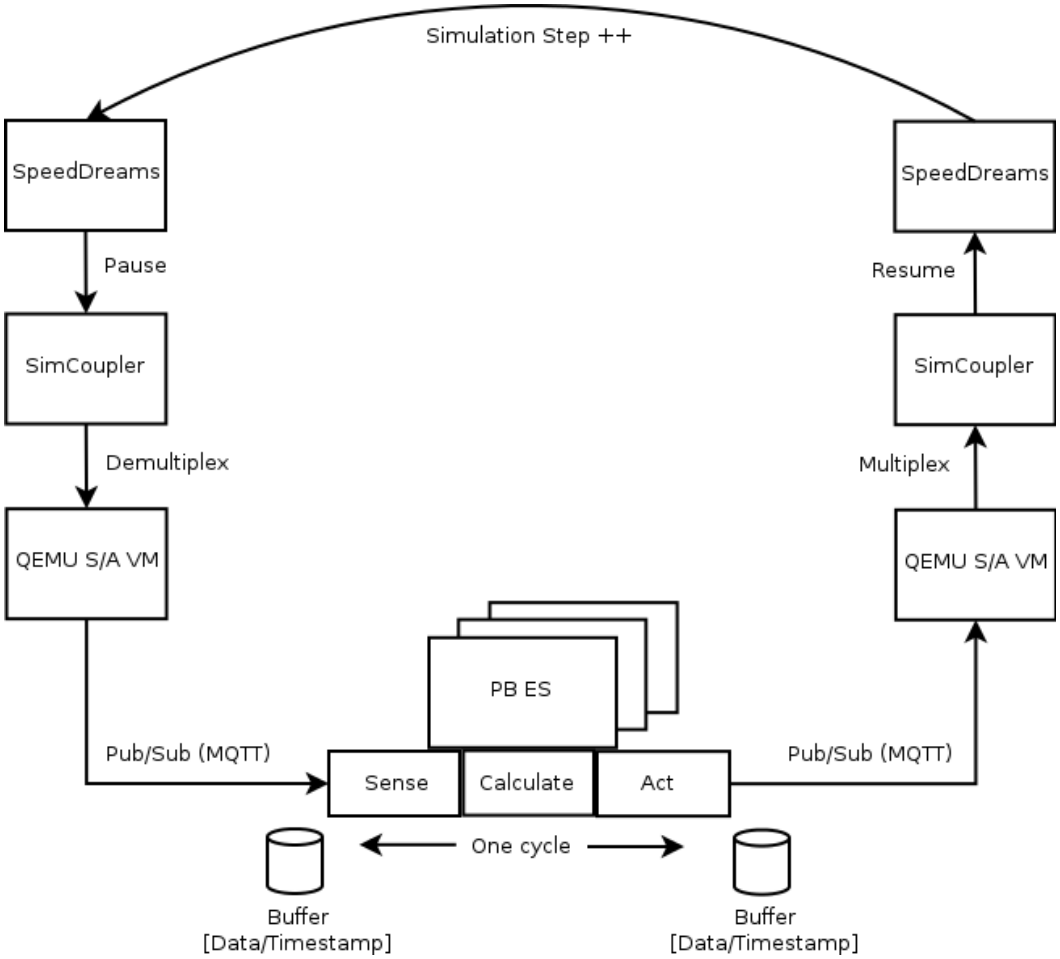
https://global.handelsblatt.com/wp-content/uploads/2016/04/Scania_11127-010_sm.jpg

- Autonomous Parking
  -> fail-safe



http://www.whichcar.com.au/media/1841/image94342_c.jpg?width=500&height=372.23168654173764

Connection: virtual & physical world



Agenda

- About
- Vision
- Project: KIA4SM
- Dynamic Reconfiguration
  - Migration
  - Partitioning
- Test Setup
- **Next Steps**

Conclusion

- Motivation: Future automotive systems
- Introduction of homogeneous integration architecture to combine virtual and physical world (OC-based microkernel)
- Migration as part of dynamic reconfiguration
- Clearly identified OS research areas
  - Migration planning
  - Checkpoint/Restore
  - Flexible Thread Management
- Test case & Setup
  - Implementation on top of L4 Fiasco.OC and Genode
  - COTS-Hardware in experimental model vehicle

Next Steps

- Evaluation (C/R)
  - Test of migration (+ de/serialization & transfer)
  - Test of migration + decision making
  - Test of migration + decision making + hybrid simulator
- Knowledge-based algorithms
  - Quantification of scheduling decision
  - Algorithms ranging from reactive (classic) to knowledge-based decision making
  - (Autonomous) optimization of decision for better system utilization
- L4/Genode Multi-Core (2+ cores) support
  - Porting of Genode to Quad-/Octa-Core platforms (e.g. Cortex-A9 based Freescale i.MX6 or Samsung Exynos SoCs)
  - Multi-Core placement of C/R, Monitoring and Analysis/Planning

Thank you!

https://www.os.in.tum.de/

Daniel Krefft, M.Sc.
TUM – Technische Universität München
Informatik F13 (Operating Systems)
Boltzmannstr. 3
85748 Garching
T: +49 89 289 18791
E: daniel.krefft@tum.de
www.os.in.tum.de