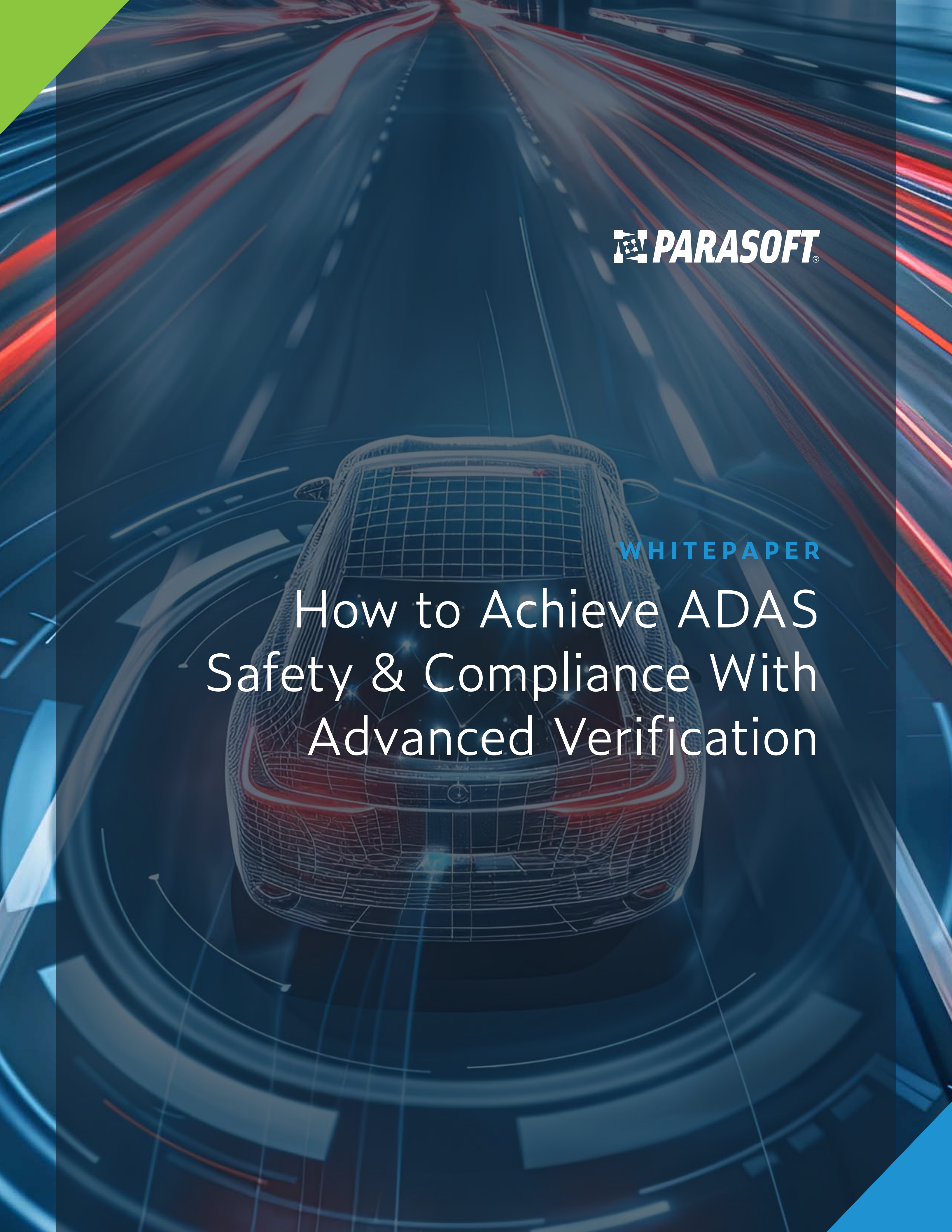

PARASOFT®

WHITEPAPER

How to Achieve ADAS Safety & Compliance With Advanced Verification

INTRODUCTION

Advanced driver assistance systems (ADAS) are revolutionizing automotive safety. These systems offer features like adaptive cruise control, lane-keeping assistance, and collision avoidance. However, the complexity of these systems, comprising millions of lines of code, sensor fusion, and real-time decision-making, demands rigorous verification and validation (V&V) to ensure reliability and compliance with stringent safety standards.

