Verification-Driven Scenario Optimization Across a Co-Simulation Continuum for AV Certification

Kurt Wilson*, Zhishan Guo*, Danjue Chen*, George List*, Frank Mueller*
{zguo32,dchen33,glist,fmuelle}@ncsu.edu
*North Carolina State University, Raleigh, NC 27695, USA

***Abstract*—Autonomous-vehicle (AV) functionality increasingly relies on machine-learned perception and tightly coupled driver-assist subsystems. Yet certification remains ad-hoc, with little guidance on what to verify, what to test, and when to re-test after software or hardware changes. We propose COSACC, a methodology and toolchain that (i) optimizes certification effort by deriving a minimal, coverage-effective battery of test scenarios from verification bounds, and (ii) carries proofs and constraints across a co-simulation continuum — from digital twin, through hybrid (real sensors/actuators in the loop), to full vehicle testing — so re-verification/re-certification is needed only when constraints actually break. Core ingredients of COSACC include compositional verification over subsystem dependence graphs, chain-aware mixed-criticality scheduling for ROS2 workloads, and WCET budget modeling on CPUs/GPUs/DSPs. We outline how the resulting constraint models inform test selection and retest decisions, and we sketch a validation plan spanning CARLA/ROS2, hybrid replay/streaming from a Lincoln MKZ, and controlled-facility runs.**

***Keywords*—certification, co-simulation, autonomous vehicle**

I. Introduction

Regulatory procedures (e.g., for Automatic Emergency Braking, AEB) use a handful of clean car-following scenarios with acceptance criteria, but they omit critical environmental and interaction effects (curvature, grade, adverse weather, oscillatory traffic) and the realities of cross-subsystem coupling, such as AEB with LKA (Lane Keeping Assist) or ACC (Adaptive Cruise Control). As a result, both coverage and guidance for what to (re)test when software or hardware changes are inadequate. Our project starts from the observation that no finite battery guarantees perfect safety; instead, one should maximize the likelihood of safe operation under explicit, defensible constraints, focusing tests on safety-critical, structured, and knowingly interrelated control situations that expose weaknesses.

We therefore unify compositional verification and constraint-aware testing. We will prove what is provable under explicit timing/functional bounds. We furthermore test only where proofs cannot safely generalize — and do so with an optimized scenario set that is sensitive to the fraction of the system that is virtual vs. real at each stage of the continuum.

II. COSACC Overview

Co-simulation continuum. COSACC executes the same software stack across (i) a digital twin (CARLA/Autoware/ROS2), (ii) a hybrid setup that incrementally replaces virtual devices with real sensors/actuators while streaming or replaying field data, and (iii) a controlled physical-vehicle environment (see Fig. 1). The key premise is that many verified properties remain valid across stages if designed with foresight; only constraint changes or violations trigger selective re-verification and targeted re-testing. Our plan leverages real-world data and a consistent compute architecture to keep timing/functional assumptions stable as we migrate from simulation to hardware.

Starting from NHTSA's AEB baseline [8], we expand to interactions among common Advanced Driver Assistance Systems (ADAS). We consider couplings such as Forward Collision Warning (FCW) with Automatic Emergency Braking (AEB), LKA with ACC, and Electronic Stability Control[1] (ESC) with ABS. Using these interactions, we construct a taxonomy of influential factors (environment, dynamics/speeds, traffic configurations, sensor degradations), identify edge-of-feasibility frontiers (where constraints are tight), and then optimize a small scenario battery that exercises these frontiers with minimal redundancy. We deliberately include coupling effects (e.g., delays in ACC→AEB mode switching that inflate stopping distance), which are absent from baseline procedures but are safety-critical.

III. Foundations that Tie Proofs to Tests

Executor-aware timing via UPPAAL. Real systems rely on middleware behaviors that defy simple textbook scheduling models. We therefore model the ROS2 single-thread executor (selection among timer and subscriber callbacks) in UPPAAL [2], yielding latency and response-time properties for realistic workloads [15], [16]. This framework covers time-driven and event-driven callbacks, adapts to various executor versions, and composes with a formal digital twin so that controller verification reflects middleware-induced interference and timing. The resulting timing values parameterize controller models and propagate as assumptions into later stages of the continuum.

