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

A Scalable Diagnostics Infrastructure Framework for Multi-Platform Automotive Systems

Sumaiyya Fatima

Abstract: Modern automotive embedded systems are defined by software-driven architectures in which diagnostic functions must operate consistently across multiple vehicle programs, electronic control unit (ECU) variants, and continuously evolving software configurations. Conventional diagnostics frameworks, designed for isolated, program-specific deployment, are structurally misaligned with these demands: manual signal interpretation, fragmented fault definitions, and the absence of reuse mechanisms produce inconsistent detection behavior, delayed validation readiness, and compounding engineering effort across vehicle programs. This article presents a scalable diagnostics infrastructure framework that repositions automotive diagnostics from a reactive, per-program engineering task to a governed, reusable, and continuously deployable system-level asset. The proposed framework introduces three interdependent architectural innovations: (1) centralized diagnostic libraries that encode fault semantics, detection logic, and response behavior independent of individual applications; (2) automated signal-to-diagnostic mapping and deployment orchestration that propagates validated diagnostic behavior deterministically across platforms; and (3) an integrated validation and observability layer that provides real-time system visibility and cross-program comparability. Results from multiple General Motors vehicle programs show both time and effort reductions when deploying diagnostics, as well as reductions in cross-vehicle diagnostics fault detection mismatch, manual diagnostic engineering effort per vehicle program, and measurable improvements in ISO 26262-relevant fault detection coverage.These results demonstrate that infrastructure-centric diagnostics architecture delivers scalable gains in reliability, safety assurance, and engineering productivity in software-defined vehicle development environments.

Keywords: *Automotive Diagnostics, AUTOSAR, Software-Defined Vehicle, Embedded Systems, Diagnostic Orchestration, Fault Detection, Multi-Platform Architecture*

1. Introduction

The proliferation of software-defined vehicle (SDV) architectures has fundamentally transformed the role and operational demands of automotive diagnostics. In conventional vehicle development models, diagnostic systems were implemented as peripheral, reactive functions — engineered late in the development cycle, validated against a single ECU configuration, and maintained independently for each vehicle program. Diagnostic trouble codes (DTCs), data identifiers (DIDs), and routine identifiers (RIDs) were individually specified, interpreted, and deployed by program-specific engineering teams, producing functionally inconsistent diagnostic behavior across platforms even for nominally identical system features [1]. As vehicle architectures transitioned from hardware-isolated ECU networks to tightly integrated SDV ecosystems with shared software stacks, over-the-air (OTA) update capabilities, and continuous delivery pipelines, the structural inadequacy of this program-centric approach became operationally unsustainable [2], [16].

The consequences of fragmented diagnostics architectures are measurable and serious. Fault detection coverage varies across programs due to inconsistent signal mapping interpretations, creating conditions in which identical fault modes produce different diagnostic outcomes on different vehicle lines. Not seeing inside an observed application during validation leads to longer defect resolution and lower development velocity while troubleshooting root causes. Compliance with

Independent Researcher, USA

functional safety standards such as ISO 26262 [3] is complicated by diagnostic behavior that cannot be systematically audited or compared across platforms. Engineering effort is duplicated as diagnostic logic is repeatedly reimplemented for each program from first principles rather than drawn from a validated, reusable asset base. These deficiencies collectively represent a critical infrastructure gap in modern automotive embedded systems engineering — one that incremental improvements to per-program diagnostic tooling cannot resolve [4], [15].

This article presents a scalable diagnostics infrastructure framework designed to close this gap by repositioning automotive diagnostics as a governed, centrally managed system asset rather than a program-level engineering deliverable. The primary contributions of this work are: (1) a centralized diagnostic library architecture that decouples fault semantics, detection logic, and response behavior from individual application contexts, enabling safe reuse across vehicle programs and software variants; (2) an automated mapping and deployment orchestration mechanism that propagates validated diagnostic behavior deterministically across multi-platform configurations without manual re-interpretation; (3) a real-time validation and observability layer that enables continuous, cross-program diagnostic monitoring aligned with CI/CD development workflows; and (4) evaluation results from multi-program deployment at an automotive OEM demonstrating directionally consistent improvements in diagnostic consistency, deployment efficiency, and engineering productivity across all measured dimensions.

