Whitepaper

From Code to Road:
Building an End-to-End Software Factory for Software-Defined Vehicles

Simona Almajan and Guillaume Cordon, NXP Semiconductors

Abstract

The central challenge is no longer whether automotive companies can write more software. It is whether they can integrate, validate and deliver software fast enough to keep pace with vehicle complexity without compromising safety, security or quality. An end-to-end software factory addresses that challenge by connecting development, testing, integration and delivery across the semiconductor-to-vehicle value chain.

This paper outlines an end-to-end DevOps approach for automotive software systems, spanning semiconductor software, platform software and system-level integration. Using NXP's software ecosystem and the existing CoreRide platform reference in this document as examples, it shows how a connected software factory can help engineering teams validate earlier, reduce integration friction and scale development more effectively.

The paper explains how continuous integration, continuous testing and continuous delivery can be applied across the automotive stack to improve software quality, traceability and release readiness. It also highlights why virtualization, pipeline orchestration and cross-ecosystem integration are becoming essential engineering capabilities for software-defined vehicles.

To measure progress, engineering teams can use software delivery metrics from DORA, including deployment frequency, change lead time, change failure rate and recovery time. For OEM and Tier 1 engineers, the central message is practical: DevOps is no longer a software-only discipline. It is a system-level capability for delivering safe, scalable and continuously evolving vehicle platforms.

Introduction

The automotive industry is moving from hardware-centric vehicle programs to software-defined vehicles, where software increasingly determines functionality, differentiation and lifecycle value. This shift is changing not only vehicle architecture, but also how engineering teams must build, integrate, validate and maintain software.

Today's automotive software ecosystem faces unprecedented challenges:

- **Exponential complexity:** Modern vehicles run over 100 million lines of code, managing everything from ADAS and electrification to complex infotainment.
- **Stringent safety and compliance:** Unlike consumer electronics, automotive software must strictly adhere to international standards ISO 26262 (ASIL) and ASPICE as well as international homologation constraints, where a failure in the field can have life-critical consequences.
- **The "always-connected" expectation:** Consumers now expect their cars to improve over time through smartphone-like updates, requiring OEMs to reliably deliver secure Over-the-Air (OTA) updates.

To bridge the gap between "silicon speed" and "safety requirements," the industry needs to strengthen its DevOps strategy.

At NXP, this shift is driving a move from isolated component delivery to a more connected software lifecycle built around continuous integration, testing and delivery. The goal is to help customers manage complexity through pre-integrated platforms, earlier validation and more reliable system-level software delivery.

Reference architecture: the NXP automotive software stack

To ground the discussion, we consider a representative automotive software stack aligned with NXP's ecosystem.

a) NXP software platform/SDK layer: integrated software development kits, include NXP hardware board support package (BSP), NXP and partners real time operating system (RTOS) and middleware frameworks, toolchain and configuration environments.

b) CoreRide platform: hardware/ software foundation for building an electronic control unit (ECU), validated and performance optimized. Reduces integration complexity via comprehensive software BSP integrated with partners software (Autosar OS, Gliwa T1), shift left quality through CI/CT/CD ready development environment.

While NXP creates and delivers higher value with CoreRide platform, new challenges are introduced at the interfaces between the hardware and software layers, therefore, there is the necessity for new integration and regression strategies powered by NXP System Software Factory.

Continuous "everything" (Cx) across the automotive stack

Once the product architecture is defined, the next challenge is making it executable at scale. Engineering teams must connect software units, software development kits (SDKs), middleware, operating systems and integrated platforms in a way that remains stable as features evolve, variants multiply and requirements change.

The integration strategy must preserve quality, cybersecurity, performance and functional safety across the stack. That is why continuous integration, continuous testing and continuous delivery have become foundational capabilities for software-defined vehicle development.

Continuous integration (CI) brings frequent software changes into a stable mainline and triggers automated regression from software units to integrated products. In automotive environments, CI must support large build artifacts, multiple hardware and software variants and the fast feedback loops engineers need to keep development moving.

CI must operate across multiple levels:

- Software unit level integration aligned to silicon changes
- SDK level integration across configurations and variants
- System level integration across ECUs and domains

A key requirement is pipeline connectivity, so that a change at any level can be traced through dependent layers and validated through the appropriate regression checkpoints. This enables earlier defect detection, more reliable mainline quality and faster issue triage when regressions occur.

