The Journal of Open Source Software

CAWSR: Carla-AutoWare Scenario Runner

David Gasinski [1], **Olek Osikowicz** [1], **Gwilym Rutherford** [1], and **Donghwan Shin** [1]

1 The University of Sheffield

DOI: 10.xxxxxx/draft

Software

- Review
- Repository
- Archive

Editor: Hugo Ledoux

Reviewers:

- @tensorsofthewall
- @nitishsanghi

Submitted: 22 December 2025
Published: unpublished

Summary

CAWSR (**C**ARLA-**A**uto**W**are-**S**cenario **R**unner) facilitates the simulation-based testing of the open-source autonomous driving system, Autoware, within CARLA, the state-of-the-art open-source driving simulator. Building on existing tools, this project introduces a research-oriented testing framework for the execution of complex driving scenarios, as well as supporting the implementation of a wide range of verification strategies.

Statement of Need

Verifying Autonomous Driving Systems (ADS) is a critical step before they can be deployed. However, relying only on real-world testing is too expensive, inefficient, and potentially dangerous. Consequently, simulation-based testing has become essential, allowing researchers to safely test driving agents against critical situations at scale. Among these tools, CARLA (Dosovitskiy et al., 2017) has become the de-facto standard in the research community due to its rich ecosystem of open-source tools, benchmarks, and documentation.

Currently, the standard for evaluating ADS in CARLA is the CARLA Leaderboard and its engine, Scenario Runner (SR) (CARLA, 2025). This framework is typically used to test "black-box" driving agents, such as ML-based systems which expose only sensor-level inputs and driving control outputs. By running a set of predefined, challenging driving scenarios, researchers can systematically assess agent performance using common metrics like driving score, infractions, and route completion. However, applying this testing framework to industry-grade ADS, such as Autoware (Kato et al., 2018) or Apollo (Baidu, 2017), remains difficult. Although communication bridges exist between CARLA and these systems (Guardstrikelab, 2023; Kaljavesi et al., 2024), they lack native support for scenario execution engines, which limits their utility for scenario-based testing.

This gap has created a significant bottleneck for the research community. Previously, researchers developing scenario generation algorithms mainly relied on combining Apollo with the LGSVL simulator (Rong et al., 2020). However, LGSVL is now outdated, with official support ending in January 2022. This leaves many researchers without a suitable industry-grade "subject" for evaluating their algorithms. While recent tools like PCLA (Tehrani et al., 2025) attempt to simplify deploying Autoware (and other ADS implementations) into CARLA, they focus primarily on simplifying the ADS implementations and abstracting the setup process across different CARLA versions. They lack the deep integration required between the agent and simulator to execute complex, route-based scenarios.

CAWSR aims to bridge this gap by enabling the evaluation of Autoware in complex driving scenarios within CARLA. By building on the established CARLA platform, this work provides a modern replacement for the outdated Apollo/LGSVL workflow. It also allows Autoware to be directly compared with state-of-the-art research agents on the CARLA Leaderboard.

Effective ADS verification requires the ability to systematically explore the operational design domain. To support this, CAWSR provides a flexible interface for algorithmic scenario generation. This facilitates a wide range of verification strategies based on common metrics, such as the CARLA Leaderboard's driving score (CARLA Team, 2024).

Lastly, it is worth noting that simulators can often introduce unintended nondeterminism, which leads to inconsistent test results (Chance et al., 2022; Osikowicz et al., 2025). Therefore, CAWSR is designed to minimise such nondeterminism throughout the evaluation pipeline.

Research Impact

CAWSR provides a significant research impact by bridging the gap between the widely-used CARLA simulator and the industry-grade Autoware ADS. It addresses a critical bottleneck in the ADS testing research community by offering a modern replacement for the outdated Apollo/LGSVL workflow, which has lacked official support since 2022.

It enhances research reproducibility by implementing a fully synchronous evaluation pipeline that minimises unintended nondeterminism in simulation-based testing. To ensure community readiness and ease of use, it is distributed as a Docker container.

Furthermore, by adopting CARLA Leaderboard metrics, CAWSR enables researchers to directly compare Autoware with other state-of-the-art driving agents in the CARLA environment. It provides an essential foundation for the systematic evaluation of various testing approaches for ADS.

Software Design

CAWSR is a fully synchronous testing framework that directly integrates the CARLA simulator, Scenario Runner (as the scenario executor), and Autoware (as the System Under Test) to facilitate autonomous driving testing research. The tool is distributed as a containerized deployment using Docker to manage complex dependencies and simplify the setup process. Currently, two modes of operation are supported:

1. *Scenario Generation Mode:* Enables the dynamic generation and execution of scenarios (e.g. iterative scenario generation) provided by a user-defined algorithm. This is particularly useful for assessing the performance of new simulation-based ADS testing techniques.
2. *Benchmark Mode:* Allows the execution of a predefined set of scenario definitions provided by the user. This is useful for standardised evaluations and comparisons between different driving agents.

The evaluation pipeline is engineered to be fully synchronous, minimising unintentional non-determinism to facilitate reproducible results. However, it is noted that minor variations may still persist due to inherent non-determinism in upstream dependencies, such as the driving simulator or the driving agent itself (Chance et al., 2022; Osikowicz et al., 2025).