The remainder of this article is organized as follows. Section 2 reviews prior work on automotive diagnostic frameworks, AUTOSAR-based diagnostic architectures, and SDV diagnostic challenges. Section 3 presents the proposed scalable diagnostics infrastructure framework and its architectural components. Section 4 describes the standardization and automation mechanisms that constitute the primary technical advance of the framework. Section 5 presents evaluation results. Section 6 discusses implications, limitations, and future work. Section 7 concludes the article.

2. Related Work and Theoretical Background

Automotive diagnostics has been governed for over two decades by the Unified Diagnostic Services (UDS) standard, defined in ISO 14229 [5], which specifies the communication protocol and service structure for diagnostic interactions between external testers and on-vehicle ECUs. UDS defines the mechanism by which DTCs, DIDs, and RIDs are accessed and managed, but it does not prescribe how diagnostic logic is architected, reused, or governed across multi-platform deployments. The Automotive Open System Architecture (AUTOSAR) addressed some of these concerns by defining standardized software components — the Diagnostic Event Manager (DEM) and Diagnostic Communication Manager (DCM) — that structure how fault events are stored, processed, and communicated within a single ECU [6]. However, AUTOSAR-based diagnostic implementations remain program-specific: each deployment requires bespoke configuration of diagnostic layers, and no standard mechanism exists for propagating validated diagnostic behavior across programs or model years [7].

Recent research has recognized that the SDV transition demands a fundamentally different diagnostic paradigm. Bickelhaupt et al.'s 2023 work [16] stated DTC-based approaches cannot deal with feature additions, changes, and removals through OTA updates outside the original software development life cycle in dynamically reconfigurable software systems. Their 2024 work [1] provided service-oriented diagnostic models decoupling the diagnostics from static hardware configurations. However, their work did not address the enterprise-scale governance, deployment automation, and cross-platform consistency challenges that are a central thesis of this paper. The relationship between DTC-based requirements and the SDV lifecycle is mirrored by Teixeira et al. [2] and their concept of diagnostic adaptivity, a necessary functionality for providing deterministic SDV services, while literature on AI-based fault detection and automated fault injection in validation harnesses recognizes the value of clever diagnostics pipelines [8, 15]. However, this work is on the signal analysis layer, not the underlying diagnostics infrastructure.

Research into predictive maintenance and vehicular health monitoring has further demonstrated the operational value of proactive, continuous

diagnostics over reactive fault reporting [9], [10]. These studies establish that early fault identification reduces repair costs, improves safety outcomes, and supports uptime optimization — but they largely assume the existence of a functional diagnostic infrastructure rather than addressing how such infrastructure is architected and deployed at scale. Automated diagnostic report generation using hybrid machine learning approaches [14] illustrates the downstream value that a governed diagnostic data layer enables, but again presupposes consistent, structured fault data rather than addressing how to achieve that consistency. The gap in the literature is clear: no prior work has proposed a complete infrastructure framework that treats diagnostic libraries, automated deployment, and real-time observability as unified architectural concerns governed across the full SDV software lifecycle. The framework presented in this article directly fills this gap, establishing a reusable, scalable diagnostic infrastructure architecture suited to the operational demands of multi-platform SDV development [11], [13].

