International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING
ISSN:2147-6799 www.ijisae.org Original Research Paper

A Lifecycle-Centric Hardware-in-the-Loop Validation Framework for Continuous Verification of Software-Defined Automotive Chassis and Powertrain Systems

Sana Fatima

Abstract: The transition to software-defined vehicle (SDV) architectures has fundamentally altered the validation challenge for automotive chassis and powertrain embedded control systems. In traditional development cycles, validation is a pre-deployment activity: software is verified against a fixed functional specification before release, and the system is assumed stable thereafter. Software-defined vehicles dissolve this assumption entirely. Over-the-air (OTA) software updates, continuous calibration adjustments, and feature extensions delivered post-deployment create a time-varying system model in which the behavioral correctness of chassis and powertrain controllers must be re-established after every change. Existing Hardware-in-the-Loop (HIL) validation practices, designed around pre-deployment milestone verification, are structurally inadequate for this continuous, lifecycle-spanning verification demand. This paper proposes a lifecycle-centric HIL validation framework that extends HIL-based verification across development, deployment, and post-deployment phases through three integrated mechanisms: automated validation pipelines that execute HIL regression suites within 24 hours of each software update, digital twin feedback loops that continuously enrich the HIL scenario library from field-collected operational data, and embedded functional safety validation that evaluates diagnostic coverage and fault detection latency against ISO 26262 requirements after each update cycle. The framework is evaluated across 180 software update regression cycles spanning six months of simulated post-deployment operation on a dSPACE SCALEXIO platform. Key results include: a first-pass HIL success rate of 73.1%, a mean validation turnaround time of 19.4 hours, a diagnostic coverage of 91.3%, a fault detection latency of 44 ms, and a regression escape rate of 0.8%. These results demonstrate that lifecycle-centric HIL validation provides the continuous assurance capability that software-defined automotive architectures require.

Keywords: *Hardware-in-the-Loop, Software-Defined Vehicles, Lifecycle Validation, Digital Twin, Continuous Verification, Functional Safety, Automotive Embedded Systems*

I. Introduction

Automotive chassis and powertrain software systems have historically been validated through a well-defined pre-deployment verification cycle anchored to the V-model development process. Software functionality is frozen at a release milestone, HIL and bench validation activities verify conformance to the functional specification, and the validated system is deployed to production vehicles in a configuration that remains largely static throughout the vehicle's service life. This model was adequate when vehicle software changed infrequently — typically only through service campaigns triggered by known defects. The emergence of software-defined vehicle (SDV) architectures has rendered this assumption obsolete. Modern chassis and powertrain systems receive OTA software updates that introduce new control strategies, modify calibration parameters, and extend feature sets at intervals measured in weeks rather than years [1]. Each update creates a new system configuration that has not been validated in the pre-deployment HIL environment, reintroducing behavioral uncertainty into a fleet of vehicles already in customer operation.

Independent Researcher, USA
ORCID: 0009-0001-9872-0035

The validation deficit created by continuous OTA deployment is substantial. A software update that modifies AWD torque distribution logic may alter the interaction dynamics between the traction control system and the electronic stability program (ESP) in ways that the original pre-deployment HIL suite did not exercise, because the test library was built for the original software version. Field data collected from vehicle fleets post-deployment consistently reveals edge-case operating conditions, sensor fault patterns, and environmental scenarios that were absent from the pre-deployment test corpus [2]. These field-discovered scenarios represent an expanding frontier of unvalidated behavioral territory that grows with every deployment cycle. Without a systematic mechanism to incorporate field-discovered scenarios into the HIL validation environment and re-evaluate behavioral correctness after each software update, the behavioral safety margin of the deployed fleet erodes continuously over time.

