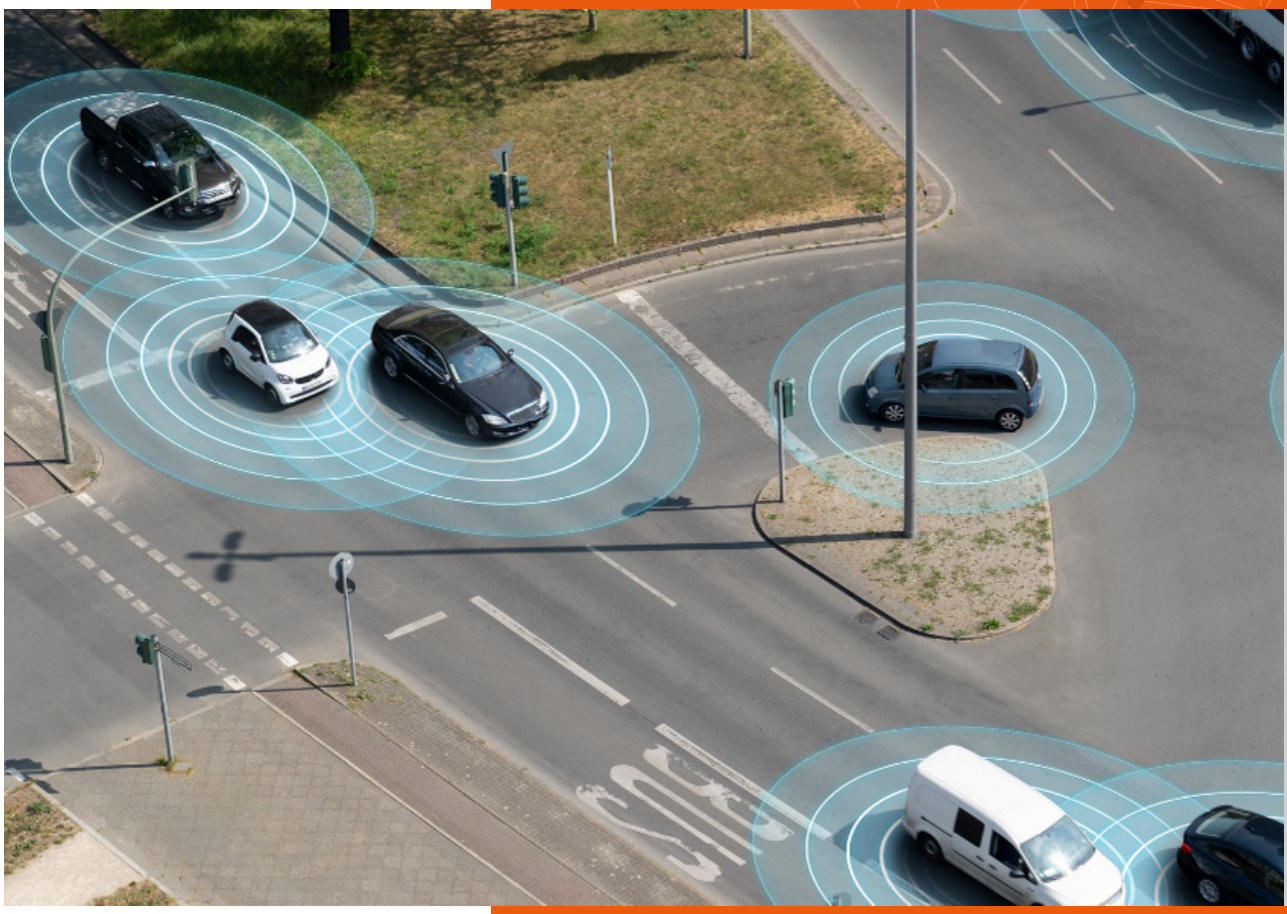
TRUST IN SOFT

Automotive Industry
White Paper

April 2022



TRUSTINSOFT

ON THE ROAD TO ZERO BUG VEHICLES

HOW EXHAUSTIVE STATIC ANALYSIS CAN HELP MANAGE AUTOMOTIVE SOFTWARE CHALLENGES OVER THE NEXT 10 YEARS.

A NEW ERA FOR VEHICLES

Software is rapidly transforming the automotive industry.

Over the next 10 years, expansion of the software and electrical/electronics market in the sector is expected to far outpace the automotive market as a whole. As a consequence, automotive companies and their executives have become focused on software and electronics. In their quest to realize that projected growth, however, they will face numerous challenges.