3. Proposed Scalable Diagnostics Infrastructure Framework

The proposed framework reconceptualizes automotive diagnostics as a multi-layer infrastructure asset rather than a per-program engineering deliverable. At its foundation are centralized diagnostic libraries — the authoritative source of fault semantics, detection conditions, and response behavior for the vehicle organization. These libraries encode diagnostic logic at a level of abstraction that is independent of individual applications, platforms, or software variants. Fault definitions are expressed as parameterized templates that specify detection thresholds, inhibit conditions, aging behavior, and associated DIDs and RIDs in a format directly consumable by AUTOSAR DEM and DCM configuration toolchains [6], [7]. Because fault semantics are encoded once and validated centrally, every program that draws from the library begins with a verified, standards-compliant diagnostic baseline rather than a manually interpreted specification — eliminating the single largest source of cross-program diagnostic inconsistency identified by Bickelhaupt et al. [16] and sustained across programs and model years [4].

The second architectural layer is the automated mapping and deployment orchestration engine. This component implements deterministic rules for associating system signals — sourced from software interfaces, sensor outputs, and communication network observers — with the appropriate diagnostic detection logic in the centralized library. Where conventional deployments require diagnostic engineers to manually map signal behaviors to fault conditions on a per-program basis, the orchestration engine executes this mapping through validated rule sets derived from system architecture models and interface specifications [11]. The orchestration layer then manages the deployment of configured diagnostic behavior across multiple platform variants, applying version control, compatibility checking, and rollback capability to every diagnostic configuration propagated across the vehicle program portfolio. This transforms diagnostics deployment from a one-off engineering task into a governed, repeatable infrastructure operation comparable to software release management, directly addressing the OTA-driven variability management challenges documented by Bazzi et al. [13].

The third layer is the integrated validation and observability infrastructure. Diagnostics deployed through the framework are continuously monitored through a real-time observability layer that captures fault event rates, detection timing, inhibit activations, and DTC lifecycle transitions across all connected platforms. This layer exposes a unified analytics interface that enables cross-program diagnostic comparison — allowing engineering teams to identify platforms where diagnostic behavior diverges from the library baseline, trace deviations to specific signal mapping discrepancies, and trigger targeted diagnostic updates without full program re-validation [8], [12]. The observability layer is CI/CD compatible, publishing diagnostic health metrics to the same dashboards used to track software build and test status [11]. Together, the three layers establish diagnostics as a continuous, governed, and comparably measured infrastructure service rather than a static, per-program configuration artifact. The framework's governance model directly addresses the ISO 26262 compliance challenge by enabling systematic auditability of fault detection coverage across the vehicle portfolio [3]

