Delivering safer cars, faster: automation and continuous delivery in the auto industry

By **Andreas Dharmawan**

Some cars today boast more than 300 million lines of code – and growing. Software is increasingly becoming more prevalent – and critical – in today's automobiles, as cars become smarter, more connected, and more complex.

Unfortunately, vehicle recalls due to software glitches have been on the news lately, with some caused by serious software malfunctions that may have put the driver and passengers' safety at risk.

The automotive industry has always been bound by strict safety regulatory certifications and compliance rules, designed to ensure the safety of the vehicle and the software embedded in it. However, despite industry efforts to put best practices in place, the rate of issues and software failures has been increasing.

This article considers the key regulations and compliance guidelines that the automotive industry is required to meet, and the challenges faced by both vehicle manufacturers and suppliers when trying to implement and enforce safety and security practices in order to meet these regulations. It will also attempt to explain how Continuous Delivery and automation solutions are helping automotive manufacturers to address these challenges while improving the overall quality of the vehicle and its underlying software and electronics system.

MISRA

The majority of embedded software systems – such as those embedded in cars – are written in C or C++. MISRA C is a set of software development guidelines for the C programming language developed by MISRA (Motor Industry Software Reliability Association), aiming to facilitate code safety, portability and reliability in the context of embedded systems.

PRQ, Klocwork, Coverity, and other static analysis tools support MISRA compliance testing. These tools scan the code to identify bad practices or exceptions that do not meet the guidelines specified by MISRA.

Challenges with MISRA Compliance

Executing a static analysis on the code base is very time consuming and requires numerous manual hand-offs amongst team members. It is common for companies in the automotive industry to have a dedicated engineer to handle code analysis for MISRA compliance using a specialised tool. Usually, team members who want to check in code to the main trunk must first put their code into a branch in the source code management tool, create a ticket with IT to request the specialised engineer to analyse their code and perform a set of tests and verifications prior to check-in.

In addition to the compliance engineer being a bottleneck in the process, the number of licenses for the code analysis tool is often restricted as well, with team members having to "get in the queue" and wait to use the floating licenses. This creates a series of bottlenecks in dev and test cycles during component testing, integration testing, and verification testing.

These bottlenecks hinder the feedback loops in dev and test and slow the cycle times. Because of this, test coverage during component testing often remains low, with some areas of the code not being tested for MISRA compliance (or other compliance frameworks), or that are not tested often enough – as engineer try to minimise wait time to accelerate the pipeline. This leads to bugs being discovered later on in the process: during integration testing, or – worse – when the vehicles are already on the market.

Not only do today's cars incorporate numerous software applications, they also integrate a lot of components manufactured by different vendors and suppliers. Those (as well as the manufacturer) are often being penalised for not meeting compliance requirements or for introducing bugs, or delaying releases. The later in the process that software bugs are found, the more costly they are to fix, and the more impact each bug in a specific component has on delaying the schedule of other integrated components, and delaying the release date of the manufacturer.

How does automation and Continuous Delivery help with MISRA?

Automating all the various tools, tasks and manual handoffs involved in the software delivery process (from code-check in, through the various testing stages, through deployments and all the way to final release) significantly accelerates the pipeline and increases the quality of the product.

