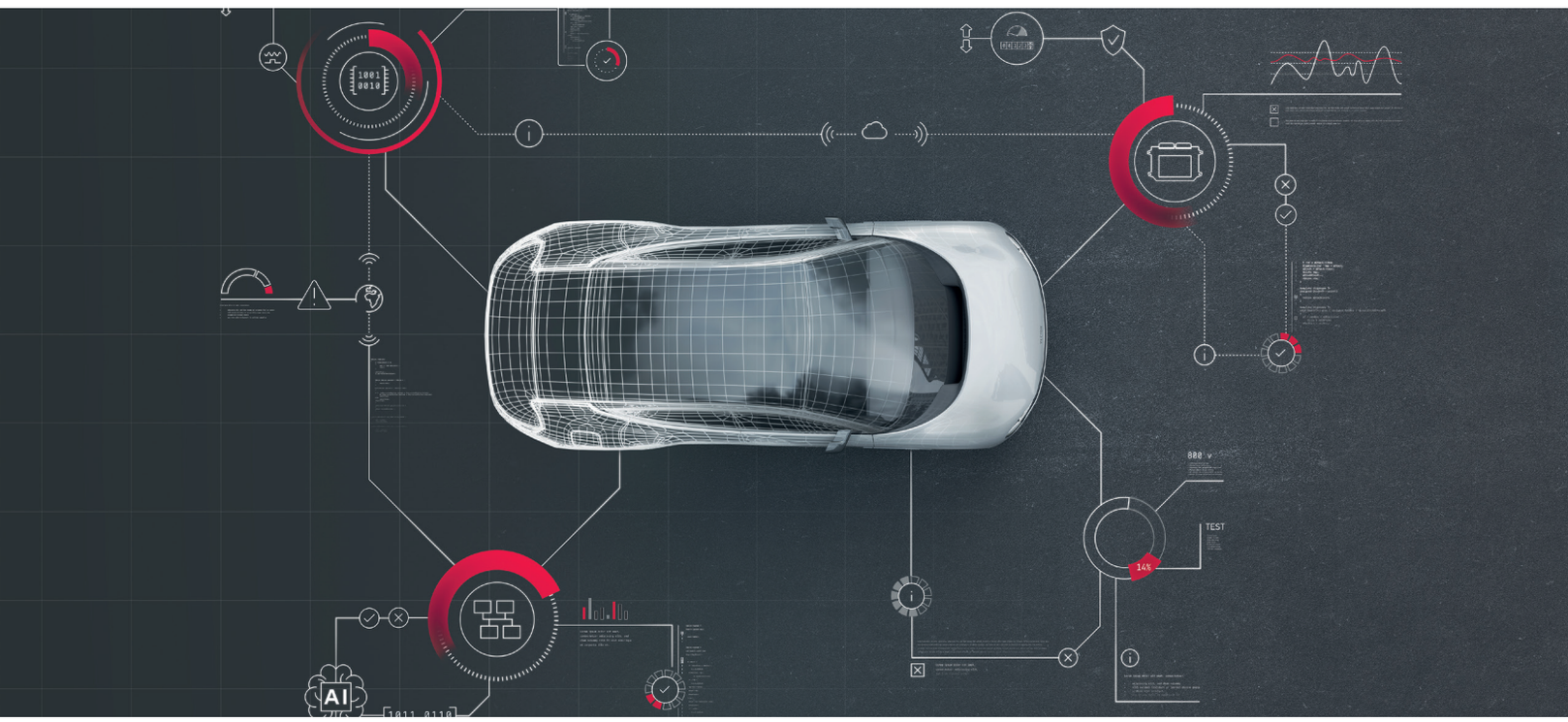

Product Maturity Starts in the Code

How SIL Closes the Gap Between Fast Software Development and Limited Test Capacity

Validation can no longer keep pace with modern software development. Organizations that build product maturity earlier accelerate feedback, reduce pressure on limited HIL resources, and reserve test benches for their intended purpose: final system validation.

Public discussion of Software-Defined Vehicles (SDVs) usually focuses on zonal E/E architectures, central High-Performance Computers (HPCs) (see Figure 1), or Over-The-Air Updates (OTA). In day-to-day engineering, however, a different question determines delivery reliability, quality, and time to market: How can software be validated at the required pace without HIL and test-bench capacity becoming a bottleneck?