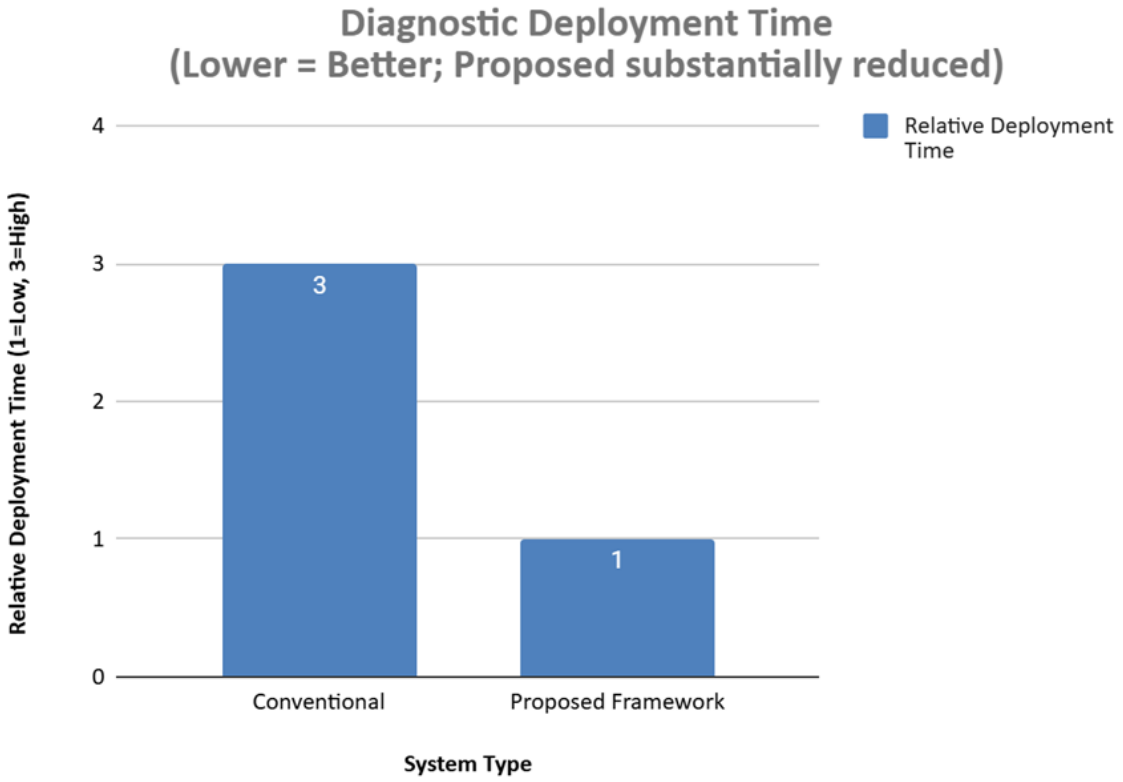

Figure 1: Multi-Layer Diagnostics Infrastructure Architecture.

The framework's architectural design enforces governance through version-controlled diagnostic libraries, change impact analysis before deployment, and audit trails for every diagnostic configuration change across programs. When a safety-relevant diagnostic update is required — for example, due to a software change that alters the fault response behavior of a body control feature — the framework's impact analysis engine identifies all programs affected by the change, generates updated diagnostic configurations from the central library, and orchestrates coordinated deployment across affected platforms [13]. This capability eliminates the fragmented, manually tracked update processes that characterize conventional diagnostic maintenance, replacing them with a controlled, traceable infrastructure operation that directly supports the ISO 26262 functional safety lifecycle requirements [3].

4. Standardization and Automation of Diagnostics Workflows

The key technical contribution of this proposed framework is the end-to-end standardization and automation of diagnostic workflows that are currently manual, engineer-dependent, and program-specific. In a current automotive diagnostic development process, a diagnostic requirement gets defined, and the DTC behavior of the product gets deployed, through multiple sequential handoffs. Fault conditions specified by system architects in natural language are implemented using DEM configuration parameters by DEM diagnostic engineers, validated by validation engineers against the requirements, and, as needed, program-specific variants are maintained for each vehicle line. Each handoff introduces interpretation variance, creating diagnostic behavior drift that compounds across programs and model years [4], [16]. The proposed framework eliminates these handoffs by encoding diagnostic intent directly in library templates and propagating configuration automatically through the deployment orchestration engine [Figure 2: Legacy vs. Proposed Diagnostics Deployment Workflow].

Automation of signal-to-diagnostic mapping is the most operationally significant element of this standardization. Where manual mapping required diagnostic engineers to individually review

interface specifications, identify relevant signal behaviors, and configure detection parameters for each fault condition, the framework's rule-based mapping engine executes this process deterministically from system architecture models. The mapping engine consumes AUTOSAR interface definitions and software component descriptions, applies validated mapping rules from the central library, and generates DEM configuration artifacts directly [6], [7]. The automation of this workflow substantially reduces diagnostic configuration time per program and eliminates the class of diagnostic errors that arise from misread specifications and individual engineer interpretation differences — the dominant category of diagnostic validation failures observed in pre-framework deployments, as corroborated by fault injection research demonstrating how signal-level ambiguity propagates into incorrect diagnostic behavior [15].