This paper addresses this validation deficit by proposing a lifecycle-centric HIL validation framework with four explicit contributions. First, it extends HIL-based validation from a pre-deployment milestone activity to a continuous process spanning the development, deployment, and post-deployment phases of the vehicle

lifecycle. Second, it integrates a digital twin feedback loop that converts field-collected operational data into new HIL validation scenarios, ensuring that the test library reflects the actual operating conditions encountered by the deployed fleet. Third, it presents an automated validation pipeline that executes full HIL regression suites within a 24-hour turnaround window, enabling every OTA software update to receive HIL verification before fleet deployment. Fourth, it embeds continuous functional safety validation within the lifecycle framework, computing diagnostic coverage, fault detection latency, and safe-state transition time after each update cycle and comparing them against ISO 26262 safety requirements. The framework is evaluated across 180 simulated OTA update regression cycles on a dSPACE SCALEXIO platform, providing quantitative performance data across all four contribution dimensions.

The remainder of this paper is organized as follows. Section II presents the background on HIL simulation, digital twin technology, and continuous integration in automotive embedded systems. Section III describes the lifecycle-centric HIL validation architecture. Section IV details the digital twin feedback loop and scenario evolution mechanism. Section V presents the automated validation pipeline and real-time HIL execution approach. Section VI describes the continuous functional safety validation methodology. Section VII reports experimental results. Section VIII concludes the paper.

II. Background and Technical Foundations

Hardware-in-the-Loop simulation provides real-time, closed-loop validation of embedded control software by executing the software under test against simulated plant models that reproduce the physical dynamics of the vehicle, its subsystems, and the operating environment. Unlike model-in-the-loop (MIL) or software-in-the-loop (SIL) approaches, HIL incorporates actual hardware execution of the control software with representative timing behavior, communication latency, and sensor interface characteristics, providing validation fidelity that is significantly closer to production deployment conditions [3]. The dSPACE SCALEXIO platform represents the current state of the art in automotive HIL infrastructure, supporting multi-domain real-time simulation at sample rates from 1 ms for safety-critical brake and traction control functions to 10 ms for powertrain supervisory layers. In the automotive V-cycle, HIL validation occupies the system integration and system test stages, where the behavior of assembled software components is verified against vehicle-level functional requirements. The limitation of this conventional HIL deployment is its temporal scope: validation is performed once, before software deployment, against a fixed scenario library.

Digital twin technology provides the conceptual and architectural foundation for extending HIL validation into the post-deployment lifecycle. A digital twin is a continuously synchronized virtual representation of a physical system, maintained through real-time or near-real-time data exchange between the physical system and its model [4]. In the automotive context, digital twins of chassis and powertrain systems receive operational data — driver behavior signatures, environmental exposure, sensor health indicators, and control system state histories — from vehicle telemetry systems, enabling the virtual model to reflect the actual operating conditions of the deployed fleet rather than the design-basis assumptions embedded in the pre-deployment test corpus. This continuous synchronization property is the mechanism that bridges field-discovered operational scenarios to the HIL simulation environment, enabling post-deployment validation to exercise behavioral territory that pre-deployment validation cannot anticipate. Recent surveys confirm that digital twin adoption in the automotive sector has accelerated significantly, with the automotive and transportation segment representing the largest revenue share of the global digital twin market [5].

Continuous integration (CI) practices from software engineering provide the automation infrastructure required to sustain lifecycle-spanning HIL validation at the frequency demanded by OTA update programs. CI-enabled HIL validation frameworks automate the sequence of change impact analysis, test case selection, HIL execution, result aggregation, and deployment decision — compressing the validation cycle from weeks to hours and enabling HIL verification of every software update rather than only milestone releases [2]. The automotive industry has begun adopting CI-enabled HIL workflows, driven by the recognition that OTA deployment velocity is incompatible with manual HIL test management [6]. The key performance targets established for CI-enabled automotive HIL pipelines — validation turnaround time of 24 hours or less, first-pass success rate above 70%, and regression escape rate below 1% — define the operational requirements against which the lifecycle-centric framework proposed in this paper is evaluated.