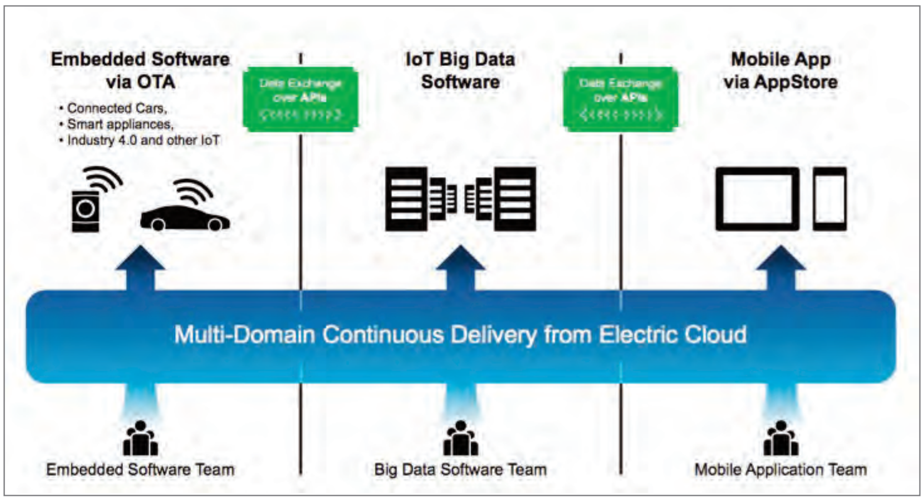
In the case of static analysis tools, execution through an orchestration platform, which automates triggering the test and eliminates manual hand offs that are time consuming and error prone, removes some of the major bottlenecks in the development process.

By automating static analysis, and other tests, the code coverage increases, along with the chances of identifying bugs earlier in the process. In addition, parallelisation of the tests across a cluster of machines speeds up the run time to improve utilisation and shortens idle/wait time. Additionally, jobs can have different priority levels, enabling them to be distributed in 24-hour periods, with the test and integration workflows being executed automatically, around the clock, with no need for human intervention.

With faster static analysis execution, automotive manufacturers are able to have more dev and test cycles to harden the product's quality. Automotive manufacturers using automation and orchestration as part of their software delivery pipeline see 5x to 24x acceleration in cycle times, resulting in reduction of IT expenses by of up to 30%, as well as improved quality and faster time-to-market.

ISO26262

ISO26262 is a functional safety standard that guides automotive development phases, covering specification, design, implementation,



integration, verification, and production release. There are workflow tools that monitor and measure the processes involved in these phases. These tools tend to have templates to measure the recorded processes against compliance requirements such as ISO26262.

Challenges with ISO26262 Compliance

ISO 26262-6:2011 specifies the requirements for product development at the software level for automotive applications. This section covers the a) Software unit design and implementation, b) Software unit testing C) software integration and testing, and d) Verification of software safety requirements. There follows an explanation of how to track and report on compliance to verify that the Unit and Integration design, implementation and testing were carried out in accordance with the guidelines.

For compliance reporting to be comprehensive it requires a tool that automatically captures every single step and input/output of the process, to record every bit that makes its way through the pipeline, and report on it in a meaningful way. While Process tools used by some manufacturers can provide visibility and compliance reports, often they require the data for each step in the development/ production sequence to be entered manually into the system for record keeping. This is both time consuming, error prone, and requires dedicated resources.

Remember that a myriad of point-tools are involved in the software delivery process (in design, build, component, integration, model-in-the-loop, simulation-in-the-loop, hardware-in-the-loop tests, verification, etc.), and each specific automotive component would likely go through hundreds, if not thousands, of tasks and workflows as part of the development pipeline.

In this daisy-chain of hundreds of specialised point-tools, the output of each tool becomes the input for the next one in the sequence, and determines the next step in the process. (As a simple example: if a unit test fails – then the code needs to be fixed, if it passes with certain conditions – the code will be packaged and progress further down the pipeline, to Integration testing).

The challenge with ISO compliance is that it can best be described as 'painful'. It necessitates the engineers to not only develop the code, manage these point-tools, review their output to determine the next action to be taken – but also, keep painstakingly detailed records and enter in the Process tool each one of those input/output along the workflows and the steps being taken. And by whom, at what time, on which environment, for what reason, under whose approval, and so on…

The manual nature of this process not only slows down the process considerably, but often results in compliance data that is inaccurate or missing. When the project is behind schedule, engineers may abandon the data-entry process, and compliance concerns take a back seat. The compliance data is now either out of synch or not collected at all. After the release is complete, the organisation may attempt some 'forensic' efforts to trace back the required data for each step in the workflow, but more often than not this data would have been lost by now, or is considerably lacking.

