Qt Quality Assurance

WHITEPAPER

Software Quality for Industrial Vehicles

Ensuring Excellence When Developing Software for Construction, Mining, Goods- and Warehouse-Handling, Forestry and Agriculture

The industrial vehicle market is a dynamic and rapidly growing sector. This growth is driven by technological advancements, which come with new challenges: Especially in safety & security-critical environments comprehensive analysis and testing of software is required to minimize the risks of failure.

The Benefits of Analysis and Testing — Quality Assurance in Software Development for Industrial Vehicles

Create high-quality software, which meets the strictest industry standards and your customers' expectations.

Quality assurance and software testing are vital for ensuring the safety, reliability, and efficiency of industrial vehicles. Neglecting these processes can lead to severe consequences, including accidents, operational disruptions, and in some cases even legal issues.

Safety & Security

Quality assurance and software testing are paramount when developing software solutions for industrial vehicles. They ensure that the software controlling these vehicles is reliable, safe, and efficient. Industrial vehicles, such as those used in mining, warehousing, and agriculture, often operate in hazardous environments. Ensuring that the software governing these vehicles functions correctly is crucial to prevent accidents and protect operators' lives.

It therefore does not come as a surprise that this industry is highly regulated. Adhering to industry standards and regulations, such as ISO 26262, ensures that the vehicles meet the necessary safety and functional requirements. By applying quality assurance processes manufacturers can demonstrate compliance and avoid legal issues.

Reliability and Efficiency

High-quality software also ensures that industrial vehicles operate smoothly and efficiently, minimizing downtime and maintenance costs. This is particularly important in industries where time is money, and any disruption can have significant financial implications. As example: Downtime can disrupt operations in warehouses, delaying shipments and causing financial losses and lead to a negative brand image.

Economical Advantage

Optimized code quality also ensures the software is easy to adapt to changing needs. With the right tools in place, bugs can be detected at an early stage and easily be fixed. Automating these processes brings the added value of freeing the time of your development and test engineers and producing more reliable analysis and testing results. Fewer resources for updates or even refactoring result in lower development costs, better time-to-market, longer product life-cycles and ultimately, a higher return on investment (ROI).

The Importance of Static Code Analysis

A powerful tool to enhance code quality, security, performance, and architectural integrity.

Static code analysis is crucial for identifying potential issues in the code without needing to execute it. For industrial vehicles, where safety and reliability are paramount, static code analysis offers significant benefits. It enables early detection of defects, identifying coding violations, dead code, and other issues early in the development cycle, thus reducing the risk of bugs making it to the production stage. Moreover, adherence to coding standards is a critical aspect. Static code analysis helps ensure compliance with industry standards and coding guidelines, such as ISO 26262, which is essential for maintaining functional safety in industrial vehicles. Continuously analyzing the codebase with these tools helps maintain a high standard of code quality, preventing issues like code bloat and technical debt.

Coding standards are essential for maintaining a uniform codebase, especially in large projects with multiple developers. By ensuring that everyone adheres to the same guidelines, the code becomes easier to maintain. This can be achieved through static code analysis. Another critical benefit is the early detection of security vulnerabilities. In today's world, where cyber threats are increasingly sophisticated, ensuring the security of software is paramount. Static code analysis can identify potential security risks such as buffer overflows, injection vulnerabilities, and other common coding issues that could be exploited by attackers. By addressing these vulnerabilities early, developers can significantly reduce the risk of security breaches and protect sensitive data.

Additionally, architecture verification is another crucial aspect that static code analysis can address. Software architecture defines the high-level structure of the system, including its components and their interactions. Ensuring that the implementation adheres to the intended architecture is vital for the system's reliability and scalability. Sophisticated analysis tools are capable of verifying architectural constraints, ensuring that the code follows the designed patterns and dependencies. This verification helps in maintaining the system's integrity, preventing architectural erosion, and supporting long-term maintainability.

Applying automated architecture and static code analysis also optimizes the overall efficiency of development teams. They can focus on developing the code and refactor critical segments of the code to improve the quality of the software. This is particularly important in industries where performance is critical, such as in real-time systems.

The Importance of Automated GUI Testing

Consistency and Reliability

Time Efficiency

Comprehensive Coverage

In the development of software for industrial vehicles, the graphical user interface (GUI) is a critical component, as it directly affects user interaction and experience. Automated GUI testing ensures that the interface behaves as expected across different platforms and use cases.