This whitepaper explores how test automation solutions for C/C++ testing streamline ADAS development and address critical challenges, enabling the delivery of robust, safety-compliant systems. By integrating automated testing, static analysis, and seamless CI/CD pipelines, automotive teams can accelerate time to market while mitigating risks.

Ensuring Safety & Reliability of ADAS

ADAS are tasked with operating flawlessly in an unpredictable world where heavy rain obscures lane markings, sudden pedestrian crossings demand split-second decisions, and sensor data can be compromised by glare or fog. There's no margin for error. Even minor flaws in perception, decision-making, or actuation can lead to catastrophic outcomes.

To mitigate these risks, compliance with globally recognized standards like ISO 26262, ISO 21448 (SOTIF), and UN R171 is not just a regulatory checkbox but a lifeline for safety-critical development. For example, ADAS enhances reliability by deploying redundant AI models that operate in parallel. If one model fails, others compensate through consensus or confidence-based voting—maintaining safe operation while adhering to these stringent standards.

The Role of Safety Standards in ADAS

ISO 26262

Governing functional safety for road vehicles, ISO 26262 ensures systems operate safely even when hardware or software malfunctions. It mandates:

- **Automotive Safety Integrity Level (ASIL)** classification entailing assigning risk levels A through D to system components. For example, a steering control module may require the highest integrity level, ASIL D, due to its life-critical role.
- **Hazard analysis and risk assessment (HARA)** for complex ADAS architectures, such as sensor fusion across LiDAR, radar, and cameras, require rigorous safety analyses, including Failure Modes and Effects Analysis (FMEA) and Fault Tree Analysis (FTA).
- **Fault tolerance,** which includes designing redundancy, fail-operational mechanisms, and watchdog timers, for instance, a brake-by-wire system must have backup circuits to prevent total failure.
- **Safety mechanisms** that detect and mitigate faults, such as check sums for memory integrity or heartbeat monitors for software processes.

Compliance with ISO 26262 safeguards against catastrophic failures. It also fosters industry-wide trust in the reliability of modern vehicles. Through ASIL classification and fault-tolerant design, this functional safety standard ensures that systems like Automatic Emergency Braking (AEB), classified as ASIL D, continue to operate safely even during hardware failures. As ADAS and vehicle autonomy advance, ISO 26262 remains a cornerstone of safe and dependable automotive innovation.

ISO 21448—SOTIF

Safety of the intended functionality (SOTIF), also known as ISO 21448, addresses hazards arising from limitations in the system's intended functionality, even when no defects exist. Validation requires exhaustive scenario-based testing, often thousands of simulated and real-world cases, to prove the system behaves safely in "unknown unknowns." Here are some examples.

- **Sensor limitations** like cameras failing to detect pedestrians in low-light conditions.
- **Algorithmic gaps** such as misclassifying a plastic bag as an obstacle, leading to unnecessary emergency braking.
- **Edge cases** include handling rare scenarios like a child darting into traffic from behind a parked car.

Complying with SOTIF demands rigorous scenario-based testing, spanning thousands of simulated and real-world cases. This process is complicated by:

- **Complex software architectures** such as sensor fusion, AI/ML models, and real-time data processing.
- **Regulatory compliance challenges** like overlapping standards requiring traceability, documentation, and risk mitigation.
- **Real-time performance** needs like split-second decision-making with zero tolerance for latency.

Together, these factors underscore the critical balance between innovation and reliability in ADAS development.

UN Regulation No. 171

UN Regulation No. 171 (UN R171) defines Driver Control Assistance Systems (DCAS) as systems that assist the driver in controlling the longitudinal and lateral motion of the vehicle on a sustained basis while not taking over the entire driving task.

Effective since 2024, UN R171 regulates systems like lane-keeping assist (LKA) and automated lane changing. Balancing automation with driver responsibility, especially during handover scenarios, is a crucial challenge. Key requirements include:

- **Driver engagement monitoring.** Ensuring drivers remain attentive and can override the system.
- **System performance limits.** Restricting operation to highways with clear lane markings or speeds below 130 km/h.
- **Fail-safe interventions.** Graceful degradation, such as gradual lane departure warnings, if system confidence drops.