How does automation help with ISO26262?

An end-to-end Continuous Delivery (CD) solution automates and orchestrates the execution of all the various tools used in the different phases of software development defined in ISO26262-6:2011 – including software implementation, unit testing, and integration and testing.

With a single orchestration platform that automatically triggers the tasks along the pipeline, and automatically records the input/output and all related attributes for each task in the workflow – compliance becomes painless, with automation essentially guaranteeing auditability.

Developers and testers only have to trigger the process they would like to execute and let the CD solution perform the job for them and trigger all consequent steps in the workflow. For example, when a developer is about to check-in their code, the platform can automatically trigger a battery of preflight and unit tests, using dozens of tools, and execute downstream processes depending on the results. Not only does the pipeline progresses without human intervention, but a complete, real-time, log is automatically recorded from each point tool and from each task, in a centralised location. This data can later be packaged with a click of a button as a compliance document in any format (with pre-set templates, such as for Aerospace & Defense or Automotive industries), or feed into higher-level workflow tools that cover areas beyond the software delivery lifecycle. Automotive manufacturers using a CD solution to enable compliance and auditability see a reduction in the efforts having to go into producing compliance reports from "person weeks" to minutes (!), which translates to millions of dollars in savings annually.

SPICE (ISO/IEC 15504)

SPICE is a set of technical standards for software development process and related business management functions. It defines the maturity of an organisation in six levels: (0) incomplete process, (1) performed process, (2) managed process, (3) established process, (4) predictable process, and (5) optimising process.

Challenges

Many organisations have established repeatable processes. But they do not have an automatic way to measure the process performance or catch and prevent process violations. Often, the SPICE compliance audit is performed after the fact and is based on unreliable or incomplete data. This manual process of collecting data after the product is released is time consuming and inaccurate, and often results in a report that does more harm than good, since the organisation is not discovering areas for improvement.

Additionally, every team member, whether a developer, a tester or a release engineer, dreads being asked to participate in data collection for compliance, and they often try to avoid it, as this process "gets in the way" of their actual job.

How does Automation help with SPICE?

Similarly to ISO26262, a CD solution accelerates SPICE compliance processes by automatically gathering compliance data from all tasks and tools involved in the end-to-end process.

Automotive manufacturers using a CD solution easily meet SPICE compliance level 3 (established process) because the process is recorded automatically in a centralised location and everyone has visibility into the data.

Because the process execution data is always collected, the CD solution can chart the performance of the different teams, pass/fail trends, and many other KPIs to monitor and ensure the health of the business processes. After running several projects with a CD solution, the system has average performance data across many different processes, which allows the organisation to meet SPICE level 4 (predictable process). This data can then be used to develop additional processes that provide real-time feedback loop towards realising level 5 (optimising process).

Summary

The automotive industry has developed and agreed on standards and guidelines for automotive software coding and development processes. However, the rate of software malfunctions in today's cars is continually increasing.

The challenges lie with the heavily manual, time-consuming and error-prone processes necessary to adhere to these standards and guidelines. Given how competitive the automotive industry is today, automotive manufacturers and their suppliers do not have enough time to follow the required compliance standards and implement the recommended best practices for automotive software development processes.

The necessary standards and guidelines are available but the implementations and tracking are spotty and incomplete. Automating the software delivery process and implementing an end-to-end Continuous Delivery solution help eliminate the bottlenecks caused by these manual, slow, error-prone processes.

The orchestration and acceleration enables manufacturers servicing the automotive industry to greatly increase their dev and test iterations and find bugs much earlier in the cycle so they have plenty of time to resolve them. This improves overall product quality (and the safety of all of us, drivers!), reduces IT expenses, and increases time to market.

***Andreas Dharmawan** is VP of Technical Field Operation at Electric Cloud*