By automating GUI tests, developers can consistently replicate tests across various iterations, ensuring that changes or updates do not introduce new issues. Manual GUI testing is labor-intensive and time-consuming. Automation speeds up the testing process, allowing for quicker detection and resolution of issues. Automated tests can also cover a wide range of scenarios, including edge cases that might be overlooked during manual testing, thus enhancing the overall quality of the software.

Automated GUI testing is also crucial for cross-platform compatibility, ensuring that the user experience remains consistent across different devices and operating systems. This is particularly important for industrial vehicles, which may use a range of hardware and software configurations. By employing automated testing, developers can quickly identify and fix issues specific to certain platforms, preventing potential malfunctions that could impact the safety and efficiency of the vehicles.

Additionally, regression testing is another significant benefit of automated GUI testing. Whenever new features are added or existing ones are modified, there's a risk that these changes could inadvertently affect other parts of the software. Automated tests can be re-run after each update to verify that previous functionalities are still working as expected. This continuous validation process helps maintain the integrity of the software over time, reducing the likelihood of bugs slipping through and causing disruptions in real-world applications.

Finally, automated GUI testing allows for scalability in the development process. As industrial vehicle software becomes more complex, the number of test cases required to ensure its reliability can grow exponentially. Automated testing frameworks can handle this increase in test cases without a corresponding rise in manual effort, enabling development teams to keep pace with expanding project scopes and tight deadlines. This scalability ensures that quality assurance processes remain robust and efficient, even as the software evolves.

Future-proof the Development of Your Software for Industrial Vehicle Solutions

We offer an extensive range of solutions to help you create better software. From advanced architecture verification and static code analysis tools to code coverage and automated GUI testing. All tools seamlessly integrate into existing development environments and will improve the quality of the code. Tool qualification kits are available for additional safety and security.

Axivion Architecture Verification

Axivion ensures the software architecture aligns with the code, preventing architecture erosion and maintaining system coherence. It not only features automated software architecture checks, it also supports architecture recovery and architecture archaeology.

Even compliance with a safety architecture, the coexistence of functions with different ASIL classifications, to show the independent software elements and their interfaces can be achieved with Axivion. This is the basis for freedom from interference and assures functional safety of the software.

Learn more: **www.qt.io/axivion-av**

Axivion Static Code Analysis

Axivion detects coding violations, clones, dead code, and other defects early in the development process, reducing time and costs. The automated analysis effectively reduces technical debt and makes it easier to maintain the software. This significantly increases the life-time of the software and ultimately, the return on investment.

Axivion Static Code Analysis is certified by SGS-TÜV Saar GmbH as suitable for use in the development of safety systems up to the highest level of the safety requirement contained in standards such as ISO 26262.

Learn more: **www.qt.io/axivion-sca**

Squish

Squish automates GUI testing to ensure the user interface behaves as expected across various platforms and technologies. By automating GUI testing, the growing complexity of software is no threat to safety and security. Squish is a powerful tool with various features for:

- Ease of Test Creation and Maintenace
- Support for Visual and Functional Testing
- Integration with CI/CD Pipelines
- Scalability and Parallel Execution

Learn more: **www.qt.io/squish**

Test Center

Test Center connects test automation with the entire development process and offers advanced analytics and reporting. View test suite statistics, visualize trends and analyze historical data of test executions with built-in, automatic statistical reporting of your imported data. Test Center was built from the beginning to be a central, lightweight web database – access it from any device, at anytime — for instant analysis of your pipeline projects.

Learn more: **www.qt.io/test-center**

Coco

Coco provides code coverage analysis and supports various coverage levels, such as function coverage, statement coverage or MC/DC to identify untested code, guide testing efforts and improve the overall code quality. Test data generation reaches the highest feasible coverage level faster, sorts out redundancies in code coverage analysis, and gets automatic collections of test data that includes edge cases.

Learn more: **www.qt.io/coco**

Free Demo or Evaluation

Seeing is believing. If you want to learn more about our products, have a specific use case you would like to discuss or want to try the tools yourself — our experts for industrial vehicles software will be happy to help.

Contact us

www.qt.io/quality-assurance