Continuous testing (CT)
extends CI with automated validation at multiple checkpoints, including pull requests, daily builds, nightly builds and release candidates. For automotive software, CT must cover not only functional behavior, but also performance, robustness, security and compliance evidence across the lifecycle.

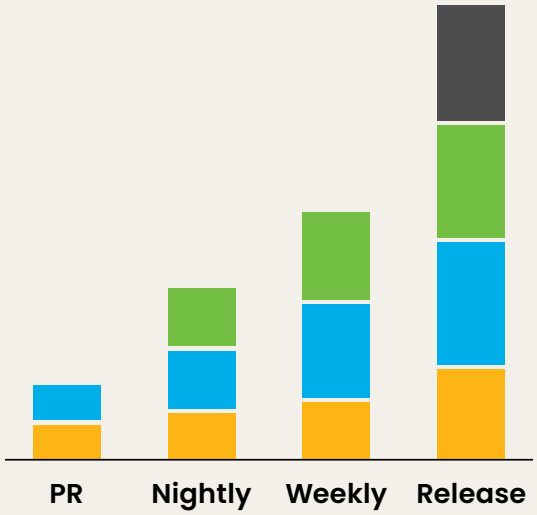
CT helps ensure that software remains release-ready by applying validation at different cadences across the stack. It supports shift-left engineering by moving more integration and compliance checks earlier in the lifecycle, reducing late-cycle surprises and improving confidence in release quality.

Continuous delivery (CD)
ensures that once software is validated, it can be packaged and delivered in a repeatable way to internal teams, ecosystem partners and customers. In practice, this means versioned, traceable, production-ready deliverables that include software, configuration and the evidence required for downstream integration and release decisions.

- Deliver everything as code (config, integration code, build toolchain, test specifications artifacts, etc.)
- Deliver change history (in compliance with IP protection and data privacy) allowing complete transparency on code changes and possibility to cherry pick specific features by customers.

CoreRide platform is extended with specific validation techniques which are specific to its value proposition:

- Key performance indicators are measured every day to validate the overall platform performances (such as power consumption is all platform states, cold/warm boot time including safe and secure boot, network performances across diverse network load and network gateway scenarios).
- Robustness of the platform including fault-injection at system (ECU) level (back-to-back boot, malicious traffic, pen-testing, etc.)



- Smoke and basic stability
- Impacted functionality (change-based regression)
- Key integration paths and selected NFR
- Full functional and full NFR (robustness, timing, soak)

In conclusion:

- CI focuses on building and integrating software in a continuous manner. Catching regression as soon as possible (shift left from integrated systems to unit development) and allow quick revert of a contribution that broke the mainline.
- CT ensures comprehensive testing at all levels and left-shifted with regression suites from next level of integration, as well as security/safety/compliance as code directly in the testing pipelines. It also includes extended test campaigns for specific hardened products with KPI, robustness/reliability tests.
- CD enables releasing at a high frequency, seamlessly, ideally in a common environment between suppliers in the chain and customers, where DevOps pipelines are interconnected.

Together, CI, CT, CD form a close-loop system that supports rapid, yet safe innovation.

A CI/CT/CD-ready environment is ideal for rapid adoption, continuous builds and accelerated software quality through higher frequency end-to-end feedback loops.

