Springer Vieweg

ATZ extra

Ai-assisted Test-Development that makes Vehicle Software reliable

Cognizant Mobility

© Cognizant Mobility

AI-Driven Test Automation for Embedded Electronic Systems

WRITTEN BY

Tomasz Witkowski, M. Sc. Eng.
is Technical Lead E/E System Design & MBSE Competence Lead at Cognizant Mobility in Munich (Germany).

Dipl.-Inform. Jörg Hofmeyer
is Head of E/E XiL-Test & SW at Cognizant Mobility in Munich (Germany).

Dr. Daniel Isemann
is Head of Data Science & AI at Cognizant Mobility in Munich (Germany).

Cognizant Mobility provides a practical report on the use of AI along the entire test value chain – from requirements to test case analysis. The article shows how each step in this chain can be supported by AI and automation technologies.

Modern high-end vehicles contain more computing units than some spacecrafts. Ensuring that onboard electronics, infotainment systems, ADAS/AV functions and connectivity reliably meet all requirements has become challenging. Traditional quality assurance is therefore increasingly supplemented by Model-Based Systems Engineering (MBSE) and test automation in order to achieve higher test coverage at lower cost.

MODEL-BASED SYSTEMS ENGINEERING

Model-Based Systems Engineering (MBSE) has been gaining traction since the mid-2000s with the introduction of SysML 1.0

automation begins with precise, testable requirements from which a case design is derived, including clear preconditions, expected results, and systematic input values. From this, the case specification is created with concrete test cases and data. Implementation into automated testing is carried out, depending on the test type, using generators, keyword-driven approaches, or fully scripted solutions, with the variety of variants and the need for residual bus simulations posing particular challenges. In the test environment, scripts, simulations, and configurations are integrated and connected to the ECUs; ideally, this step is automated via CI/CD. The execution includes running the tests, monitoring, and handling unstable tests, followed by analysis, defect documentation, and reporting. A clear test strategy and professional release management complete the process.

Experience shows that AI can support nearly every step of the testing pipeline. Domain and system knowledge can

and further increased in importance through INCOSE's MBSE 2020 Vision [1, 2]. For automotive development, it provides an end-to-end system model across the entire V-Model, improving traceability, consistency, and collaboration while reducing the effort required for tool integration. To manage growing system complexity, OEMs and suppliers have increasingly adopted MBSE over the past decade. Tools such as Cameo, Enterprise Architect, and Rhapsody, as well as automotive-specific platforms like PREEvision, Siemens Capital, or SystemDesk, support this transition. To this day, MBSE complements traditional multi-tool environments. It offers an excellent foundation for automating and AI-enhancing the entire testing process [3].

STATE-OF-THE-ART TESTING

The starting point of a holistically understood test process [4] is a deep domain and system understanding that considers the overall system beyond the individual component, because what works in isolation may fail within the chain of effects over all ECUs, **FIGURE 1**. Test