Today, vehicle functions are no longer developed in isolation on individual ECUs. They result from the interplay of distributed software stacks, communication layers, and platform software across different architectures, operating systems, and platform concepts. Every change can affect multiple domains. At the same time, the number of variants is growing, release cycles are getting shorter, and expectations for OTA updates throughout the vehicle lifecycle are rising. Validation is therefore becoming harder to scale across the industry, creating an operational bottleneck for many development teams.

The V-Model Has Lost Its Rhythm

The traditional V-model assumes that hardware, software, and the vehicle program all mature at roughly the same pace. In SDV development, that assumption no longer real-

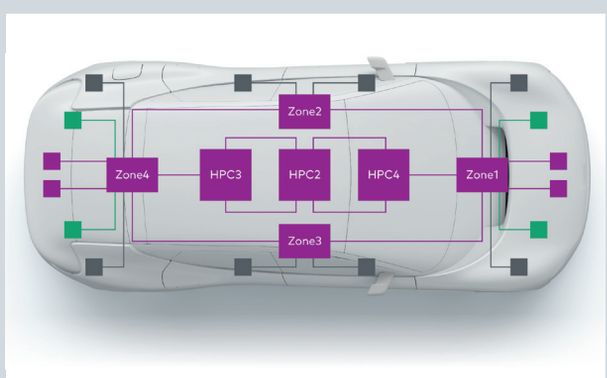

Figure 1: Central computers and zonal E/E architectures are changing the validation logic in the SDV

ly holds. Software moves in rapid sprint cycles, while hardware samples, integration environments, and HIL setups take longer to provision. From a cost perspective, this can no longer be solved simply by adding more validation layers or HIL capacity.

This has far-reaching consequences for complex vehicle projects. Defects often only become visible late in the process when the system is integrated or during vehicle testing. Root-cause analysis spans the entire software stack, from application logic and basic software to communication between ECUs. What may look like an isolated technical issue is usually a symptom of an underlying structural problem: validation can no longer keep up with the pace of software development.

For engineering, testing, and project management, this means the goal is not simply to test more, but to build validation earlier, at scale and in a more cost-effective way. Software maturity needs to be achieved earlier, so HIL and test-bench resources are not tied up by issues that could have been caught virtually.

Software-In-The-Loop: Validation Keeps Pace With Development

This is exactly where Software-in-the-Loop (SIL) comes into play. Rather than waiting for ECUs or test-bench capacity to become available, the software under test can run virtually within simulated interfaces, communication models, and virtual ECUs (vECUs). This makes it possible to execute, observe, and automatically evaluate functions much earlier as soon as new software versions are available.

The impact is structural: issues reach the responsible teams earlier, often shortly after a new software version enters the automated pipeline. Regressions become visible while the context of the change is still fresh. At the same time, root-cause analysis takes less effort because there are less time and less integration depth between the trigger and the symptom.

SIL also offers a clear scaling advantage. Virtual test environments can run in parallel, on developers' workstations, in build infrastructure, in data centers, or in the cloud. This decouples test throughput from the availability of physical hardware. For organizations with multiple teams, platforms, and product lines, this is what enables validation to scale across the organization.

One point remains essential: SIL does not make hardware validation obsolete. It shifts validation activities earlier and frees up HIL and test-bench resources for the cases that require real hardware, timing, I/O, and electrical behavior. SIL does not make HIL less relevant, it makes every HIL hour more valuable and more cost-effective.

Division of Labor Instead of Replacement

The question "SIL or HIL?" is misleading. The real question is: Which validation task belongs where in the development pipeline?

SIL is at its best when used for early, frequent, and highly repeatable checks of application logic, communication behavior, ECU configuration, and integration across stack boundaries. HIL remains essential wherever real electronics, real-time behavior, and electrical effects cannot be modeled sufficiently.

Organizations that establish SIL as a core practice relieve HIL in two ways: they reduce the volume of physical testing and reserve HIL for the validation tasks where it adds the most value. Test benches are freed up for the very tasks they were designed for (see Figure 2).

A publicly documented Mercedes-Benz case study clearly shows the efficiency potential of a server-based, automated testing approach: Approximately 95% of defects were already detected in Software-in-the-Loop before they reached later validation stages [1].