This regulation strikes a vital balance between advancing automation and preserving human oversight, fostering trust in semi-autonomous systems while paving the way for safer, more reliable vehicle automation. As ADAS evolves, UN R171 serves as a cornerstone for harmonizing innovation with real-world safety demands.

ISO/SAE 21434

A globally recognized standard, ISO/SAE 21434 mandates cybersecurity risk management for road vehicles. ADAS are increasingly software-defined and interconnected, making them vulnerable to cyber threats such as sensor spoofing, CAN bus attacks, and malicious over-the-air (OTA) updates. The ISO/SAE 21434 standard provides a structured framework to address these risks by embedding cybersecurity into the entire life cycle of automotive systems.

The framework prioritizes risk assessment to identify and prioritize threats, such as hacking via vulnerable infotainment systems. It emphasizes security by design, embedding cybersecurity measures from the earliest stages of development, and continuous monitoring to adapt defenses as new threats evolve. Additionally, it fosters collaboration across automakers, suppliers, and developers to ensure aligned security practices, creating a unified defense against evolving cyber risks. Key requirements for ADAS cybersecurity include:

- **Threat analysis and risk assessment (TARA).** Identify attack surfaces and vulnerabilities for ADAS components like LiDAR, cameras, V2X modules, and ECUs.
- **Risk prioritization.** Assign severity scores to threats, for example, a hacked steering system poses higher risk than a tampered GPS.
- **Secure development life cycle.** Ensure secure coding practices and enforce coding standards to prevent buffer overflows or injection attacks.
- **Encryption and authentication.** Protect data in transit, such as OTA updates, and restrict access to critical systems.
- **Intrusion detection systems (IDS).** Monitor in-vehicle networks for suspicious activity, for example, unexpected Controller Area Network (CAN) bus messages.
- **Documentation.** Maintain records of TARA results, security controls, and incident response plans.
- **Traceability.** Link security requirements to design, testing, and validation phases.

ISO/SAE 21434 fosters collaboration across automakers, suppliers, and developers to ensure robust defenses while balancing innovation with resilience. As vehicles grow more software-defined and interconnected, ISO/SAE 21434 is indispensable for safeguarding ADAS and autonomous systems against cyberattacks, ensuring trust in an era where digital security is inseparable from physical safety.

Validating AI/ML & V2X in ADAS Systems

As ADAS capabilities evolve, the integration of artificial intelligence (AI), machine learning (ML), and vehicle-to-everything (V2X) communication introduces new dimensions of complexity. These technologies enable smarter decision-making and connected vehicle behavior. But they also present unique verification and safety challenges.

AI/ML in Perception & Decision-Making

Modern ADAS platforms use ML models for object detection, lane recognition, pedestrian tracking, and path planning. Unlike traditional rule-based systems, ML-driven systems exhibit nondeterministic behavior. This makes it difficult to verify their correctness using conventional testing approaches and presents the following challenges.

» **Nondeterminism** appears in model behavior across diverse inputs.

» **Limited observability** into internal decision logic like black box models.

» **Lack of traceability** from safety requirements to learned behavior.

» **Scenario explosion** requires thousands of variations needed for confidence.

To make AI systems more reliable, freezing the model is the first step to determinism—ensuring consistent behavior. Additionally, explainable AI (XAI) techniques improve observability, fostering greater confidence and trust in using AI. However, teams must back this up with extensive scenario-based testing and simulations to flush out edge cases and unthought-out scenarios.

While Parasoft doesn’t perform AI scenario-based testing, C/C++test does assist in testing the implemented C/C++ guardrails to ensure that AI results don’t cause hazardous outcomes. In addition, it offers AI/ML features to help teams improve productivity and efficiency.

» Prioritizes static analysis violations from highest to lowest risk.

» Suggests fixes for identified coding violations.

» Provides instant answers to tool-related questions through an AI chatbot assistant that helps users complete tasks step by step.