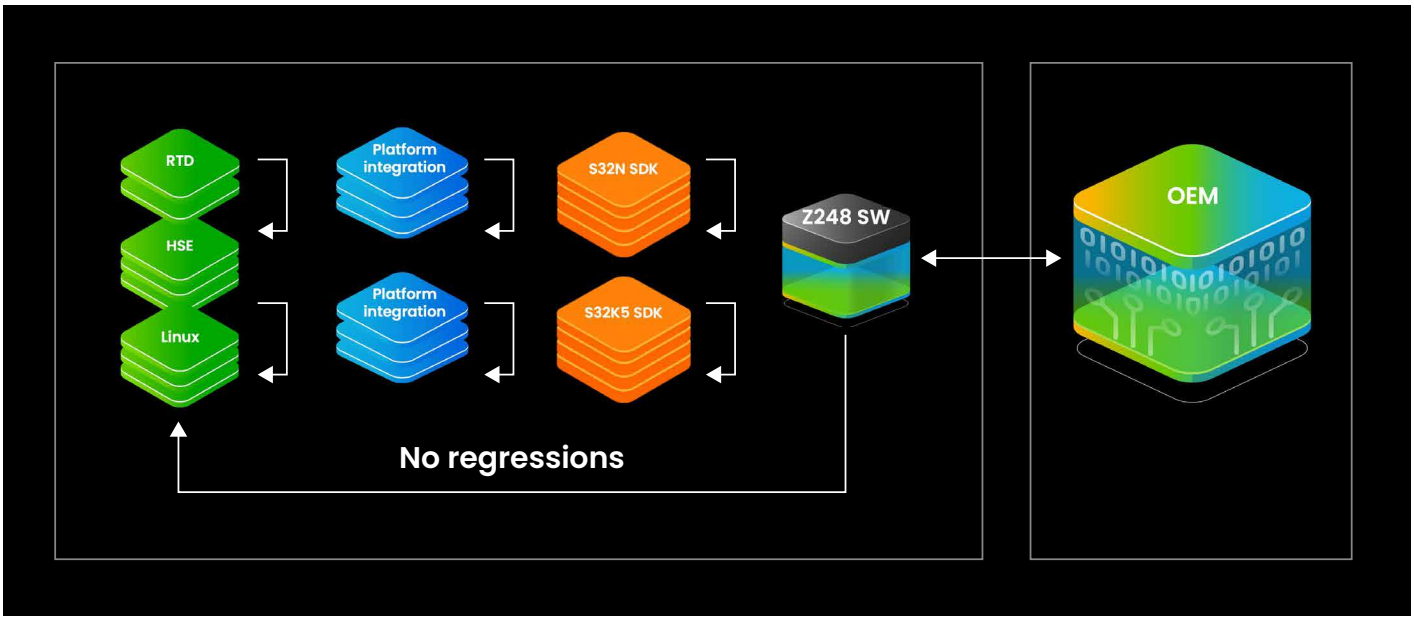
An integrated DevOps approach enables:

- Increased deployment frequency through reliable DevOps pipelines.
- Reduced lead time via automated, cross-layer CI.
- Lower failure rates through early and continuous testing.
- Faster recovery through improved observability, feedback loops and AI-accelerated triage/revert sessions.

Measuring DevOps performance with DevOps research and assessment (DORA) metrics

At NXP, we embrace DORA metrics:

- **Deployment frequency use:**
  How often NXP is able to release integrated products (SDK and CoreRide platform).
- **Lead time for changes:**
  The time from code commit to validated SDK or CoreRide platform (in form of an qualified engineering release).
- **Change failure rate:**
  The percentage of changes that introduced a regression.
- **MTTR (mean time to recover):**
  How quickly the teams can restore a stable baseline after error introduction.



DevOps connects ecosystem players — NXP provides the platform

As the automotive industry spans across multiple ecosystem partners (semi vendors, Tier1, OEMs, software partners/integrators) delivery times are critical and lack of interconnected infrastructures between contributors in the value chain is introducing bottlenecks.

NXP enables standardized artifact interfaces, end-to-end traceability across organizational boundaries and integration with customers' and partners' environments.

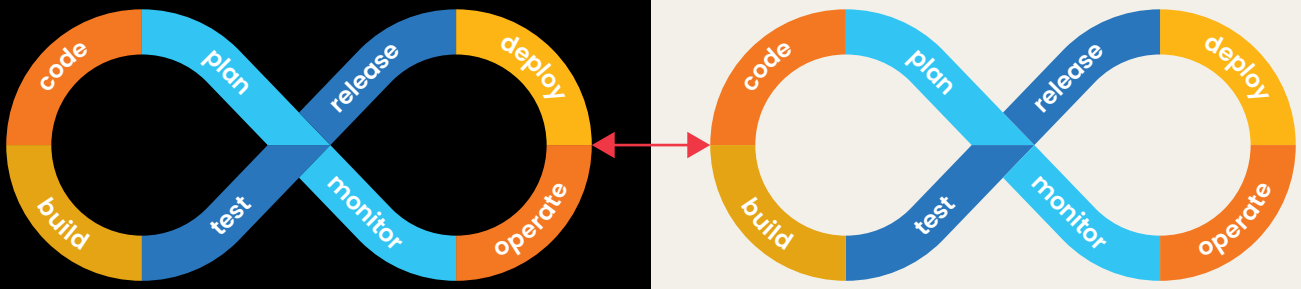
This creates a continuous DevOps value chain reducing integration friction and improving transparency.