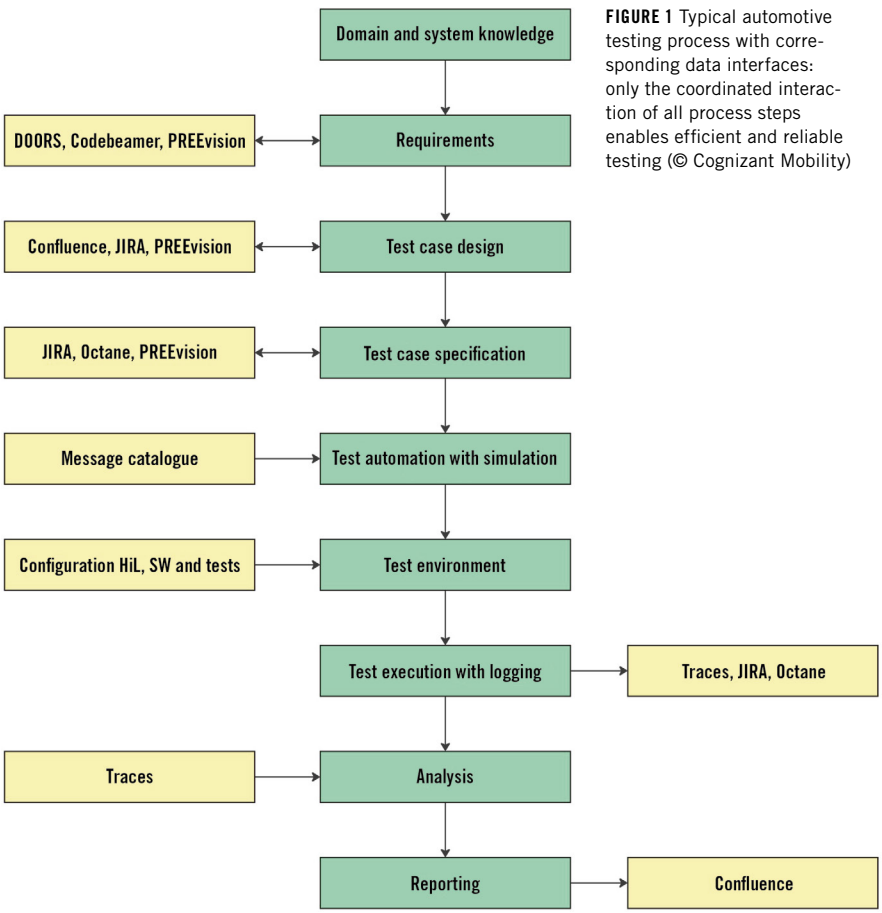


FIGURE 1 Typical automotive testing process with corresponding data interfaces: only the coordinated interaction of all process steps enables efficient and reliable testing (© Cognizant Mobility)

be accessed much quicker using Large Language Models (LLMs) or bots that draw on internal sources such as PDFs, Word documents, or Confluence. Additional information, such as message catalogues or configurations, can be efficiently integrated via RAG or MCP connectors (JIRA, Confluence). A clear creation and review cycle is essential to ensure traceable results and a high-quality standard. The review step harmonizes existing artifacts. This article shows how an existing MBSE foundation can be systematically extended with AI.

PREEVISION MEETS OLLAMA

Cognizant Mobility uses the requirements engineering tool PREEvision by Vector. Since PREEvision does not yet offer a native GenAI integration, a web app was developed internally, **FIGURE 2**. To ensure information security for the sensitive requirements data, the LLMs needed for inference are hosted locally on an Ollama server within a secured computing environment. After an evaluation phase using various open-weight models, a model from Mistral was adopted. The architecture enables seamless switching between different open-weight models as well as providers such as OpenAI, Anthropic, and Google. Development focused on effective, low-complexity applications such as automated requirement quality checks and AI-assisted test case generation.

REQUIREMENTS REVIEW

For the requirements review, a few-shot prompting system was implemented to perform a qualitative assessment of the requirements provided as input. The LLM was supplied via targeted prompting mechanisms with its role, purpose, and context, quality criteria, a plan for conducting the review, examples of correctly and incorrectly evaluated requirements, as well as the desired JSON output schema. Using a moving-window approach, all requirements were processed sequentially, with the two neighboring requirements before and after included for additional context.

The result of the review consists of:

- *Overall Quality* – a mathematical average across all ten quality criteria
- *Review Statistics* – insight into the tokens used and the time required to review a single requirement to understand potential costs and the time needed to assess entire sets of requirements
- *Quality* – a score between 0 and 5 for each of the ten criteria for well-written requirements: clarity, testability, unambiguity, atomicity, consistency, feasibility, verifiability, and measurability, accompanied by a brief explanation of the given score, **FIGURE 3**
- *Issues* – for criteria rated 3 or below, the system presents suggestions for improvement to the user
- *Open questions* – follow-up questions raised by the system that should be considered when formulating the requirements
- *Suggestions* – proposals generated by the AI: simple corrections to grammar, punctuation, and typographical errors or a rephrasing of the requirement while preserving its meaning as well as splitting the requirement into smaller, more atomic variants if atomicity receives a low rating, **FIGURE 4**

