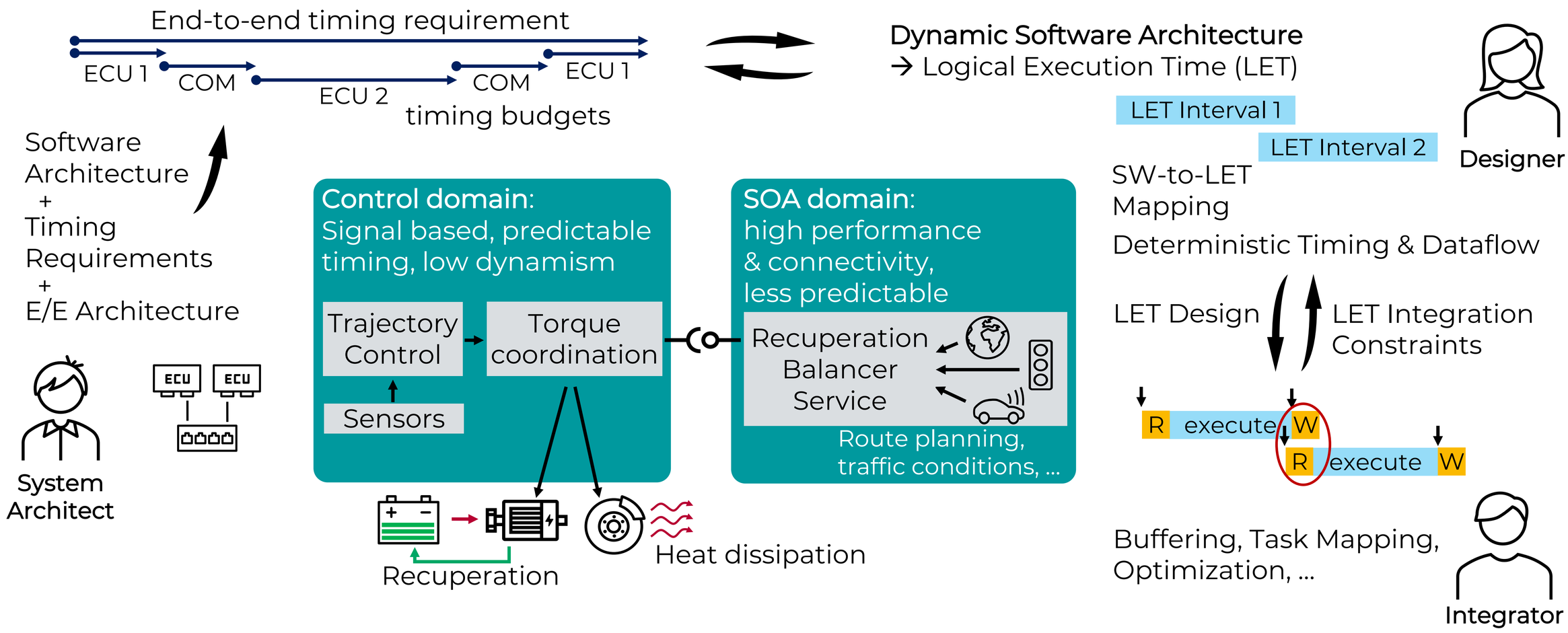
Deterministic Communication

Recuperation Balancing

autotech.agil

With funding from the:
Federal Ministry of Research, Technology and Space

Recuperation aims to reclaim energy during deceleration. This process involves balancing the electrical and mechanical brakes while managing the battery and thermal systems. Traditional control systems can be realized through a pure signal-oriented approach. Advanced recuperation systems use predicted track profiles, traffic jam information, and other cloud-based data to increase efficiency. This is the domain in which SOA can leverage its strengths. Thus, signal-oriented and SOA-based communication must cooperate to achieve system-wide functionality.