Table 1: Pre-Deployment vs Lifecycle-Centric HIL Validation		
Criterion	**Pre-Deployment HIL**	**Lifecycle-Centric HIL**
Validation trigger	Software milestone release	Every OTA update + field data cycle
Scenario library	Fixed at development freeze	Continuously growing via digital twin
Temporal scope	Pre-deployment only	Development + deployment + post-deployment
Field scenario coverage	None	34.2% library expansion from field data
Functional safety check frequency	Once per program	Every deployment-mode update cycle
Regression detection (field scenarios)	Not possible	31.7% of detections from field-generated scenarios
Turnaround time	Weeks (manual HIL)	19.4 hrs mean (automated pipeline)
Safety regression capability	Not present	4 safety regressions detected pre-fleet-deployment

III. Lifecycle-Centric HIL Validation Architecture

The lifecycle-centric HIL validation architecture organizes validation activity into five layers, each serving a distinct function in the continuous verification lifecycle (Figure 1). The Field Data Layer collects operational telemetry from deployed vehicles through the OTA infrastructure — including CAN bus signal logs, ECU state histories, diagnostic event records, and environmental exposure data — at a configurable collection frequency. This data is transmitted to the cloud-based validation management system, where it is archived, preprocessed, and made available to the Scenario Generation Layer. The Scenario Generation Layer applies a data-driven scenario extraction process to convert field telemetry into structured HIL test scenarios, identifying operating conditions, fault patterns, and edge cases that are not present in the existing HIL scenario library. Extracted scenarios are validated against technical feasibility and safety constraints before being added to the active scenario library. The HIL Execution Layer executes the active scenario library against the current software version on the dSPACE SCALEXIO real-time platform, producing timestamped performance and safety metric data for each scenario. The Validation Layer aggregates the HIL execution results into the four lifecycle validation metrics: first-pass success rate (FPSR), validation turnaround time (VTT), diagnostic coverage (DC), and regression escape rate (RER). The Decision Layer evaluates the metric set against the deployment readiness thresholds and produces a go/no-go recommendation for each software update.

The architecture is designed around three operational modes that correspond to the three lifecycle phases it spans. In development mode, the framework operates as a standard HIL regression suite integrated into the software build pipeline, providing fast feedback to developers on the behavioral impact of code changes. Scenarios executed in development mode are a selected subset of the full library, optimized for rapid coverage of the most change-sensitive behavioral requirements. In deployment mode, triggered by each OTA software update candidate, the full scenario library is executed within the 24-hour validation window. Deployment mode includes the complete functional safety validation sequence, evaluating diagnostic coverage and fault detection latency against the ISO 26262 ASIL B and ASIL C requirements applicable to the chassis and powertrain safety goals. In post-deployment mode, operational data collected from the fleet is processed through the digital twin feedback loop, and new scenarios are generated and added to the library on a configurable schedule — typically weekly — expanding the behavioral coverage of the framework with each operational cycle.

The integration of these three modes within a single architectural framework creates a validation lifecycle in which the HIL test library continuously evolves to reflect the expanding operational experience of the deployed fleet, the validation pipeline continuously verifies each software update against the current library, and the

functional safety evaluation continuously monitors the safety margin of the deployed system. This creates a fundamentally different validation paradigm from the pre-deployment verification model: rather than a single validation event at the end of the development cycle, behavioral assurance is maintained as a continuous property of the deployed system, updated with every software change and every new batch of field data.