Safety in Connected V2X Environments

V2X enables real-time data exchange between vehicles, infrastructure, pedestrians, and the cloud. It enhances situational awareness, allowing ADAS systems to anticipate hazards beyond sensor range, such as sudden braking in a vehicle ahead or changing traffic light signals.

The challenges of delivering safe and secure connected V2X environments include:

» **Timing and latency sensitivity.** Split-second delays may cause incorrect responses.

» **Cybersecurity threats,** such as spoofed messages or unauthorized network access.

» **Interoperability issues** between different communication stacks and vehicle types.

Solutions like Parasoft support the development of safe and secure V2X communication by providing cross-platform dynamic testing to validate C/C++ communication layers under both simulated and real-world conditions. Tools like Parasoft C/C++test enable fuzz testing and static analysis help uncover vulnerabilities in message parsing, buffer handling, and protocol execution, critical for preventing unexpected behavior or crashes.

Beyond this, Parasoft SOAtest validates V2X protocol robustness (DDS, SOME/IP), while Parasoft Virtualize simulates edge-case scenarios for hardware-free testing of ADAS sensor fusion.

For compliance, Parasoft DTP (Development Testing Platform) centralizes traceability evidence for standards like ISO 26262, ISO 21448 (SOTIF), and ISO 21434, ensuring redundant AI models and communication stacks meet stringent automotive safety requirements. Like ISO 26262, DTP also provides end-to-end traceability for AI functional safety requirements under ISO/PAS 8800:2024 and ISO/IEC TR 5469:2024. It seamlessly links test cases to safety objectives while automating the generation of audit-ready reports, ensuring compliance across both traditional and AI-driven automotive safety standards.

Meeting ISO/SAE 21434 cybersecurity requirements demands robust secure coding practices to protect the integrity and authenticity of vehicle communications. Additionally, validating V2X behavior can be challenging without access to physical infrastructure or other connected vehicles.

Enforcing secure coding practices helps ensure compliance with cybersecurity standards while protecting critical vehicle communication systems. Leveraging hardware-in-the-loop (HIL) and virtualized testing enables embedded development teams to thoroughly evaluate V2X interactions—even without physical infrastructure or other connected vehicles—reducing risk and accelerating development cycles.

Accelerating ADAS Development Through Intelligent Automation

ADAS development faces mounting pressure to deliver innovation rapidly without compromising safety. Traditional manual testing struggles to keep pace with the exponential growth of code complexity and regulatory requirements.

Test Automation

Comprehensive C and C++ test automation solutions empower automotive developers to build robust, safety-critical ADAS. Solutions like Parasoft provide a platform that provides static code analysis with advanced control and data flow techniques to detect defects like memory leaks, race conditions, and buffer overflows early. At the same time, enforcing compliance with automotive coding standards like MISRA, AUTOSAR C++14, and CERT.

For dynamic validation, Parasoft automates unit testing and fault injection. Developers can simulate edge cases, such as sensor failures or adversarial inputs, to ensure functional correctness under real-world conditions. The code coverage analysis (statement, branch, and MC/DC) guarantees that safety-critical logic meets ASIL (A-D) targets, while seamless CI/CD integration accelerates feedback loops by embedding testing into any CI pipeline.

Machine learning defect prioritization and preconfigured Test Configurations in C/C++test for ISO 26262 enable teams to:

- Reduce manual effort by up to 70%.
- Cuts remediation costs.
- Deliver audit-ready traceability reports, ensuring ADAS systems are market-ready and inherently safe.

C and C++ test automation solutions, like C/C++test, bridge the gap between rigorous safety standards and Agile development. Developers can innovate faster without compromising reliability.

> ***With CI/CD integrations, teams achieve 30% faster release cycles while maintaining ISO 26262 compliance.***

CI/CD Integration

By integrating with Jenkins, GitLab, GitHub, Bamboo, Bazel, Docker, and Azure DevOps, Parasoft embeds testing into every code commit. Developers receive instant feedback on defects, code coverage, and compliance violations. An example of this instant feedback at play is a continuous testing pipeline flagged a memory leak in an adaptive cruise control module within minutes of code submission. This quick response prevented a critical downstream failure.