NXP environment

Single NXP system software factory environment - connected to customers with a standard interface

Customer environment

Customer CI/ CD interconnected systems for fast feedback based on validation in customer environment



Investment areas: building the foundation for end-to-end automotive DevOps

Achieving measurable improvements in software development performance requires deliberate and sustainable investment. NXP is investing across software, hardware, safety and ecosystem integration to help customers reduce complexity and build the foundation for end-to-end automotive DevOps.

Some examples of investments areas:

Implementation of end-to-end DevOps pipelines orchestration across software layers from software components to system level software.

This necessitates:

- Cross layers pipelines triggering and dependency management
- Unified artifact management and traceability
- Bi-directional feedback loops (e.g system level defects traced back to component level changes)
- End-to-end orchestration is critical for reducing the lead time for changes by eliminating manual handoffs and late integration cycles

Scalable compute infrastructure and modern tooling

Due to intensive computationally workloads (e.g due to high volume of automated testing, high number of hardware variants) NXP invests in: cloud native build environments, modern build tools like Bazel, one trunk development practices.

Achieving measurable improvements in software development performance requires deliberate and sustainable investment. NXP is investing across software, hardware, safety and ecosystem integration to help customers reduce complexity and build the foundation for end-to-end automotive DevOps.

Variant and configuration management

NXP platforms, such as S32, support a wide range of hardware variants, derivatives, configurations, customer specific adaptations.

- Effective DevOps must system on chip (SoC)/hardware variant multi SoC (multi-dimensional build and test matrices)
- Consistency of configurations across layers
- High automation and optimization of the variant specificities

NXP investments in these areas are essential to maintain low Change Failure Rates at scale.

Data, observability and feedback loops

A robust dashboarding solution closes the loop between development, validation and infrastructure. This implies the need for:

- Real-time pipelines observability (build, test, deployment metrics)
- Traceability from requirements to code and validation results
- Feedback loops from system to software component layers

Strong observability is critical for reduction of MTTR (mean time to recover) and enabling continuous improvement across the lifecycle.

Customer benefits – translating DevOps investment into business value

The ultimate objective of NXP's investments in end-to-end DevOps is not only process optimization – it is delivering measurable value to our end customers: OEMs and Tier 1 suppliers.

The following benefits are crucial: reduced time-to-market, improved software quality and safety, lower integration risk, proven system capabilities, stronger collaboration across value chain, improved operational resilience (lower MTTR, lower failures, etc.)

NXP's investment in DevOps methodologies translates directly into competitive advantages for OEMs and Tier-1 suppliers:

Accelerated time-to-market:
Platforms like CoreRide simplify hardware-software integration, significantly reducing the time needed to launch new SDV architectures.

Reduced development costs:
Early testing via virtual development platforms allows teams to start validation months before silicon is available. This "Shift-Left" approach eliminates costly errors in later stages.

Quality and safety assurance:
Automated compliance checks for **ISO 26262** and **ASPICE** ensure every deliverable meets strict safety criteria, minimizing the risk of software-related recalls.

Ecosystem scalability:
Access to integration partners provides flexibility and can reduce cross-domain integration effort.

Lower integration risk:
less product failures, seamless integration.

Proven system capabilities
like CoreRide platform which is a system-level solution that integrates compute, networking and power management with energy distribution aggregated with foundational software.

Stronger collaboration across value chain
by connected NXP with Tier1/OEM's DevOps pipelines.

Improved operational resilience
ensured via DevOps metrics like lower MTTR (mean time to recover) and lower CFR (change failure rates).

Conclusion

For software-defined vehicles, DevOps must evolve from isolated pipelines into an end-to-end engineering capability that spans semiconductor software, platform software, system integration, validation and delivery across the broader ecosystem.

When continuous integration, continuous testing and continuous delivery are applied across the full stack, engineering teams gain earlier validation, stronger regression control, better traceability and faster recovery from defects. Measured through DORA metrics, these capabilities can improve software delivery performance while reducing integration risk.

NXP's opportunity is to turn its software portfolio into a scalable system platform that helps OEMs and Tier 1s move faster from architecture to production. By combining pre-integrated platforms, ecosystem software and software-factory practices, NXP can help customers reduce complexity while accelerating innovation in software-defined vehicles.

For engineering teams building next-generation vehicle platforms, software leadership will increasingly be defined by the ability to integrate, validate and deliver at system scale. That is the role an automotive software factory is designed to play.

nxp.com/SDV

Document Number: CODESDVWP REV 1