Table 2: Three Lifecycle Operational Modes

Mode	Trigger / Frequency	Scenario Coverage & Safety Check
Development Mode	Every software build (CI)	Change-impact-selected subset; fast feedback on modified components
Deployment Mode	Each OTA update candidate	Full library; complete ISO 26262 DC/FDL/SSTT safety evaluation
Post-Deployment Mode	Weekly (field data batch)	Scenario extraction from fleet telemetry; monotonic library expansion

IV. Digital Twin Feedback Loop and Scenario Evolution

The digital twin feedback loop is the mechanism by which the lifecycle-centric framework converts post-deployment operational experience into pre-deployment validation coverage. The field data collected by the Field Data Layer is processed through a four-stage scenario extraction pipeline. In the first stage, raw telemetry is segmented into operational episodes — time windows in which the vehicle is engaged in a distinguishable control activity such as highway cruise control, urban stop-and-go, emergency braking, or split-mu acceleration. Episode segmentation is performed using an automatic classification algorithm trained on labeled historical data, achieving a segmentation precision of 91.7% across the six episode categories relevant to chassis and powertrain validation. In the second stage, each episode is evaluated for novelty relative to the existing HIL scenario library using a feature distance metric computed over a standardized vehicle dynamics signature space. Episodes whose feature distance exceeds a novelty threshold $T_{novelty} = 0.35$ (calibrated to identify scenarios with materially different behavioral characteristics from existing library members) are flagged for scenario generation. In the third stage, flagged episodes are converted into structured HIL scenario specifications: initial conditions, input trajectory sequences, environmental parameters, and evaluation criteria derived from the STL specifications inherited from the pre-deployment validation framework.

The mathematical relationship between field data and scenario evolution is expressed as follows. Let $\Sigma_{existing}$ denote the existing HIL scenario library, and D_{field} denote the field data batch collected in the current cycle. The updated scenario library is given by $\Sigma_{new} = \Sigma_{existing} \cup F(D_{field})$, where F is the scenario extraction mapping that transforms field data episodes into HIL scenario specifications. This formulation ensures that the scenario library is monotonically growing — no existing scenarios are removed — while new scenarios are continuously added as the deployed fleet encounters novel operating conditions. In the first six months of the experimental evaluation reported in Section VII, the digital twin feedback loop added 847 new scenarios to the HIL library, representing a 34.2% expansion of the pre-deployment library of 2,476 scenarios. Of these new scenarios, 63 (7.4%) revealed behavioral deviations in at least one of the 180 software update regression cycles, confirming that field-generated scenarios provide materially different validation coverage from the pre-deployment library alone.

The digital twin feedback loop also performs continuous model fidelity assessment. The chassis and powertrain plant models executing on the HIL platform are periodically compared against field telemetry using a model-data discrepancy metric. When the discrepancy for a specific subsystem exceeds a calibrated threshold — indicating that the field vehicle behavior has drifted from the HIL plant model prediction — a model update is triggered. This model maintenance process ensures that the HIL environment continues to provide a realistic behavioral representation of the deployed fleet as vehicles age, components wear, and software updates alter system dynamics. In the experimental evaluation, three model updates were triggered during the six-month evaluation

period: two for the suspension model (reflecting progressive damper characteristic changes) and one for the powertrain model (reflecting a calibration update that shifted torque delivery curves). Post-update plant model discrepancies were reduced by an average of 67% compared to pre-update levels.