With CI/CD integrations, teams achieve 30% faster release cycles while maintaining ISO 26262 compliance. Other benefits include:

- **Simulation-based validation.** Use HIL testing and virtualized environments to replicate real-world scenarios efficiently.
- **Cross-platform testing.** Validate code on virtualized ECUs and real hardware, ensuring seamless integration across heterogeneous architectures.

Showcasing these benefits is Parasoft's HIL testing capabilities, which enable engineers to validate ADAS algorithms in simulated environments, reducing dependency on physical prototypes and cutting validation time by up to 50%.

Detecting Defects Early to Reduce Cost and Time to Market

In ADAS development, the adage "a stitch in time saves nine" is a matter of life and death. Studies reveal that defects discovered post-deployment can cost 100x more to remediate than those identified during coding.

For ADAS, where a single line of flawed code could misclassify a pedestrian, delay emergency braking, or cause unintended acceleration, the stakes are astronomical. Late-stage defects can lead to risk recalls, regulatory fines, and reputational damage. Most critically, they can also endanger lives. For example, a buffer overflow in a lane-keeping assist module might remain dormant until a specific sensor input triggers it during a highway merge—potentially leading to catastrophic lane drift.

Static Code Analysis

Static analysis employs control flow and data flow analysis to ensure algorithms behave as intended under all execution paths. They uncover race conditions in multithreaded code or uninitialized variables in perception algorithms. For example, this analysis detected a buffer overflow in a parking assist module that could have caused unintended acceleration.

> ***Static analysis can identify more than 90% of defects before dynamic testing begins, cutting remediation costs by 40%.***

Static code analysis includes compliance checks to automatically enforce coding standards like MISRA, CERT, and AUTOSAR C++14. These checks substantially reduce manual code reviews and may even eliminate them all together.

Parasoft uses machine learning to prioritize defects based on severity—highlighting critical issues like memory leaks in collision avoidance systems ahead of lower-risk items such as coding style violations. Additionally, a generative AI capability enhances the static analysis process by suggesting code fixes for identified coding violations, helping developers resolve issues faster and with greater confidence.

Static analysis can identify more than 90% of defects before dynamic testing begins, cutting remediation costs by 40%.

By catching defects early in the implementation phase, before integration or deployment, teams can transform their ADAS development from reactive firefighting to proactive risk mitigation. This shift-left approach safeguards against costly rework and compresses validation timelines, ensuring OEMs meet aggressive market deadlines without compromising safety. Early defect detection is the difference between a near-miss and a headline-making disaster.

Dynamic Testing

Dynamic analysis tools, like C/C++test, validate applications through automated unit, integration, and regression testing. They uncover runtime defects, validate real-time performance, and ensure compliance with stringent automotive safety standards like ISO 26262 and ISO 21448 (SOTIF).

Executing code in controlled environments enables teams to address risks that static analysis alone cannot detect, including memory corruption, race conditions, and logic errors under dynamic operational conditions.

Dynamic analysis includes the following.

- Auto-generate test cases for C/C++ functions, classes, and modules, reducing manual effort. For example, tests for an adaptive cruise control algorithm can validate throttle response across speed ranges, like 0 to 120 mph.
- Parameterized testing by defining input ranges, such as sensor values and environmental variables, to stress-test algorithms under diverse scenarios.
- HIL integration to validate code on target hardware, like automotive ECUs, using cross-platform testing like Parasoft's. For example, test a parking assist system's interaction with steering actuators in a simulated environment. Retrieve test results via JTAG, Ethernet, or serial ports, even with limited hardware connectivity.
- CI/CD pipeline integration with embedded dynamic tests in Jenkins, GitLab, Azure DevOps, or other pipelines for continuous feedback. For example, a nightly regression test suite flags performance regressions in a collision avoidance system before they reach integration.

By combining rigorous dynamic analysis with automation and compliance rigor, embedded development teams can be confident their ADAS systems will perform reliably in the real world, where milliseconds and millimeters determine safety outcomes.

How Parasoft C/C++ Test Automation Solutions Enhance ADAS Development

ADAS continues to revolutionize automotive safety, enabling features like adaptive cruise control, lane-keeping assist, and autonomous emergency braking. These systems rely on complex, real-time C/C++ code to process sensor data and execute split-second decisions. Ensuring their reliability and compliance with stringent automotive standards, such as ISO 26262 and ISO 21448/SOTIF, demands rigorous testing.