For management, this changes the investment logic. Rather than continuing to scale physical test capacity linearly, a growing share of validation demand can be met through virtual stages, delivering better scalability per euro invested.

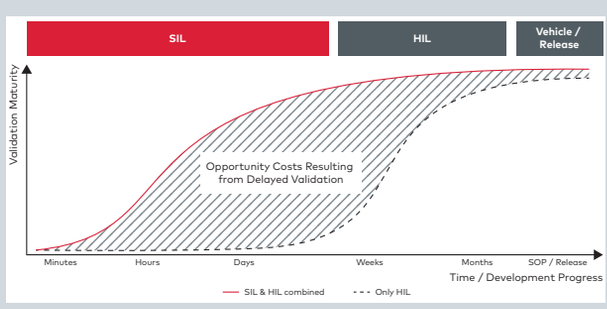

Figure 2: SIL helps build software maturity earlier and reduces late integration risks before the HIL phase

Validate in Stages Instead of Simulating Everything at Maximum Fidelity

Virtual ECUs are typically classified into five levels, from Level 0 to Level 4, based on their degree of integration (see Figure 3). For decision-makers, the taxonomy matters less than the economic logic behind it: the highest level of simulation is not automatically the best choice, the right level is the one that makes economic sense.

Every additional level of realism adds setup, maintenance, and execution time. Organizations that overlook this simply shift the bottleneck elsewhere.

In practice, two virtual stages deliver most of the value. At Level 1, the application software runs in isolation, making it

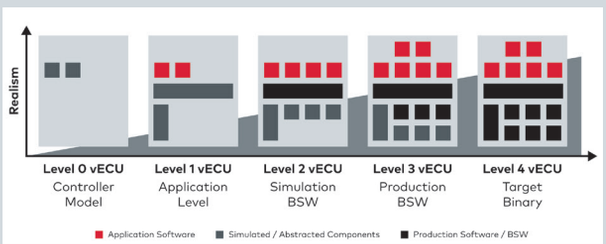

Figure 3: Integration Levels of Virtual ECUs

ideal for fast functional testing, early debugging, and regression testing directly in the development flow without relying on basic software or target hardware.

At Level 3, the vECU also includes the basic software, ECU configuration, and communication behavior. At this stage, teams can answer costly questions early: Does the configuration work with the application? Does communication behave as specified? Can the entire software stack be integrated?

HIL comes into play where models reach their limits. The strategic logic is clear: software maturity is built step by step, rather than being left to a late acceptance phase.

DevOps Without SIL Ends at the Build

Continuous Integration (CI) and Continuous Delivery (CD) are already well established in many development organizations. End-to-end Continuous Testing (CT) for embedded vehicle software, however, is still far less common. This requires test environments that can be run, automated, and reproduced whenever needed without relying on hardware availability.

This is where SIL makes its strategic contribution. Embedded in build and integration pipelines, SIL becomes an executable quality gate. After every change, defined test scopes can be started automatically, results evaluated, and release decisions prepared.

What matters most is not virtualization itself, but how well it is integrated into the process: linking requirements, software versions, variants, test data, and reports into a traceable chain. Only this connection can answer the questions that truly matter to development organizations: Which software version passed which tests? Which variant was covered? Which requirements are affected by a failed result?

SIL thus becomes part of the operating model not as another testing discipline, but as a way to manage software maturity.

Complexity Can Only Be Managed Through Collaboration

One of the biggest challenges in modern SDV development is the wide range of platforms involved. Today's technology landscape spans everything from microcontrollers to high-performance CPUs, and from safety-critical real-time environments to POSIX-based systems and cloud resources. This further adds to the diversity of toolchains, development kits, libraries, operating systems, and processor architectures. Different supplier structures add another layer of complexity.

This cannot be addressed through isolated tool silos. What is needed is an integrated validation platform that connects virtual and physical validation stages, enables the reuse of test artifacts, and provides clear visibility into maturity, quality, and integration readiness (see Figure 4).

In practical terms, this means vECUs at different integration levels, a consistent simulation and execution environment across SIL and HIL, structured test design with reusable test sequences, robust CI/CT integration, and reporting and traceability that link results to requirements and variants. Above all, it means being able to reuse test logic across SIL and HIL rather than maintaining two separate testing worlds.