The sector's four biggest trends—autonomous vehicles,connectivity, powertrain electrification, and shared mobility—all rely heavily on software. Automotive players need to adapt their processes and tools and include approaches like agile practices, continuous integration and automated testing to compete with the industry's software White Paper: On the road to zero bug vehicles 2 development leaders.

To face this new challenge, developers in the automotive industry must consider drastically supercharging the verification and validation process of software, through enhanced testing and analysis of the source code to mitigate errors.

A recent report by McKinsey and Company warns that as the landscape shifts, automakers who lack sufficient software capabilities will face major risks, including start-of production delays and budget overruns. They may fall far behind competitors and new entrants who can bring more innovative products to market faster. Software issues could lead to massive recalls and leave companies vulnerable to customer-safety risks resulting from hacking attacks. [1]

In this white paper, we'll look closely at the key, software-driven trends expected to dominate the automotive sector over the next decade. We'll also examine the pressing challenges the industry will have to overcome to realize this expected and necessary software growth.

Finally, we'll look at an emerging technology, called exhaustive static analysis, that promises to help companies adapt to this changing automotive landscape.

We'll examine how this new tool can help automotive OEMs and their suppliers transform their software development toolchains to support automated testing and greatly improve their productivity. We'll look at how it can ease the transition to agile development and continuous integration practices. Most importantly, we'll see how it can help automotive companies guarantee their software is safe, reliable, and secure.

ON THE ROAD TO ZERO BUG VEHICLES

Software now drives the auto industry

Software is now the primary driver of economic growth in the automotive industry.

According to McKinsey & Company, rapid expansion of the software and electrical/electronics market will result in a compound annual growth rate (CAGR) of 12 percent from 2020 to 2030. Areas of strongest growth include software functions (with a CAGR of 11 percent) as well as integration testing (12 percent). At this rate, the SW and E/E market is expected to vastly outpace growth in the overall automotive market, which is estimated to grow at 3 percent CAGR in the same time span. [2]

In the next ten years, software will become a strategic development area for the automotive sector and will continue to transform the industry.

The four dominant disruptors in the industry—AD, connectivity, powertrain electrification and shared mobility (known collectively as ACES)—are already mutually reinforcing this transformation, according to McKinsey, with further disruption still to come.

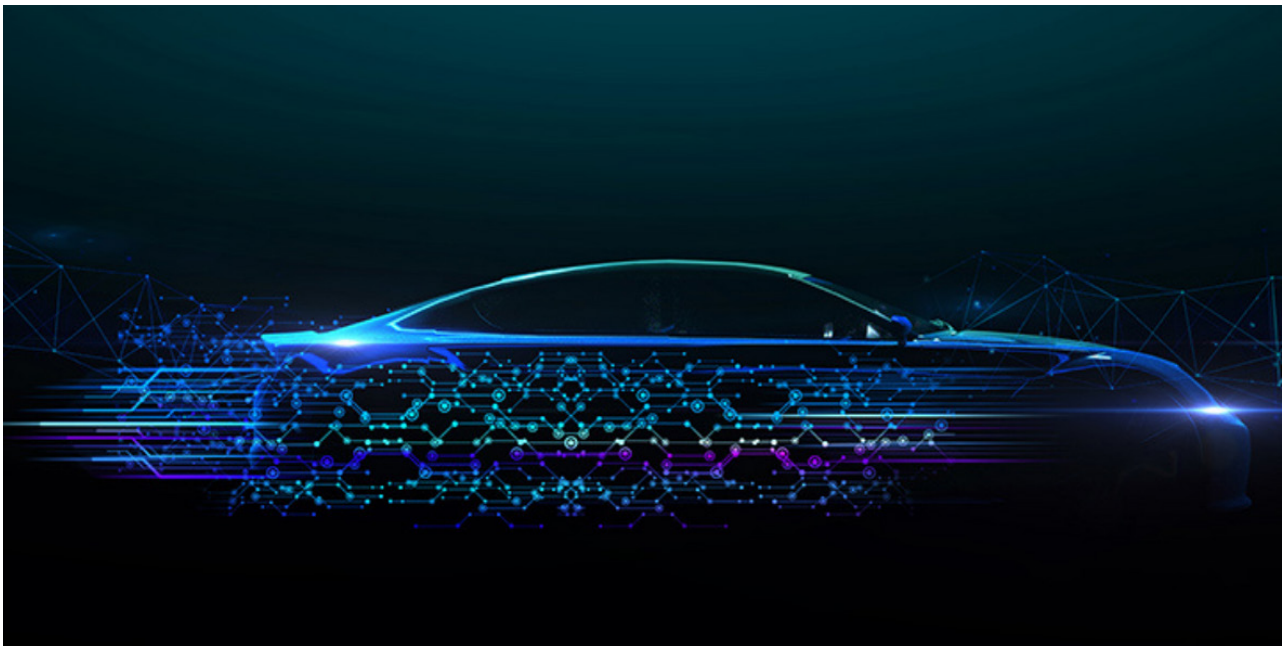
By 2030, AD SW alone—excluding advanced driver assistance systems (ADAS)—should represent more than a quarter of the market value and grow at an average yearly rate of over 20 percent until 2030. Elements like sensor fusion and environmental modeling will become more complex as AD enabled vehicles will need to handle real-world situations. [3]