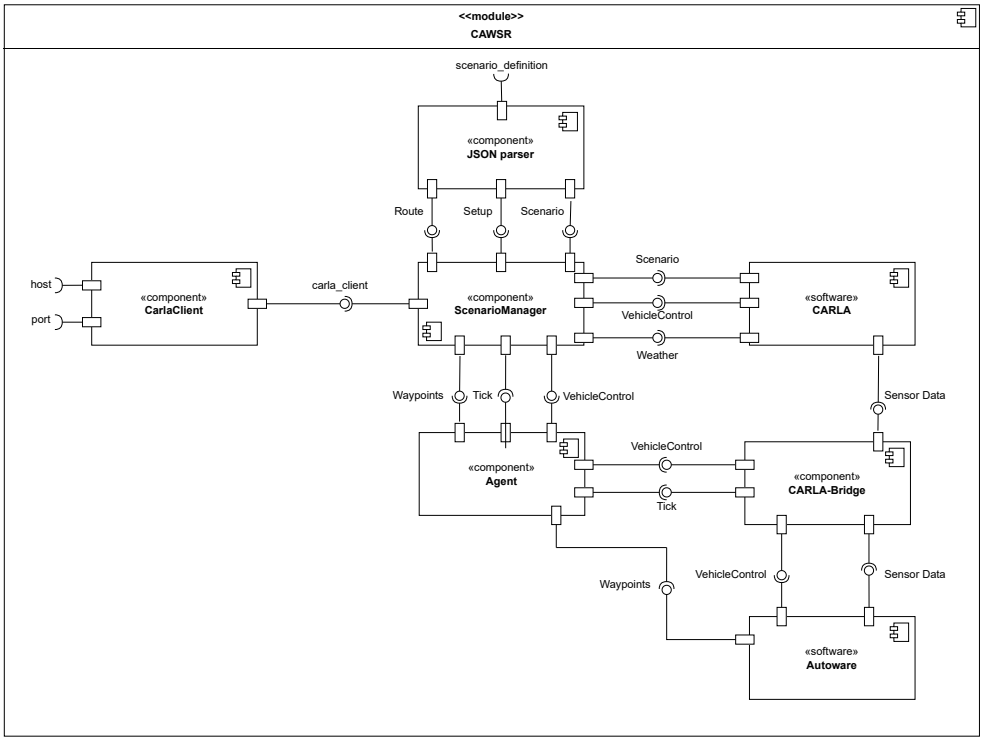

Figure 1: Internal component diagram of CAWSR.

Figure 1 illustrates the CAWSR architecture and its fundamental components. The framework operates through four primary modules:

- CarlaClient: A native CARLA PythonAPI class that establishes a TCP connection (via host IP and port). It serves as the framework's exclusive interface for extracting simulation data and spawning entities.
- JSON Parser: Translates the *scenario_definition* (see Figure 2) into a Behavior Tree (BT). It utilises Scenario Runner's *Atomic Behaviours* and *Atomic Conditions* as modular primitives to define discrete actions (e.g., spawning pedestrians) and logic triggers.
- ScenarioManager (CARLA, 2025): Orchestrates the simulation loop by evaluating the BT to update actor states and triggering CARLA simulation ticks. Execution terminates based on CARLA Leaderboard criteria (CARLA Team, 2024), as summarised in Table 1. Post-execution, the module calculates the Driving Score (DS) according to the official leaderboard metrics.
- Agent and CarlaBridge: The Agent manages the ROS2 connection to Autoware. At each timestep, the CarlaBridge (Kaljavesi et al., 2024) transforms CARLA snapshots and sensor data into the Autoware coordinate system. Autoware processes these inputs to issue control commands, which the Agent then applies to the ego vehicle.

Table 1: Termination Criteria of each scenario within CAWSR.

Termination Criteria	Description
Route_Completion	Agent reached the end of the route.
Actor_Blocked	Agent is blocked, not moving for 180s.
Simulation_Timeout	No client-server communication established (30s).

To facilitate development, we introduce a new domain model for the definition of route-based scenarios within CARLA, described in Figure 2, alongside a `JSON` implementation. This model is based on the format introduced by Scenario Runner, facilitating support between both frameworks.