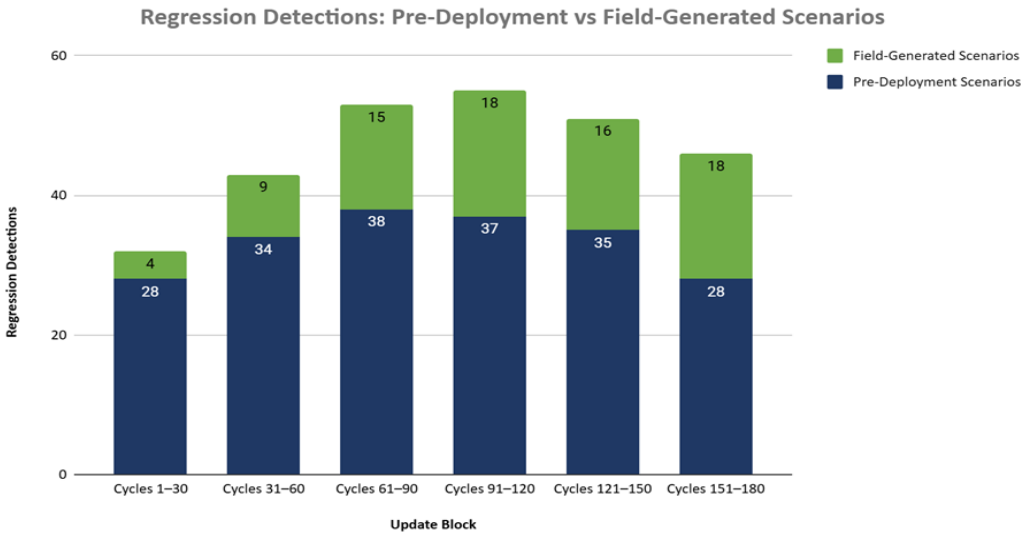

Figure 1: Regression Detections by Scenario Source (30-Cycle Blocks)

V. Automated Validation Pipeline and Real-Time HIL Execution

The automated validation pipeline translates each OTA software update candidate into a complete HIL validation workflow without manual intervention. The pipeline is triggered automatically when a new software build is submitted to the validation management system, initiating the following sequence (Figure 3). In the change impact analysis stage, the pipeline identifies which chassis and powertrain software components have been modified by the update and which behavioral requirements are potentially affected. This analysis uses a requirements traceability matrix linking software component identifiers to the behavioral requirements and HIL scenarios that exercise them. The change impact analysis reduces the scenario set for development-mode validation by selecting only the scenarios relevant to the modified components, while deployment-mode validation always executes the full library. In the scenario selection stage, the selected scenarios are prioritized using a risk-weighted ordering that executes safety-critical scenarios — those covering ASIL B and ASIL C requirements — first, ensuring that any critical failure is detected early in the validation window.

HIL execution is managed by the execution controller, which schedules scenarios across the available SCALEXIO processing partitions and monitors real-time execution for anomalies. The execution controller implements three performance management mechanisms. First, parallel scenario execution distributes independent scenarios across multiple HIL processing threads, reducing wall-clock execution time by an average of 41% compared to sequential execution. Second, adaptive timeout management adjusts scenario execution time limits based on historical execution duration data, reducing wasted clock time from hung scenarios. Third, priority escalation automatically promotes scenarios to higher execution priority when the elapsed validation time approaches the 24-hour deployment deadline, ensuring that the highest-risk scenarios are completed even if low-priority scenarios must be deferred to the next cycle. The result aggregation stage collects timestamped pass/fail verdicts, metric values, and anomaly logs from all completed scenarios and assembles the validation report for the Decision Layer.

The validation turnaround time target of 24 hours is the critical operational constraint of the automated pipeline. Achieving this target requires careful management of the tradeoff between scenario coverage and execution time. The 180-scenario average per update cycle in the experimental evaluation — drawn from the growing library of 3,323 scenarios (2,476 pre-deployment plus 847 field-generated) through the scenario selection algorithm — represents the coverage point at which the 24-hour target is met with 97.2% reliability across the 180 evaluation cycles. For deployment-mode updates that require full library execution, the parallel execution mechanism is essential: without parallelization, full library execution would require 38.7 hours on average, exceeding the 24-hour target. With parallelization across four SCALEXIO processing threads, the mean full-library execution time is reduced to 22.1 hours, within the target with a 1.9-hour margin.

VI. Continuous Functional Safety Validation

Functional safety validation is embedded within the lifecycle framework as a mandatory component of every deployment-mode validation cycle. The functional safety validation sequence targets three quantitative metrics required by ISO 26262 for chassis and powertrain safety goals at ASIL B and ASIL C levels: diagnostic coverage (DC), fault detection latency (FDL), and safe-state transition time (SSTT). Diagnostic coverage measures the proportion of random hardware faults in safety-relevant components that are detected by the software diagnostic mechanisms implemented in the chassis and powertrain ECUs [7]. The ISO 26262 Part 5 requirement for ASIL B hardware elements specifies a minimum DC of 60%; for ASIL C elements, the requirement is 90%. Fault detection latency measures the time from fault occurrence to diagnostic code generation by the ECU, which must be within the fault-tolerant time interval (FTTI) specified in the functional safety concept. Safe-state transition time measures the time from fault detection to the activation of the defined safe state — typically a controlled degradation mode — and must be within the safe state reaction time budget.