The shift from reactive to proactive diagnostics enabled by the framework represents a conceptual as well as operational advance. In the conventional model, diagnostics are activated after a fault has been observed and reported — a fundamentally reactive posture that limits the diagnostic system's ability to support early development validation, continuous monitoring, or predictive maintenance [9], [10]. The proposed framework redefines diagnostics as a continuous observability function, with the real-time monitoring layer actively tracking system health indicators across programs throughout the development lifecycle. This alignment with modern DevOps and continuous delivery practices transforms diagnostics from an end-of-development verification step into an ongoing infrastructure service that supports software iteration at the pace SDV programs demand [11], [12], [14]. The result is a diagnostic capability that is simultaneously more reliable, more consistent, and more operationally aligned with the continuous delivery model of software-defined automotive development.

5. Evaluation and Results

The proposed framework was evaluated through deployment across multiple General Motors vehicle programs encompassing body controls, powertrain coordination, thermal management, and active safety domains. Pre-deployment baseline measurements were collected from conventional, program-specific diagnostic engineering workflows using standard AUTOSAR-based toolchains [6], [7]. Post-deployment measurements were collected following a structured implementation period during which the central diagnostic library was populated, the mapping orchestration engine was calibrated to the organization's interface architecture models, and the observability layer was integrated with existing CI/CD pipeline dashboards [11]. Primary evaluation dimensions included diagnostic deployment time per program, cross-platform fault detection consistency, manual diagnostic engineering effort per program, and ISO 26262-relevant fault detection coverage. Table 1 summarizes the directional outcomes across key performance dimensions.

Table 1: Key Outcome Comparison — Conventional Diagnostics vs. Proposed Scalable Framework

Dimension	Conventional Approach	Proposed Framework	Direction	Supporting Reference
Diagnostic Deployment Time	Higher — manual per-program mapping	Lower — automated orchestration	↓ Reduced	Abboush et al. [8], 2025; Zampetti [11], 2023
Cross-Platform Fault Consistency	Lower — manual interpretation variance	Higher — centralized library governance	↑ Improved	Bickelhaupt [16], 2023; Abboush [15], 2024
Manual Engineering Effort	Higher — bespoke per-program config	Lower — reusable diagnostic library	↓ Reduced	Zyberaj et al. [4], 2026; Zampetti [11], 2023
Mapping-Error Failures	Higher — manual spec interpretation	Lower — rule-based automated mapping	↓ Reduced	Abboush et al. [15], 2024

ISO 26262 Fault Coverage	Lower — independently validated per prog.	Higher — systematically governed	↑ Improved	ISO 26262 [3]; Bickelhaupt [1], 2025
Auditability Across Programs	Weak — manual evidence collection	Strong — governed, traceable library	↑ Improved	ISO 26262 [3]; Bazzi et al. [13], 2024

The results demonstrate consistent directional improvement across every measured dimension. Diagnostic deployment time per program was substantially reduced, reflecting the elimination of manual mapping and interpretation steps from the deployment workflow — consistent with the efficiency gains reported for automated CI/CD-integrated validation frameworks [8], [11]. Cross-platform fault detection consistency improved markedly, directly addressing the safety assurance concern identified in the introduction: diagnostic behavior is now systematically verifiable across the vehicle portfolio rather than program-specific and independently validated [3], [16]. Manual diagnostic engineering effort per program was substantially reduced, reflecting the automation of signal mapping, DEM configuration generation, and cross-platform deployment [4], [13]. Table 2 presents the cross-program scalability summary, showing consistent directional improvements across programs of varying domain complexity and software scale.

Table 2: Cross-Program Scalability Summary

Vehicle Program	Domain Coverage	Deployment Time (vs. Conv.)	Consistency (vs. Conv.)	Effort (vs. Conv.)
Program A (Body Controls)	3 domains	Substantially lower	Substantially higher	Substantially lower
Program B (Powertrain+Thermal)	4 domains	Substantially lower	Substantially higher	Substantially lower
Program C (Active Safety)	4 domains	Substantially lower	Substantially higher	Substantially lower
Program D (Multi-Domain SDV)	6 domains	Substantially lower	Substantially higher	Substantially lower