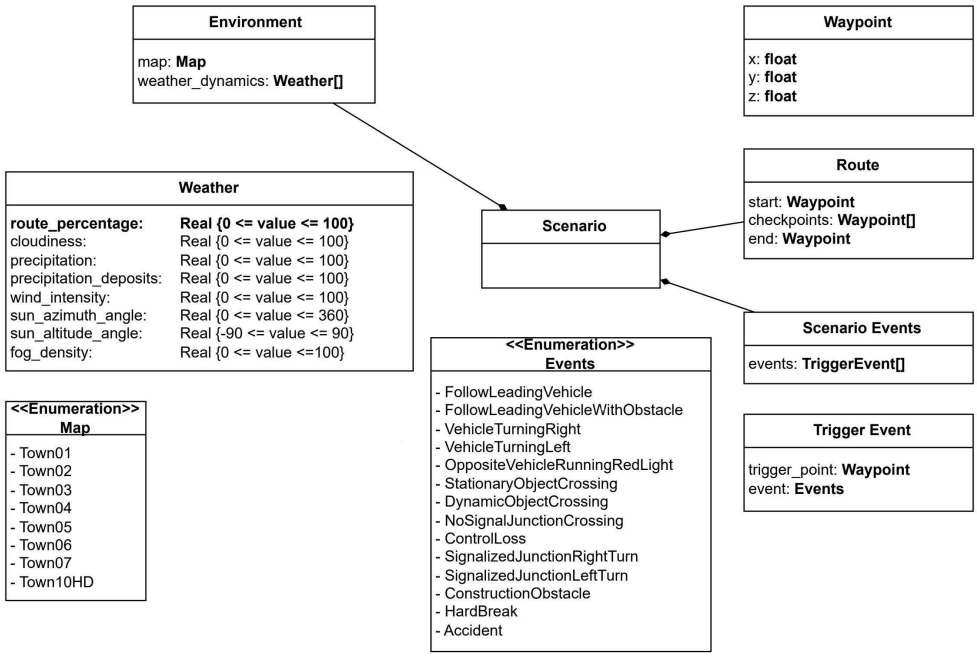

Figure 2: Scenario definition domain model.

Conclusion

To summarise, `CAWSR` provides ADS testing research community an easy to use Autoware evaluation pipeline. We hope that this work can facilitate the evaluation of new testing approaches on a state of the art driving system.

AI Usage Disclosure

Generative AI tools were used in this work solely to support high-level research concepts and structural ideas. All software implementation, including the source code, architecture, and deployment scripts, was authored entirely by the researchers without AI assistance.

Acknowledgements

This work was supported by the Institute of Information & Communications Technology Planning & Evaluation(IITP) grant funded by the Korea government(MSIT) (No. RS-2025-02218761, 50%) and by the Engineering and Physical Sciences Research Council (EPSRC) [EP/Y014219/1].

References

Baidu. (2017). *Apollo: Open Source Autonomous Driving*. https://github.com/ApolloAuto/apollo.

CARLA. (2025). *Scenario Runner: Traffic Scenario Definition and Execution Engine for CARLA*. GitHub; https://github.com/carla-simulator/scenario_runner. https://github.com/carla-simulator/scenario_runner

CARLA Team. (2024). *Get started with leaderboard 2.0. CARLA autonomous driving leaderboard*. https://leaderboard.carla.org/get_started_v2_0/

Chance, G., Ghobrial, A., McAreavey, K., Lemaignan, S., Pipe, T., & Eder, K. (2022). On determinism of game engines used for simulation-based autonomous vehicle verification. *IEEE Transactions on Intelligent Transportation Systems*, *23*(11), 20538–20552. https://doi.org/10.1109/TITS.2022.3177887

Dosovitskiy, A., Ros, G., Codevilla, F., Lopez, A., & Koltun, V. (2017). CARLA: An open urban driving simulator. *Proceedings of the 1st Annual Conference on Robot Learning*, 1–16.

Guardstrikelab. (2023). *carla_apollo_bridge: Data and Control Bridge for Apollo and Carla*. GitHub; https://github.com/guardstrikelab/carla_apollo_bridge.

Kaljavesi, G., Kerbl, T., Betz, T., Mitkovskii, K., & Diermeyer, F. (2024). *CARLA-autoware-bridge: Facilitating autonomous driving research with a unified framework for simulation and module development*. 224–229. https://doi.org/10.1109/iv55156.2024.10588623

Kato, S., Tokunaga, S., Maruyama, Y., Maeda, S., Hirabayashi, M., Kitsukawa, Y., Monrroy, A., Ando, T., Fujii, Y., & Azumi, T. (2018). Autoware on board: Enabling autonomous vehicles with embedded systems. *2018 ACM/IEEE 9th International Conference on Cyber-Physical Systems (ICCPS)*, 287–296. https://doi.org/10.1109/iccps.2018.00035

Osikowicz, O., McMinn, P., & Shin, D. (2025). Empirically evaluating flaky tests for autonomous driving systems in simulated environments. *2025 IEEE/ACM International Flaky Tests Workshop (FTW)*, 13–20. https://doi.org/10.1109/FTW66604.2025.00009

Rong, G., Shin, B. H., Tabatabaee, H., Lu, Q., Lemke, S., Možeiko, M., Boise, E., Uhm, G., Gerow, M., Mehta, S., Agafonov, E., Kim, T. H., Sterner, E., Ushiroda, K., Reyes, M., Zelenkovsky, D., & Kim, S. (2020). LGSVL simulator: A high fidelity simulator for autonomous driving. *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*, 1–6. https://doi.org/10.1109/ITSC45102.2020.9294422

Tehrani, M. J., Kim, J., & Tonella, P. (2025). PCLA: A framework for testing autonomous agents in the CARLA simulator. *Proceedings of the 33rd ACM International Conference on the Foundations of Software Engineering*, 1040–1044. https://doi.org/10.1145/3696630.3728577