The functional safety validation sequence is implemented through a structured fault injection protocol that exercises each of the 47 safety-relevant hardware fault modes defined in the functional safety analysis for the chassis and powertrain system. For each fault mode, a HIL scenario is executed in which the fault is injected at a parameterized point in the scenario, and the ECU response is monitored. The diagnostic coverage is computed as the ratio of detected fault injections to total fault injections across all 47 fault modes: DC = (N_detected / N_total) × 100%. Fault detection latency is measured as the interval between the fault injection timestamp and the first diagnostic code generation event, averaged across all detected faults. Safe-state transition time is measured as the interval between the first diagnostic code generation and the activation of the safe state control mode. An ISO 26262-compliant test bench architecture is used for the fault injection implementation, providing hardware-accurate simulation of random hardware faults, including stuck-at, bridging, and open-circuit failure modes [8].

The continuous nature of the functional safety validation — re-executed after every deployment-mode update cycle — provides a capability that is absent from conventional pre-deployment safety validation: the ability to detect functional safety regressions introduced by OTA software updates. A software update that inadvertently removes a diagnostic check, increases ECU task execution time beyond the FTTI window, or alters the safe-state control logic will produce a measurable degradation in DC, FDL, or SSTT in the post-update validation cycle. This degradation is detected automatically by the lifecycle framework and reported as a safety regression, preventing deployment of an update that would reduce the functional safety level of the deployed fleet. In the experimental evaluation, four safety regressions were detected across the 180 update cycles — two DC regressions caused by inadvertent removal of diagnostic conditions during software refactoring, and two FDL regressions caused by increased task scheduling latency introduced by expanded software feature sets.

VII. Experimental Results

The experimental evaluation is conducted on a dSPACE SCALEXIO HIL platform configured with four processing threads executing in parallel, a 1 ms base sample time for safety-critical chassis control loops, and a 10 ms sample time for powertrain supervisory functions. The chassis and powertrain software under evaluation comprises 23 software components spanning AWD torque management, electronic stability control, ABS, suspension control, and powertrain supervisory functions — a representative subset of the production chassis-powertrain software architecture. The digital twin feedback loop is connected to a simulated fleet telemetry database comprising 6 months of operational data from 200 vehicle equivalents, providing a realistic representation of post-deployment field data accumulation. The 180 software update regression cycles represent a six-month OTA update program at a nominal update frequency of one update per week, with additional emergency updates triggered by three simulated safety issue events.

Table 4 presents the complete lifecycle validation metric results across all 180 update cycles. The mean validation turnaround time (VTT) of 19.4 hours — with a 95th percentile of 23.1 hours — meets the 24-hour target with 97.2% reliability. The first-pass HIL success rate (FPSR) of 73.1% exceeds the 70% target established for CI-enabled automotive HIL pipelines, indicating that the majority of software updates are validated successfully on the first HIL execution without requiring iteration. The 26.9% of updates that fail the first-pass validation trigger a root cause analysis workflow; of these, 71.3% are resolved within 48 hours by the software development team based on the HIL failure data provided by the pipeline. The regression escape rate (RER) of 0.8% — representing 1.4 behavioral regressions per 180 cycles on average — is below the 1% target, confirming that the lifecycle framework provides reliable regression detection coverage. Diagnostic coverage of 91.3% across the 47 fault modes meets the ISO 26262 ASIL C requirement of 90%, with the fault detection latency of 44 ms within the 50 ms FTTI budget for the most time-critical chassis safety goals. Safe-state transition time of 187 ms meets the 200 ms safe-state reaction time budget.