Parasoft addresses these challenges through a comprehensive test automation solution for C/C++ software designed to streamline development, enhance safety, and accelerate time to market. The solution includes:

Static analysis. Identify vulnerabilities like memory leaks, buffer overflows, and race conditions before code execution. This proactive approach prevents critical flaws in safety-critical modules, such as braking or steering control, from progressing to later stages, reducing remediation costs.

Compliance. Enforce coding standards like MISRA, AUTOSAR C++14, and CERT C to meet ISO 26262 ASIL requirements.

Emerging AI safety standards ISO/PAS 8800:2024 and ISO/IEC TR 5464:2024. Parasoft C/C++ testing solutions validate safety-critical C/C++ code in perception stacks and multi-AI consensus systems through static analysis, dynamic testing, and fault injection, ensuring robustness against algorithmic bias and edge-case failures.

Unit and integration testing. Auto-generate test cases for individual components like radar fusion algorithms and validate interactions between subsystems like sensor-to-ECU communication.

Fault injection. Simulate hardware failures, such as LiDAR dropout, to ensure fail-safe behavior under edge case conditions.

Audit-ready reports. Generate for coding standards like MISRA and CERT, safety and security standards like ISO 26262, ISO 21434, ISO 21448 (SOTIF), and regulations like UN R171 to ensure traceability from requirements to test results.

Structural code coverage. Achieve MC/DC (modified condition/decision coverage) to validate every decision path in logic, such as collision avoidance algorithms). Parasoft ensures 100% coverage for ASIL D components, a nonnegotiable for life-critical systems.

Seamless CI/CD integration. Embed automated testing into Jenkins, GitLab, or Azure DevOps pipelines for continuous feedback. Shift-left testing catches defects early, accelerating release cycles while maintaining compliance.

HIL testing. Validate ADAS software in simulated environments, like icy roads, sensor occlusion, using Parasoft's cross-platform testing. This reduces dependency on physical prototypes and cuts validation costs by up to 50%.

Cybersecurity assurance. Align with ISO/SAE 21434 by detecting vulnerabilities like insecure OTA updates. Solutions like Parasoft SOAtest uncover hidden flaws in connected systems. It includes features for penetration testing and fuzzing, which involve injecting invalid, malformed, or unexpected inputs into a system to reveal vulnerabilities.

Automotive teams experience the following benefits:

- Faster development cycles through automation.
- Early defect detection to minimize risks and costs.
- Compliance with global safety and cybersecurity standards.
- Robust traceability and reporting for regulatory submissions.

Conclusion

ADAS systems represent the future of automotive safety. Their reliability depends on rigorous verification and validation. As software complexity increases, so do the challenges of ensuring performance, compliance, and security. Challenges like real-time performance demands and evolving regulations require a holistic approach to V&V. As ADAS evolves toward full autonomy, the margin for error shrinks.

Parasoft's C/C++ testing solutions provide the automation, insights, and traceability needed to meet these demands. With a commitment to innovation, like static analysis augmented with AI/ML and HIL testing, we're your critical partner in shaping the future of automotive safety. With support for static and dynamic testing, coding standards, safety regulations, and cybersecurity, Parasoft empowers automotive teams to deliver safer, more reliable, and more efficient ADAS systems.

TAKE THE NEXT STEP

Request a Demo to see how Parasoft can transform your ADAS development process.

About Parasoft

Parasoft helps organizations continuously deliver high-quality software with its AI-powered software testing platform and automated test solutions. Supporting the embedded, enterprise, and IoT markets, Parasoft's proven technologies reduce the time, effort, and cost of delivering secure, reliable, and compliant software by integrating everything from deep code analysis and unit testing to web UI and API testing, plus service virtualization and complete code coverage, into the delivery pipeline. Bringing all this together, Parasoft's award-winning reporting and analytics dashboard provides a centralized view of quality, enabling organizations to deliver with confidence and succeed in today's most strategic ecosystems and development initiatives—security, safety-critical, Agile, DevOps, and continuous testing.