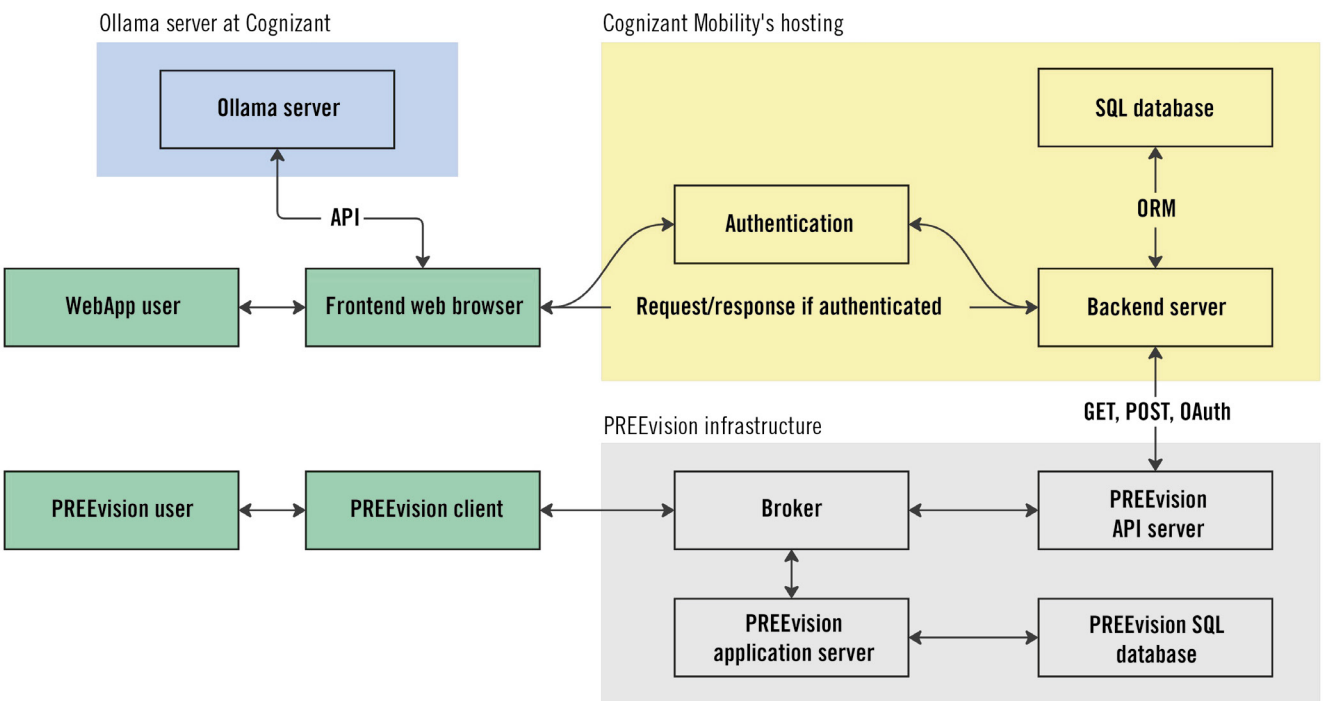


FIGURE 2 Architecture of the Cognizant Mobility WebApp: the web app uses the PREEvision REST API to flexibly connect PREEvision with AI models of various vendors via a locally hosted Ollama server (© Cognizant Mobility)