Chain-level mixed-criticality analysis for ROS2. AV software is a publisher-subscriber DAG, but end-to-end guarantees emerge along processing chains. We therefore analyze chains under Vestal-style mixed criticality with two execution-time models — optimistic (OpET) and worst-case (WCET) —

[1]also known as Electronic Stability Program (ESP)

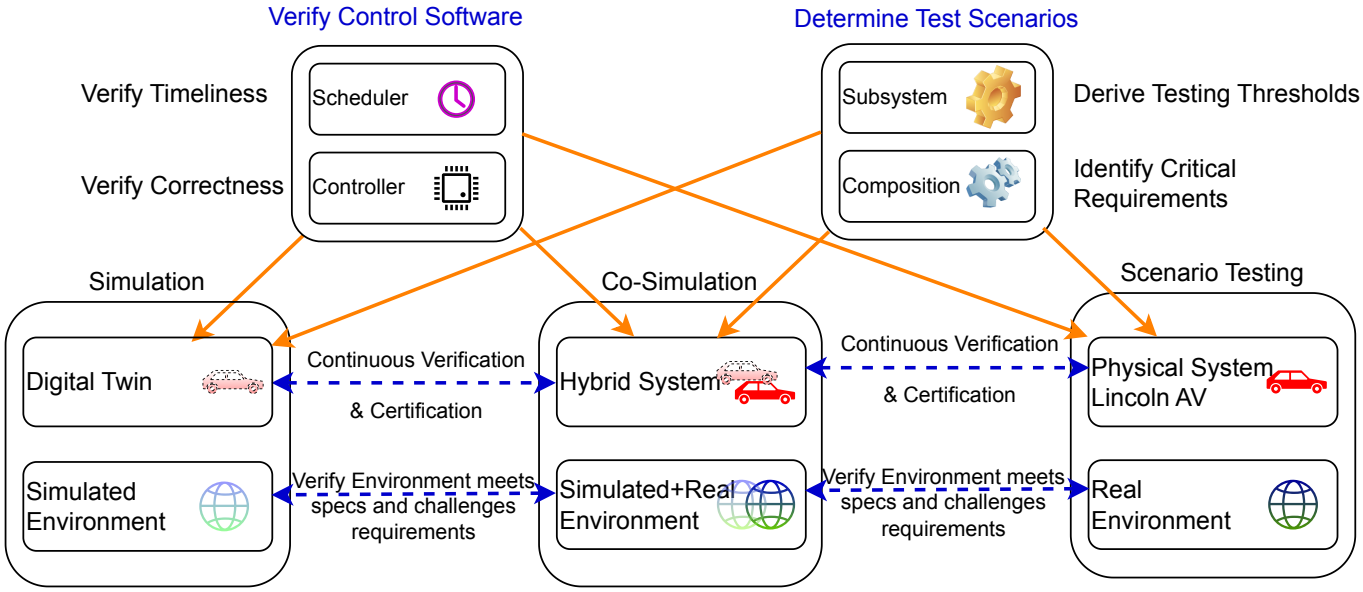

Fig. 1. Overview of Co-Simulation Continuum

and two operating modes (LO, HI) [14]. If any task exceeds OpET, the system switches to HI-mode, preserving only HI-critical chains. To reduce re-validation after updates, we partition chains to ECUs to avoid cross-chain schedulability causality, which is a knapsack-style optimization with efficient approximations. Formal analysis is complemented by demand-bound/supply-bound schedulability methods to compute end-to-end response times per chain with limited preliminary results [3], [4], [13], [17].

Stable compute budgets on heterogeneous platforms. Instead of brittle single-point WCET claims, we assign per-task compute budgets (for CPUs/GPUs/DSPs) at design time [5]–[7], [9]–[12]. Budgets, together with periods and deadlines, support schedulability analysis while providing headroom for system evolution so that adding a sensor or updating a model does not force global re-proving — unless observations escape the budget envelope, which triggers precise flags indicating where and what to re-verify or re-test.

IV. From Proofs to Tests: Scenario-Battery Optimization

Given (i) executor- and chain-aware timing bounds and (ii) budgeted resources, COSACC identifies the assumption set underpinning each guarantee: executor policy, callback timing, chain composition, mapping to ECUs, and platform budgets. Any change that leaves these assumptions intact need not trigger re-certification. Where assumptions do change, we localize their footprint via the chain/ECU partitioning and select scenarios that most directly interrogate endangered constraints (e.g., braking-distance margins under grade/curvature with ACC→AEB handover latency). Thus, the optimized scenario set solves a coverage problem over constraint frontiers rather than enumerating permutations.

To keep the battery robust across the continuum, we also require that scores and coverage remain comparable even as some components are virtual and others physical, and that sensitivity to the automation level (SAE 1-5) is explicit in the scenario design. This lets us defer expensive on-vehicle runs until after the hybrid stage, i.e., the system has already been tested and verified in terms of its constraints up to this hybrid model [1].

V. Status, Limitations, and Expected Contributions

Our preliminary executor models already support property checking for interacting components (e.g., ACC/AEB) and integrates with formal twins to expose timing-aware controller behavior. We do note limitations: executor modeling begins with a single-thread abstraction and will need to be extended to multi-threaded executors and heterogeneous callback priorities; measurement-based budgets must be resampled when architectural changes occur; and scenario optimization assumes that constraint frontiers are discoverable with feasible sampling effort. Even so, the approach is practical and designed for evolution. In particular, budgets and chain partitions are tuned so many updates do not ripple into global re-verification.

