講演番号	036

Flexible Automotive Software Architecture utilizing Container Orchestration and PCIe

Lukas Stahlbock [1] **Jan Weber** [1] **Emrecan Cetin** [1] **Takuya Nigoro** [2] **Ise Shingo** [2]

1) IAV GmbH, Rockwellstraße 12, 38518 Gifhorn, Germany
2) IAV Japan Co. Ltd, Uchikanda Chuo Bldg. 3F, 1-18-13 Uchikanda, Chiyoda-ku, Tokyo, 101-0047 Japan

KEY WORDS: (standardized) **software and its underlying technologies, service-oriented architecture (SOA), container** [E3]
(free) software-defined vehicle (SDV), PCIe, container orchestration

Modern vehicles are increasingly evolving into Software Defined Vehicles (SDVs), where functions demand highly flexible and efficient software architectures. At the same time, compliance with functional safety (ISO 26262) and cybersecurity standards (UNECE R155/156, ISO 21434) imposes strict constraints on reconfiguration behavior and timing. In this context, the paper investigates how containerization and container orchestration can be combined with Peripheral Component Interconnect Express (PCIe) to increase flexibility in in-vehicle software orchestration, focusing specifically on container image distribution.

Flexibility is defined as the system's ability to reallocate, adapt, or evolve software components at runtime without violating safety constraints. To quantify flexibility, the work uses reconfiguration time and additional overhead: shorter and more predictable reconfiguration times, and lower CPU overhead, imply higher flexibility. Dynamic container orchestration is considered on a multi-ECU reference architecture. Each ECU runs a container-capable OS and orchestration framework; at vehicle level a central control ECU hosts the orchestrator control plane and a Composer for global service allocation and scheduling. A local container registry mirror inside the vehicle provides all needed container images, decoupling reconfiguration from backend connectivity and supporting predictable failover. Conventional Ethernet-based image distribution uses HTTP over TCP/UDP, incurring variable latency, retransmissions under load, and non-negligible CPU overhead. This reduces determinism for container pull times and forces conservative strategies such as pre-pulling images or static redundancy, limiting flexibility. The paper therefore proposes a PCIe-based data plane for intra-vehicle image distribution, leveraging PCIe's high bandwidth, low latency and credit-based flow control. Container metadata (manifests, tags) still follow the OCI Distribution Specification, but image layers (blobs) are transferred via PCIe using Remote Direct Memory Access (RDMA). An ECU that needs an image layer triggers an RDMA transfer from a registry-mirror ECU.

An experimental setup with Nvidia Jetson Orin NX boards interconnected via a PCIe Gen3 x2 switched network (Dolphin MXS924 and eXpressDrive stack) compares RDMA over PCIe with UDP and TCP over Ethernet. Round-trip time (RTT) measurements for various payload sizes are used as a proxy for container pull times. Fig. 1 shows the RTT ratio of UDP/TCP compared to RDMA and demonstrates that RDMA outperforms Ethernet-based communication for all investigated payload sizes; the relative benefit further increases with larger payloads, which is particularly relevant for multi-megabyte container layers. Fig. 2 presents the average CPU utilization. For small payloads, RDMA exhibits higher CPU usage due to polling overhead, but with increasing payload size, RDMA requires significantly less CPU than UDP/TCP. The break-even occurs around 1 MB payload, beyond which RDMA clearly reduces CPU load while keeping latency low and predictable. The results indicate that PCIe-based RDMA substantially shortens and stabilizes container image transfer times and lowers CPU overhead for large transfers, enabling dynamic redundancy with on-demand instantiation instead of statically duplicated services. Consequently, compute resources can be shared more efficiently, hardware overprovisioning is reduced, and resilience against ECU failures or security incidents is improved, as critical services can be quickly re-instantiated on alternative ECUs. Future work will examine the trade-off between polling and interrupt-based RDMA mechanisms in realistic automotive scenarios to further optimize the balance between latency and CPU utilization.

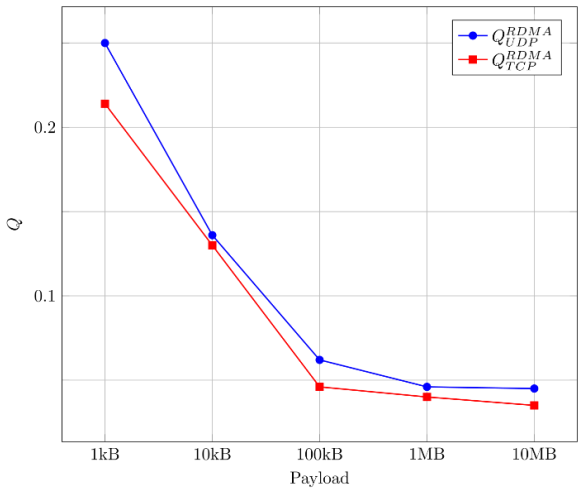

Fig. 1 RTT ratio for average data transfer latency with UDP and TCP compared to RDMA

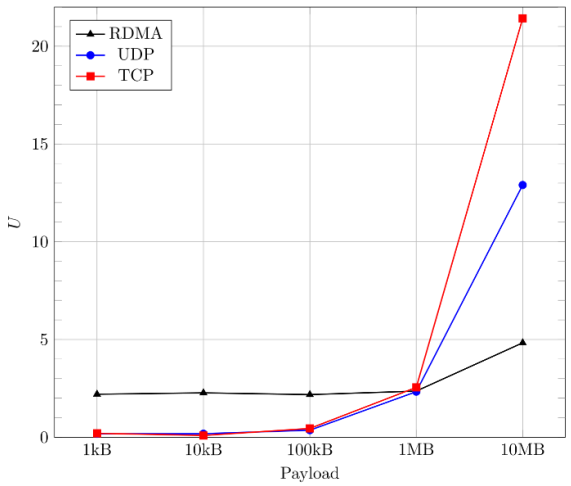

Fig. 2 Average CPU utilization for data transfer with RDMA, UDP, and TCP