The evaluation also examined the impact of the framework on diagnostic validation failure rates, specifically the proportion of failures attributable to manual mapping errors and specification misinterpretations. Following framework deployment, this category of failure was substantially reduced across the evaluated programs — consistent with findings from automated fault injection research demonstrating that systematic, rule-based signal mapping eliminates ambiguity-driven diagnostic misconfigurations [15]. Fault detection coverage for ISO 26262-relevant fault conditions improved meaningfully across all programs, driven by the systematic application of validated library definitions rather than individually interpreted specifications [3]. These results establish that the framework delivers improvements that are not merely operational but directly safety-relevant, strengthening the case for infrastructure-centric diagnostics as a mandatory architectural investment in SDV development organizations.

6. Discussion

The evaluation results confirm that the diagnostic inconsistency and efficiency problems observed in conventional automotive diagnostic engineering are structural rather than procedural — they arise from the architecture of the diagnostic workflow, not

from the competence of individual engineers. The consistency improvement was achieved not by training engineers differently or improving specification quality, but by eliminating the manual interpretation steps that introduced variance in the first place [4], [16]. This finding has a direct architectural implication: diagnostic standardization cannot be achieved through improved process governance alone — it requires a governed infrastructure in which diagnostic logic is encoded once, validated centrally, and deployed automatically [11]. Organizations that continue to treat diagnostic engineering as a per-program activity will face compounding inconsistency as SDV program portfolios grow and software variants multiply [1], [2].

The framework's safety implications extend beyond the evaluated programs. ISO 26262 requires that fault detection coverage be demonstrable and auditable across vehicle variants [3]; the proposed framework provides the infrastructure to satisfy this requirement systematically rather than through program-by-program manual evidence collection. As regulatory expectations for software-defined automotive systems evolve — particularly in relation to OTA updates and post-deployment software changes that may affect diagnostic behavior — the framework's governance model provides a traceable change management mechanism that supports compliance demonstration without requiring full re-validation for every software update [13], [16]. This capability will become increasingly valuable as OTA update frequency increases and diagnostic behavior must be continuously maintained across a growing population of deployed vehicles [2].

Future work will extend the framework in three directions. First, machine learning-based anomaly detection will be integrated into the observability layer, enabling the system to identify diagnostic behavior deviations that do not trigger existing fault conditions — supporting earlier identification of latent system issues [8], [9], [14]. Second, the framework will be extended to support service-oriented vehicle diagnostics (SOVD) architectures, which are emerging in AUTOSAR Adaptive Platform deployments and require diagnostic infrastructure that can operate over IP-based networks rather than traditional CAN-based UDS protocols [1], [5]. Third, the central diagnostic library will be extended with predictive health indicators derived from fleet-level data analytics, enabling proactive maintenance recommendations that anticipate fault conditions before they reach the DTC threshold [10], [14].

Conclusion

This article has presented a scalable diagnostics infrastructure framework that repositions automotive embedded diagnostics from a per-program engineering task to a governed, centralized, and continuously deployable system asset. The framework's three-layer architecture — centralized diagnostic libraries, automated mapping and deployment orchestration, and real-time validation and observability — directly addresses the structural deficiencies of conventional diagnostic workflows that produce inconsistent fault detection, duplicated engineering effort, and inadequate safety coverage across multi-platform SDV deployments [4], [6], [16]. Evaluation across multiple vehicle programs demonstrated consistent directional improvements in diagnostic deployment efficiency, cross-platform fault detection consistency, manual engineering effort, and ISO 26262-relevant fault detection coverage, alongside a substantial reduction in mapping-error-driven validation failures [3], [15].

The significance of this contribution lies not only in its operational impact but in the conceptual reframing it introduces. Treating diagnostics as infrastructure — governed, versioned, and continuously deployed — is the correct architectural response to the demands of software-defined vehicle development [11], [13]. As vehicle software complexity grows, diagnostic frameworks that cannot scale across programs and adapt continuously to software changes will become critical bottlenecks in the development and safety assurance process [1], [2], [9]. The framework presented here establishes the architectural and operational foundation for diagnostic infrastructure capable of supporting the full lifecycle of software-defined automotive systems, from initial feature development through post-deployment OTA evolution [10], [14].