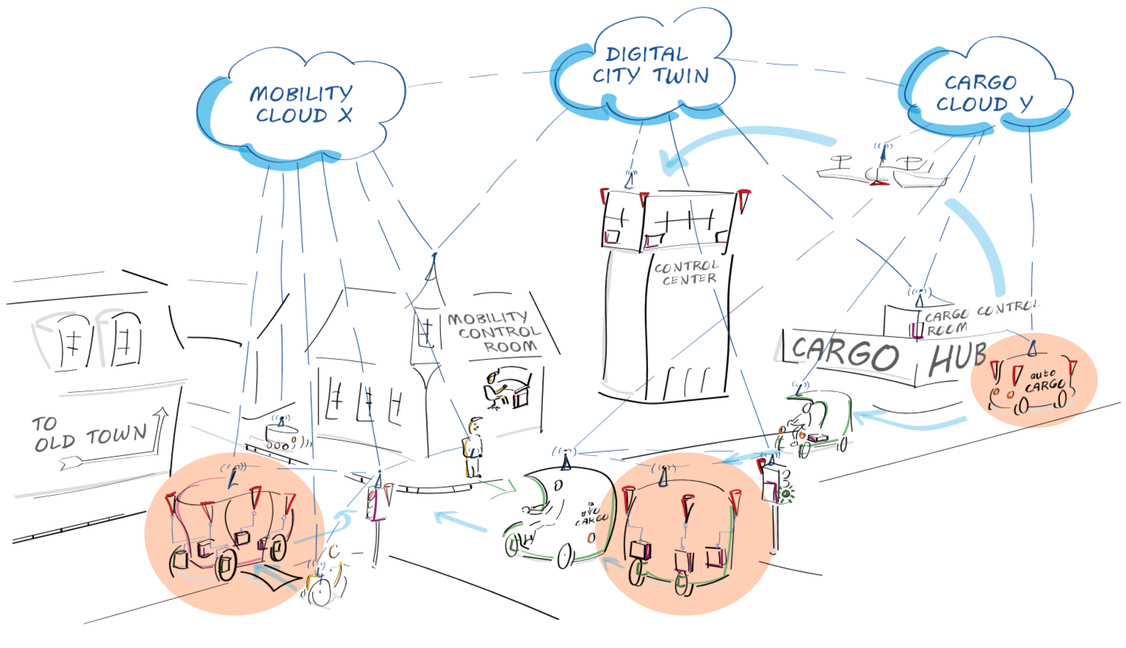


State of the Art

Logical execution time (LET) is a method for designing deterministic control systems. Originating in signal-oriented systems, such as AUTOSAR Classic, LET is challenged by the incorporation of SOA-based components, such as ASOA or AUTOSAR Adaptive. Additionally, dependencies between stakeholders' viewpoints introduce challenges in the development process. Thus, clear exchange formats and processes are necessary to prevent conflicts between design and integration.

Technological Innovations

Within autotech.agil, guidelines and models for the development of such tightly coupled systems have been introduced. These guidelines include the clear formulation of integration constraints to ensure consistency between LET specification and implementation. The **coupling of signal-oriented and SOA-based systems** has been systematically specified and applied to an industrially relevant system by enrolling **System-Level LET across AUTOSAR Classic and Adaptive ECUs**. Results have been fed back into automotive standardization.



Contributing Partners

Technische Universität Braunschweig · Mercedes-Benz · VECTOR · SCHAEFFLER

Deterministic Communication

Recuperation Balancing

autotech. agil

With funding from the:

Federal Ministry of Research, Technology and Space

Project Goals

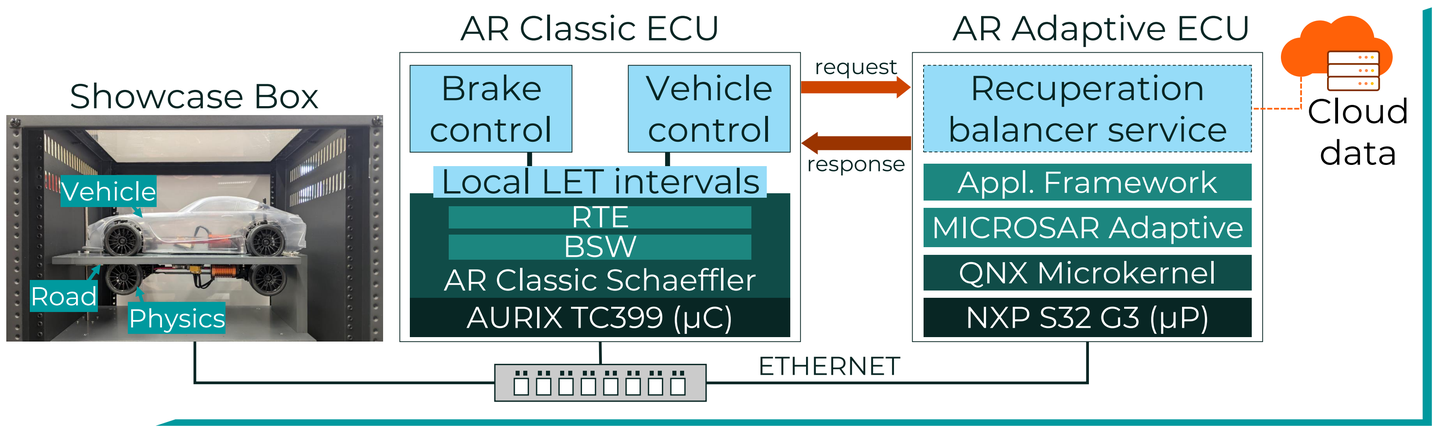
Coupling control & SOA domain for safety-critical applications using SL-LET

Define **automotive** software development **workflow with SL-LET**

Establish **interchange formats** between industrial stakeholders

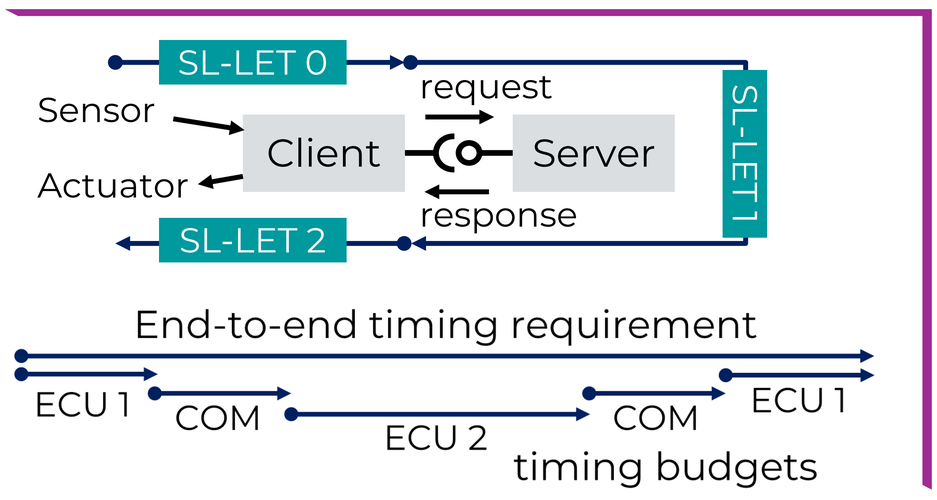
Apply findings to real-world **industrial use cases and tool chains**