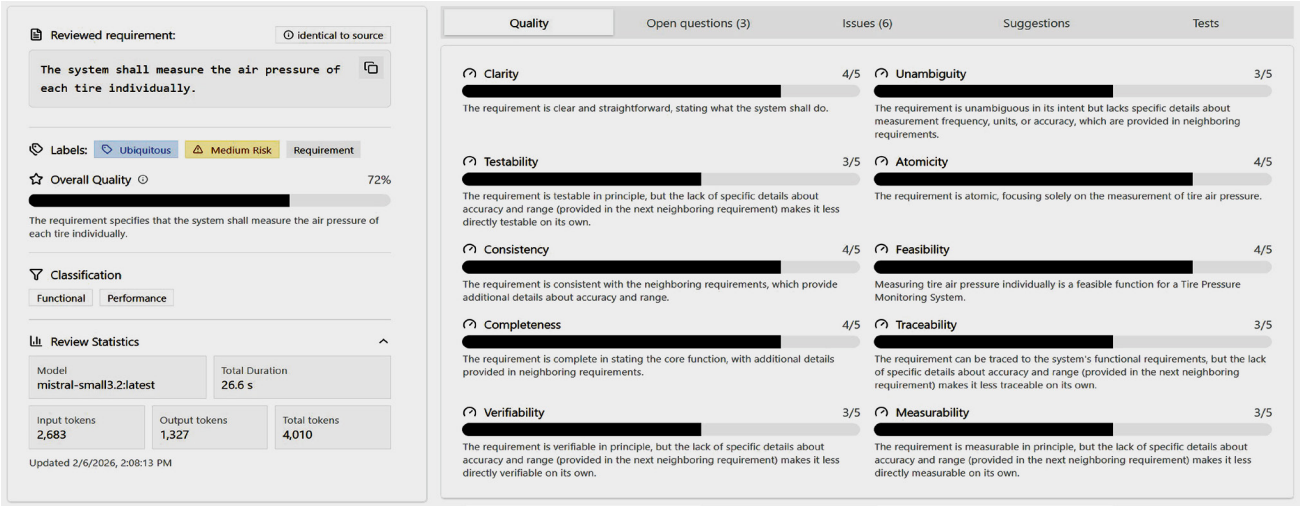


FIGURE 3 Requirement validation with AI in the Mobility WebApp: results of a quality assessment; the requirements are automatically imported from PREEvision and checked against ten quality criteria (© Cognizant Mobility)

– *Tests* – The AI-based creation of the test case specification forms the starting point of test automation, **FIGURE 5**. Using few-shot prompts following an if-then pattern, a natural-language test case specification list is generated. From this, the AI produces numerous positive, negative, and often unexpected corner-case tests. These results must be carefully validated for plausibility and completeness.

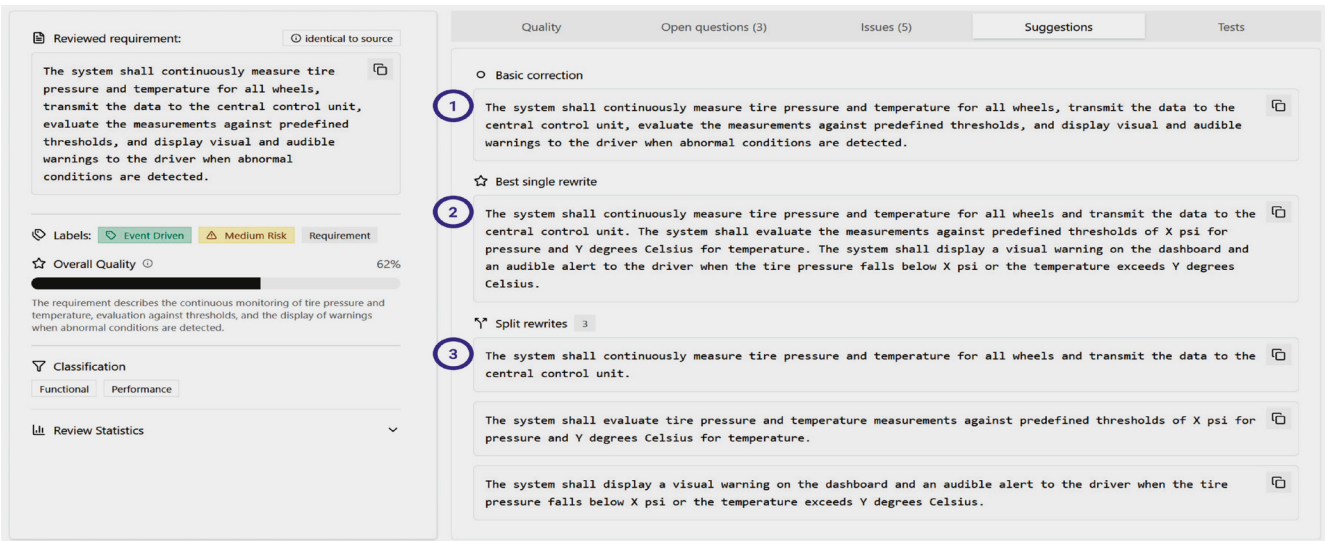
The necessary information, such as message catalogues, models or existing test scripts is provided to the LLM. For generating the test automation scripts, the AI must be given context about the test framework, libraries, signals, CAN IDs, examples, coding guidelines, and so on. Otherwise, it will produce code that cannot be executed. This also reveals how good the quality of the preliminary work was. The higher the input quality, the better the resulting scripts.