Planned contributions. (i) an executor- and chain-aware verification framework that captures middleware effects for realistic ROS2 workloads; (ii) a budgeting method for heterogeneous compute that stabilizes timing guarantees across system evolution; (iii) an optimization lens that turns certification into "what to verify, what to test, and when to retest" with minimal scenarios; and (iv) a demonstrated continuum that reduces costly re-verification while improving confidence in safety-critical behaviors.

VI. Conclusion

COSACC reframes AV certification as optimization under explicit constraints, aligning formal guarantees with empirical testing by targeting only those conditions where assumptions (and therefore safety margins) are at risk. By modeling what real systems actually execute (ROS2 executors, chain interactions, heterogeneous budgets) and by maintaining those constraints through a digital-to-physical continuum, COSACC promises leaner, more defensible certification: fewer tests, better boundary coverage, and principled decisions about when (not) to retest.

Acknowledgment

This work was supported in part by NSF awards CISE-2521121 and CMMI-2401555.

References

[1] Taxonomy and definitions for terms related to driving automation systems for on-road motor vehicles. https://www.sae.org/standards/content/j3016_202104, 2021.

[2] Johan Bengtsson, Kim Larsen, Fredrik Larsson, Paul Pettersson, and Wang Yi. Uppaal—a tool suite for automatic verification of real-time systems. In *Proceedings of the DIMACS/SYCON Workshop on Hybrid Systems III: Verification and Control: Verification and Control*, page 232–243, Berlin, Heidelberg, 1996. Springer-Verlag.

[3] Pontus Ekberg and Wang Yi. Outstanding paper award: Bounding and shaping the demand of mixed-criticality sporadic tasks. In *2012 24th Euromicro Conference on Real-Time Systems*, pages 135–144. IEEE, 2012.

[4] Haohan Li and Sanjoy Baruah. Outstanding paper award: Global mixed-criticality scheduling on multiprocessors. In *2012 24th Euromicro Conference on Real-Time Systems*, pages 166–175. IEEE, 2012.

[5] Sibin Mohan and Frank Mueller. Hybrid timing analysis of modern processor pipelines via hardware/software interactions. In *IEEE Real-Time Embedded Technology and Applications Symposium*, pages 285–294, 2008.

[6] Sibin Mohan and Frank Mueller. Merging state and preserving timing anomalies in pipelines of high-end processors. In *IEEE Real-Time Systems Symposium*, pages 467–477, December 2008.

[7] F. Mueller. Timing analysis for instruction caches. *Real-Time Systems*, 18(2/3):209–239, May 2000.

[8] Office of the Federal Register. Standard no. 127; automatic emergency braking systems for light vehicles. Code of Federal Regulations, 2024. Final rule published May 9, 2024.

[9] Xing Pan and Frank Mueller. Controller-aware memory coloring for multicore real-time systems. In *Symposium on Applied Computing*, April 2018.

[10] Xing Pan and Frank Mueller. Hiding dram refresh overhead in real-time cyclic executives. In *IEEE Real-Time Systems Symposium*, December 2019.

[11] H. Ramaprasad. *Analytical Bounding Data Cache Behavior for Real-Time Systems*. PhD thesis, Dept. of CS, North Carolina State University, July 2008.

[12] H. Ramaprasad and F. Mueller. Bounding worst-case response time for tasks under pip. In *IEEE Real-Time Embedded Technology and Applications Symposium*, pages 183–192, April 2009.

[13] Abhilash Thekkilakattil, Sanjoy Baruah, Radu Dobrin, and Sasikumar Punnekkat. The global limited preemptive earliest deadline first feasibility of sporadic real-time tasks. In *2014 26th Euromicro Conference on Real-Time Systems*, pages 301–310. IEEE, 2014.

[14] Steve Vestal. Preemptive scheduling of multi-criticality systems with varying degrees of execution time assurance. In *28th IEEE international real-time systems symposium (RTSS 2007)*, pages 239–243. IEEE, 2007.

[15] Kurt Wilson, Abdullah Al Arafat, John Baugh, Ruozhou Yu, and Zhishan Guo. Physics-informed mixed-criticality scheduling for f1tenth cars with preemptable ros 2 executors. In *2025 IEEE 31st Real-Time and Embedded Technology and Applications Symposium (RTAS)*, pages 215–227. IEEE, 2025.

[16] Kurt Wilson, Abdullah Al Arafat, John Baugh, Ruozhou Yu, Xue Liu, and Zhishan Guo. Soteria: A formal digital-twin-enabled framework for safety-assurance of latency-aware cyber-physical systems. In *Proceedings of the 28th ACM International Conference on Hybrid Systems: Computation and Control*, pages 1–11, 2025.

[17] Quan Zhou, Guohui Li, Jianjun Li, Chenggang Deng, and Ling Yuan. Response time analysis for tasks with fixed preemption points under global scheduling. *ACM Transactions on Embedded Computing Systems (TECS)*, 18(5):1–23, 2019.