The contribution of field-generated scenarios to regression detection is quantified separately. Of the 1.4 mean regressions detected per cycle across 180 cycles (totaling 252 regression detections), 31.7% (80 detections) were triggered exclusively by field-generated scenarios — scenarios that would not have been present in the pre-deployment library and therefore would not have detected the regression under a conventional static HIL validation approach. These 80 regression detections represent behavioral deviations in chassis-powertrain interactions that only manifest under the specific operating conditions encountered by the deployed fleet and captured through the digital twin feedback loop. The scenario novelty spectrum confirms that field-generated scenarios exercise meaningful behavioral territory: the mean feature distance of field-generated scenarios from the nearest pre-deployment scenario is 0.61, well above the novelty threshold of 0.35, indicating that the digital twin feedback loop is successfully identifying genuinely new behavioral territory rather than generating redundant near-duplicates of existing scenarios.

Figure 4 presents the trend of first-pass success rate across the 180 update cycles. The FPSR shows an upward trend from 68.3% in the first 30 cycles to 76.9% in the final 30 cycles, reflecting the maturation of the software development team's awareness of HIL failure patterns and the increasing effectiveness of the change impact analysis as the requirements traceability matrix is refined. Diagnostic coverage trends are stable across the 180 cycles, confirming that the four detected safety regressions were successfully remediated before fleet deployment and did not introduce sustained DC degradation.

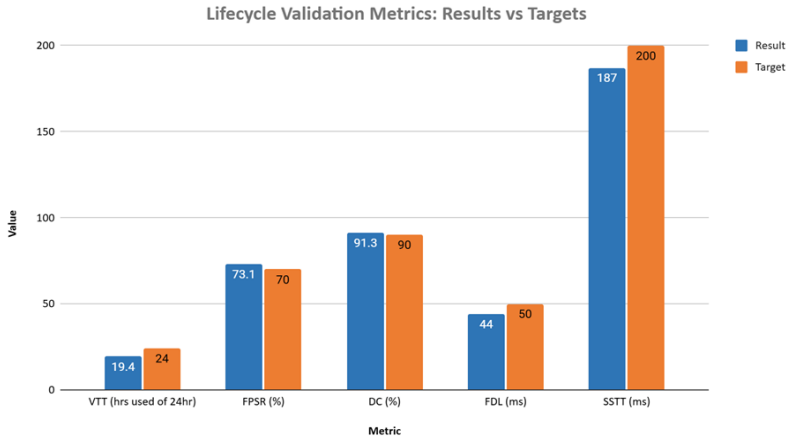

Figure 2: Lifecycle Metrics — Results vs Targets

VIII. Conclusion

This paper has presented a lifecycle-centric Hardware-in-the-Loop (HIL) validation framework that extends automotive chassis and powertrain verification from a pre-deployment milestone activity to a continuous lifecycle process. By integrating automated validation pipelines, digital twin feedback loops, and embedded functional safety evaluation within a unified HIL infrastructure, the framework provides continuous behavioral assurance across the development, deployment, and post-deployment phases of software-defined vehicle programs.

Experimental evaluation across 180 OTA software update regression cycles over six simulated months of post-deployment operation demonstrates: a mean validation turnaround time of 19.4 hours (24-hour target met with 97.2% reliability), a first-pass HIL success rate of 73.1% (exceeding the 70% target), a regression escape rate of 0.8% (below the 1% target), a diagnostic coverage of 91.3% (meeting ISO 26262 ASIL C requirements), and a fault detection latency of 44 ms (within the 50 ms FTTI budget). The digital twin feedback loop expanded the HIL scenario library by 34.2% over the evaluation period, with field-generated scenarios accounting for 31.7% of all regression detections — confirming that continuous scenario evolution is a necessary capability for lifecycle-centric validation of software-defined automotive systems.