ANALYSIS AND REPORTING

To maximize the utilization of expensive test systems such as Hardware-in-the-Loop (HiL), automated test runs are routinely executed outside normal working hours. A previous ATZelectronics article [5] demonstrated how such long-duration runs can be operated reliably and identified initial AI potentials in evaluation, at that time primarily using classical, partly unsupervised learning methods. Although methods have evolved since then, LLMs are not the primary tool at this stage.

A core problem of highly automated test drives is the flood of error messages, which often trace back to only a few, sometimes already known, root causes. The resulting data structures are complex lists or trees with information-rich nodes

FIGURE 4 Refining requirements with AI in the Mobility WebApp: improvement suggestions; a non-atomic requirement is split into three atomic ones (© Cognizant Mobility)



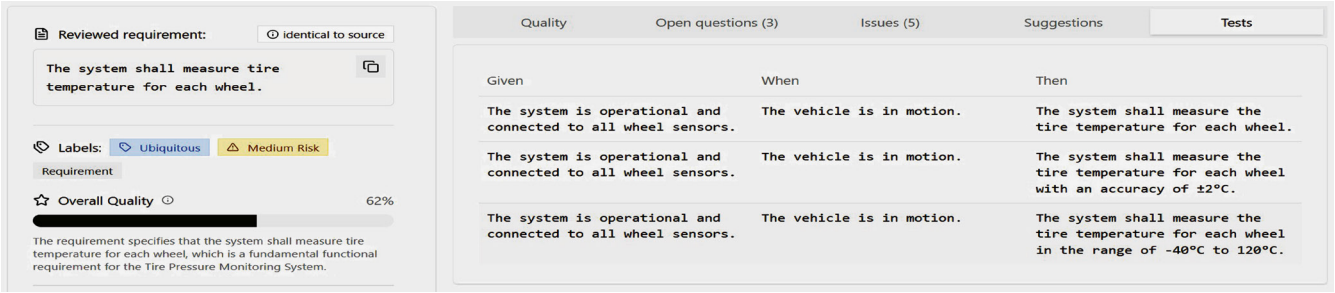


FIGURE 5 Test case specification using AI in the Mobility WebApp: based on the reviewed and improved requirements, specifications for test cases are generated automatically (© Cognizant Mobility)

on which LLMs have, so far, only been moderately effective. Therefore, Cognizant Mobility relies on graph similarity methods to detect structural relationships and combines them only in a second step with semantic analysis of the node texts – an area where LLMs can once again play to their strengths. This has proven successful in practical testing, **FIGURE 6**.

SUMMARY AND OUTLOOK

It was shown which potential AI can already unlock within a classical testing. Its strengths are particularly evident in the refinement of wording, error corrections, and the generation of technical artifacts. At the same time, quality assessments and complex analyses remain unreliable and require human review.

Despite this, user feedback highlights the strong potential of AI. It's use shifts the role of engineers: instead of creating large quantities of test cases or specifications, they increasingly orchestrate digital tools. The core competencies will lie in formulating precise specifications and critically reviewing AI-generated results. AI provides speed and volume, while humans contribute vision, contextual understanding, and quality assurance, particularly in safety-critical domains such as ISO 26262, where clear boundaries between automated assistance and human accountability remain essential.

To integrate AI sustainably and safely, data protection, tool landscapes, process integration, and ROI must be continuously evaluated. Solutions such as RAG, local models, or isolated corporate clouds can help incorporate information in a controlled manner and produce traceable results. AI thus becomes a permanent assisting tool in systems engineering. Cognizant Mobility is expanding the use of AI in the MBSE and testing environment without compromising system quality. Current areas of focus include regulatory analyses using RAG and vector databases as well as spec-driven development, MBSE chatbot agents that retrieve information on demand [6] and system analyses to identify contradictions between subsystems.

The coming years will fundamentally transform embedded software testing. The IT sector shows a clear trend toward spec-driven development and many signs suggest that embedded development will follow this path. MBSE is justified in many respects without any alternative. However, extending it with functional and non-functional requirements that are captured in a highly precise, consistent specification – one that becomes the central artifact in embedded development from which software, tests, and analyses are automatically derived – will be key. AI accelerates and supports this shift by sharpening specifications and directly generating technical artifacts from them. Time will tell whether this transformation will unfold over decades or perhaps only over a few years.