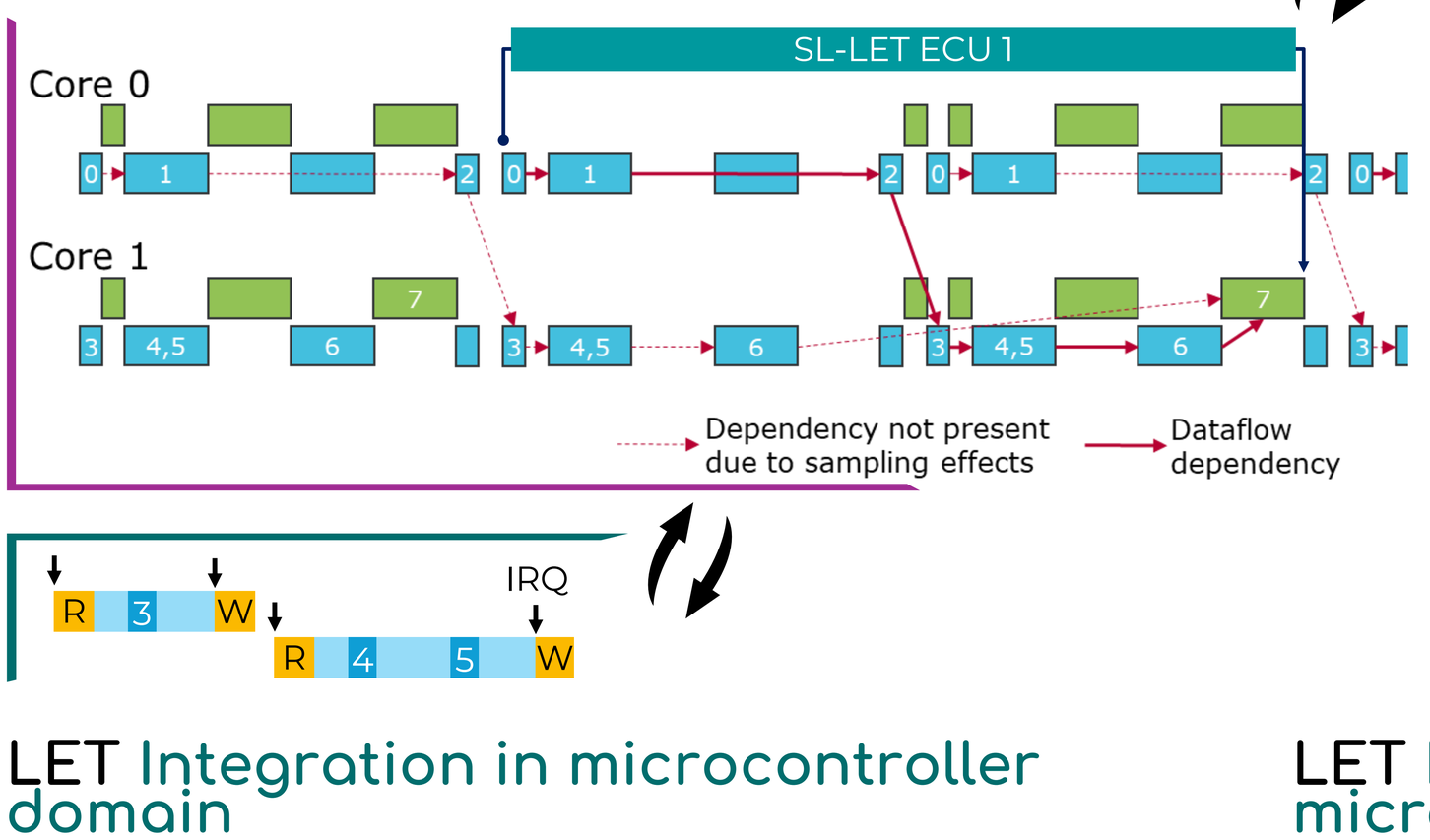
Demonstrator Setup



System Design with SL-LET

The coupling of a signal-oriented brake control system with a SOA based HPC platform is modeled in PREEvision. When SL-LET is applied to C/S communication, it can be partitioned into budgets for service call, execute and return (**D-LET**). Depending on platform capabilities, automatic translation to S/R communication is possible while preserving timing budgets.





LET Design

For each sub-system, periodic cause-effect chains are integrated into an existing LET schedule by mapping executable entities (0...7) to LET intervals. Data propagation depends on mapping, sampling effects and LET behavior. The formalized LET integration model (**I-LET**) enables designers to incorporate integration constraints into their decisions.

LET Integration in microcontroller domain

Deterministic execution and dataflow must be realized by proper implementation. This typically comprises configuration and optimization of data buffering, implementation of synchronized time phases for release, execute and terminate operations, as well as decisions on multicore distribution. Once the integration imposes constraints on design decisions, feedback to the LET designer is required. The proposed distinction between the D-LET (design) and I-LET (integration) models allows the formulation of constraints without disclosing intellectual property of the integrator.

LET Integration in microprocessor domain

Deploying SOA on a microprocessor aims for flexibility and performance. Unfortunately, event-based systems suffer from thread coordination and load peak issues, wherefore critical applications still focus on polling inputs. The planned determinism of SL-LET allows to optimize timing coordination where necessary while keeping flexibility where possible. Instead of polling inputs without coordination, a runtime on top of AUTOSAR Adaptive can coordinate activation when inputs are expected. Timing misbehavior can be detected and represented in form of quality metadata.