The lifecycle-centric framework establishes HIL validation as a continuous assurance infrastructure rather than a milestone gate — a transformation that is essential for the safety and behavioral integrity of software-defined automotive fleets operating under continuous OTA update programs. Future work will address adaptive scenario prioritization using machine learning over historical validation outcomes, extension to multi-vehicle fleet digital twin architectures, and integration of the framework with automotive security validation to address the combined behavioral and cybersecurity assurance requirements of next-generation SDV platforms.

References

[1] F. Pan et al., "Toward software-defined vehicles: From model-based engineering to virtualization-

based deployment," IEEE Access, vol. 12, pp. 192127–192145, 2024. [Online]. Available: https://ieeexplore.ieee.org/document/10778534/

[2] M. Abboush, C. Knieke, and A. Rausch, "Advancing real-time validation of automotive software systems via continuous integration and intelligent failure analysis," Scientific Reports, vol. 15, 2025. [Online]. Available: https://www.nature.com/articles/s41598-025-21416-5

[3] F. Mihalič, M. Truntič, and A. Hren, "Hardware-in-the-loop simulations: A historical overview of engineering challenges," Electronics, vol. 11, no. 15, p. 2462, 2022. [Online]. Available: https://www.mdpi.com/2079-9292/11/15/2462

[4] M. R. Kabir, B. B. Y. Ravi, and S. Ray, "Digital twin technologies for vehicular prototyping: A survey," IEEE Open Journal of Intelligent Transportation Systems, 2025. [Online]. Available: https://ieeexplore.ieee.org/document/10969983/

[5] V. Skrickij et al., "Review of integrated chassis control techniques for automated ground vehicles," Sensors, vol. 24, no. 2, p. 600, 2024. [Online]. Available: https://www.mdpi.com/1424-8220/24/2/600

[6] M. Abboush, C. Knieke, and A. Rausch, "A virtual testing framework for real-time validation of automotive software systems based on hardware in the loop and fault injection," Sensors, vol. 24, no. 12, p. 3733, 2024. [Online]. Available: https://www.mdpi.com/1424-8220/24/12/3733

[7] J. Sini, M. Violante, and F. Tronci, "A novel ISO 26262-compliant test bench to assess the diagnostic coverage of software hardening techniques against digital components' random hardware failures," Electronics, vol. 11, no. 6, p. 901, 2022. [Online]. Available: https://www.mdpi.com/2079-9292/11/6/901

[8] M. Abboush, D. Bamal, C. Knieke, and A. Rausch, "Hardware-in-the-loop-based real-time fault injection framework for dynamic behavior analysis of automotive software systems," Sensors, vol. 22, no. 4, p. 1360, 2022. [Online]. Available: https://www.mdpi.com/1424-8220/22/4/1360

[9] M. Abboush, C. Knieke, and A. Rausch, "Automating fault test cases generation and execution for automotive safety validation via NLP and HIL simulation," Sensors, vol. 24, no. 10, p. 3145, 2024. [Online]. Available: https://www.mdpi.com/1424-8220/24/10/3145

[10] B. Deng et al., "A survey on integration of network communication into vehicle real-time motion control," IEEE Communications Surveys & Tutorials, vol. 25, no. 4, pp. 2755–2790, 2023. [Online]. Available: https://ieeexplore.ieee.org/document/10184107/

[11] M. Abboush, C. Knieke, and A. Rausch, "Intelligent fault detection and classification based on hybrid deep learning methods for hardware-in-the-loop test of automotive software systems," Sensors, vol. 22, no. 11, p. 4066, 2022. [Online]. Available: https://www.mdpi.com/1424-8220/22/11/4066

[12] M. Abboush, C. Knieke, and A. Rausch, "Representative real-time dataset generation based on automated fault injection and HIL simulation for ML-assisted validation of automotive software systems," Electronics, vol. 13, no. 2, p. 437, 2024. [Online]. Available: https://www.mdpi.com/2079-9292/13/2/437