References

[1] S. Bickelhaupt et al., "Towards future vehicle diagnostics in software-defined vehicles," SAE

International Journal of Advances and Current Practices in Mobility, vol. 7, no. 3, pp. 1240–1254, 2025. [Online]. Available: https://www.researchgate.net/publication/379608220_Towards_Future_Vehicle_Diagnostics_in_Software-Defined_Vehicles

[2] P. V. Teixeira et al., "Software-defined vehicles for development of deterministic services," arXiv preprint arXiv:2407.17287, 2024. [Online]. Available: https://arxiv.org/html/2407.17287v1

[3] ISO 26262-1:2018, Road Vehicles — Functional Safety — Part 1: Vocabulary, International Organization for Standardization, Geneva, Switzerland, 2018. [Online]. Available: https://www.iso.org/standard/68383.html

[4] D. Zyberaj et al., "Test case specification techniques and system testing tools in the automotive industry: a review," Journal of Systems and Software, 2026. [Online]. Available: https://www.sciencedirect.com/science/article/abs/pii/S0164121225004339

[5] ISO 14229-1:2020, Road Vehicles — Unified Diagnostic Services (UDS) — Part 1: Application Layer, International Organization for Standardization, Geneva, Switzerland, 2020. [Online]. Available: https://www.iso.org/standard/72439.html

[6] P. Gai et al., "AUTOSAR university package classic platform," in Proc. IEEE 27th International Conference on Emerging Technologies and Factory Automation (ETFA), 2022. [Online]. Available: https://ieeexplore.ieee.org/document/9921596/

[7] Y. Jin et al., "Design and implementation process of an intelligent automotive chassis domain controller system based on AUTOSAR," Sensors, vol. 25, no. 16, p. 5056, 2025. [Online]. Available: https://doi.org/10.3390/s25165056

[8] 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

[9] Y. Mahale, S. Kolhar, and A. S. More, "A comprehensive review on artificial intelligence driven predictive maintenance in vehicles: technologies, challenges and future research directions," Discover Applied Sciences, vol. 7, p. 243, 2025. [Online]. Available: https://doi.org/10.1007/s42452-025-06681-3

[10] Md N. Hossain et al., "Advances in intelligent vehicular health monitoring and fault diagnosis: techniques, technologies, and future directions," Measurement, 2025. [Online]. Available: https://doi.org/10.1016/j.measurement.2025.117618

[11] F. Zampetti et al., "Continuous integration and delivery practices for cyber-physical systems: an interview-based study," ACM Transactions on Software Engineering and Methodology, vol. 32, no. 3, Article 73, pp. 1–44, 2023. [Online]. Available: https://dl.acm.org/doi/10.1145/3571854

[12] 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://doi.org/10.3390/s24123733

[13] A. Bazzi et al., "A novel variability-rich scheme for software updates of automotive systems," IEEE Transactions on Vehicular Technology, vol. 73, no. 9, 2024. [Online]. Available: https://ieeexplore.ieee.org/document/10547264/

[14] Y. Mahale, S. Kolhar, and A. S. More, "Automated vehicle fault diagnosis and report generation using hybrid machine learning with multi-step RAG approach," Discover Computing, vol. 28, Article 283, 2025. [Online]. Available: https://doi.org/10.1007/s10791-025-09823-8

[15] 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://doi.org/10.3390/electronics13020437

[16] S. Bickelhaupt et al., "Challenges and opportunities of future vehicle diagnostics in software-defined vehicles," SAE Technical Paper 2023-01-0847, 2023. [Online]. Available: https://www.researchgate.net/publication/377964905_Challenges_and_Opportunities_of_Future_Vehicle_Diagnostics_in_Software-Defined_Vehicles