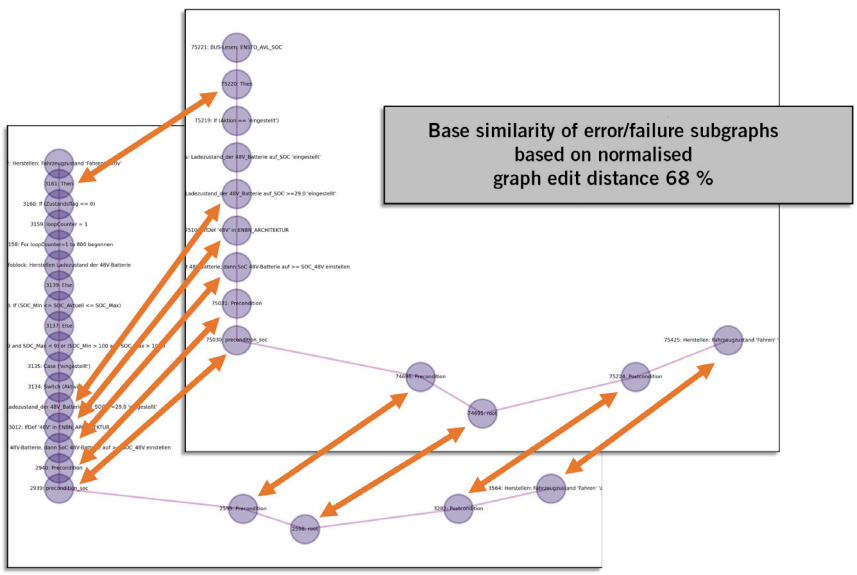


FIGURE 6 Automated analysis of test errors/failures: to determine the degree of similarity between test errors/failures subgraphs of failed test steps are compared primarily based on a normalized graph similarity measure (© Cognizant Mobility)

REFERENCES

[1] OMG Standards Development Organization (ed.): About the OMG Systems Modeling Language Specification Version 1.0. Online: https://www.omg.org/spec/SysML/1.0/, access: March 3, 2026

[2] Incose (ed.): Systems Engineering Vision 2023. Online: https://sdincose.org/wp-content/uploads/2011/12/SEVision2020_20071003_v2_03.pdf, access: March 3, 2026

[3] Kriebel, S. et al.: Improving Model-based Testing in Automotive Software Engineering. In ACM/IEEE 40th International Conference on Software Engineering: Software Engineering in Practice, 2018. Online: https://www.researchgate.net/publication/325728663_Improving_model-based_testing_in_automotive_software_engineering, access: March 3, 2026

[4] Austrian Testing Board, German Testing Board e. V., Swiss Testing Board (ed.): ISTQB Certified Tester Lehrplan Foundation Level 4.0.1a, German edition. Online: www.gtb.de/wp-content/uploads/2024/02/ISTQB_CTFL_Lehrplan-2023_V4.0.1a_DE.pdf, access: March 3, 2026

[5] Brandlmeier, A. et al.: KI-basiertes Testing und Robot Process Automation. In: ATZelektronik, 16 (2021), No. 1, pp. 52-54

[6] Zhang, W. et al.: MBSE Co-Pilot: A Research Roadmap. In: Systems Engineering 29 (2025), No. 1. Online: https://incose.onlinelibrary.wiley.com/doi/10.1002/sys.70011, access: March 3, 2026

IMPRINT:
Special Edition 2026 in cooperation with Cognizant Mobility GmbH, Ingolstädter Str. 45, 80807 München; Springer Fachmedien Wiesbaden GmbH, Postfach 1546, 65173 Wiesbaden, Amtsgericht Wiesbaden, HRB 9754, USt-IdNr. DE81148419

MANAGING DIRECTORS:
Stefanie Burgmaier | Andreas Funk | Alexandra Dambeck

PROJECT MANAGEMENT: Anja Trabusch

COVER PHOTO: © Cognizant Mobility

Cognizant Mobility

Michael Pollner
Dipl.-Inf., MBA
Head of Business Development

M +49 151 44 15 2348
E michael.pollner@cognizant.com