Vector provides the building blocks for this integrated validation approach: virtual ECUs with vVIRTUALtarget, consistent SIL and HIL testing with CANoe, distributed simulation with SIL Kit, and reusable test design with vTESTstudio. The real value lies in how these tools work together: test artifacts can be reused across stages, results become comparable, and software maturity can be assessed more consistently.

Introducing a tool may optimize local workflows. A platform, by contrast, creates scale, increases engineering throughput, and makes software maturity more transparent.

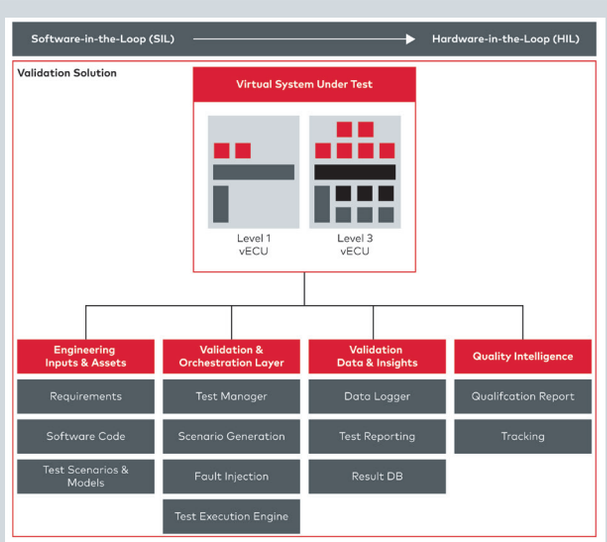

Figure 4: An integrated validation solution connecting virtual and physical test stages

Virtual Validation Belongs on the Management Agenda

SIL is less about technical feasibility than about how it is implemented across the organization. Its benefits only materialize when virtual validation is planned early, introduced with clear objectives, and embedded across development, integration, and validation. If SIL is treated solely as a validation topic, it may deliver isolated efficiency gains but not a structural increase in throughput.

Four questions are key to introducing SIL: Which types of defects should be detected early in a virtual environment? When is the software version mature enough for HIL and test-bench validation? Which test artifacts, configurations, variants, and reports need to remain consistent across SIL and HIL? And how should validation results inform maturity assessments, release decisions, and priorities?

This puts SIL on the management agenda. Clear responsibilities, investment in automation, and an economically sound balance between virtual and physical validation are key. At its core is the question of whether validation will remain a collection of individual tools or become an end-to-end platform capability across the development organization.

Conclusion: Software Maturity Must Be Built Earlier, or It Comes Too Late

The growing importance of Software-in-the-Loop (SIL) does not make hardware testing any less relevant. On the contrary, it makes targeted hardware testing even more important. SDV development needs a validation model that can keep pace with software: changes happen more often, integration dependencies are becoming more complex, and physical test infrastructure cannot scale at the same speed.

SIL shifts validation to where software is developed: earlier in the process, closer to each change, and with much shorter feedback cycles. This increases test throughput and allows HIL resources to focus on validating more mature software versions. The key is to embed SIL in an end-to-end validation platform, methodically, technically, and organizationally. Only then can a new validation economy emerge: software maturity is built earlier, late integration loops are reduced, scarce resources are used more efficiently, and quality and release readiness become more transparent.

This makes SIL more than a testing method for decision-makers. It becomes a lever for engineering performance. Organizations that establish virtual validation in a structured way not only expand the testing organization's toolbox but also lay the foundation for scalable software development in the Software-Defined Vehicle.

Dr.-Ing. Christian Köllner began his career at Vector Informatik in 2013 and has since held positions in software development and product management with a focus on virtualization. He currently serves as Vice President, overseeing SIL and virtualization tools.

Christian Bluthardt has been working in business development at Vector Informatik since 2026, focusing on SIL platforms, validation, and DevOps. In this role, he supports customers in implementing scalable Software-in-the-Loop approaches and establishing a connection between virtual validation and existing HIL processes.

Translation of a German publication in Markt&Technik www.markt-technik.de, issue 27/2026.

Literature References:

[1] Vector Informatik GmbH: Enabling SDV Innovation Through Server-Based Automated Testing. Success Story Mercedes-Benz, 2025. Available online at: https://www.vector.com/int/en/products/products-a-z/software/canoe-server-editions/success-story-mercedes-benz-vtesting/