McKinsey says three key factors will contribute to strong overall growth of the SW market:

- Customization effort for integrating functions in a growing number of platforms
- Growing labor costs for SW developers
- Increase in SW complexity in the domains most influenced by ACES trends [4]

Growing software complexity, however, will become a severe obstacle to productivity, if not managed properly.

> ***Growing software complexity, however will become a severe obstacle to productivity if not managed properly.***



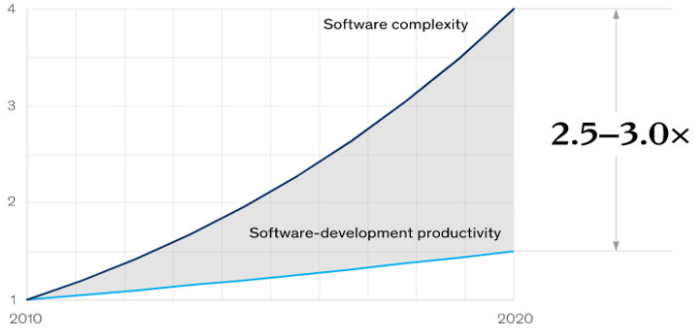
ON THE ROAD TO ZERO BUG VEHICLES

Software Complexity a Major Concern

According to McKinsey's research, software complexity grew by a factor of 4.0 over the past ten years, while software development productivity increased by only a factor of 1.0 to 1.5. [5] Today's average car contains one hundred million lines of code. That's much more than the code found in the Large Hadron Collider, and that of the world's most complex fighter aircraft, the F-35. [6] The code in vehicles is often distributed across over a hundred microcontrollers, or MCUs, many of them interconnected. "Distributing 70 of the devices (MCUs) throughout the car is quite common these days, (and) better-appointed cars can have more than 100," writes Bertel Schmitt on the automotive culture website, The Drive. [7]

Growth in software complexity more than doubles the growth in software-development productivity.

Relative growth over time, for automotive features, indexed, 1 = 2010



Source: Numetrics

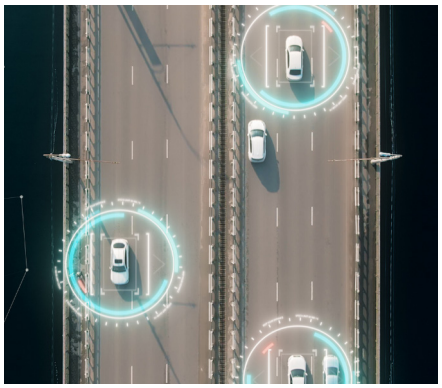
This trend is expected to shift gears over the next decade and beyond, as the industry moves towards centralization of ECUs and MCUs in order to keep up with new functionalities and greater connectivity. As it does, it will stress industry players and force them to increase their software development productivity, because of the increase in complexity that will accompany this centralization. "Software maintenance alone will rapidly use up all software R&D resources if complexity continues to grow while productivity remains unchanged, leaving little room for innovation," says McKinsey. "Ultimately, the complexity-productivity gap will reduce cost competitiveness and could lead to severe financial and reputational problems." [8]

Disparity in how organizations manage this growing complexity is creating wide gaps in competitiveness between industry players. "Organizations in the top quartile for software development achieve 3.0 times greater productivity compared with bottom-tier players, 3.5 times more throughput, and 6.0 times better quality," reports McKinsey. "Consequently, their time to market is shorter and development costs are lower for each level of new software functionality." [9]

> ***Increasing software complexity and volume combined with demand for continuous feature updates will force OEMs to find and resolve coding errors as quickly as possible. Failing to do so may create massive backlogs of bugs that overload their resources and complicate error tracing. This in turn would increase verification costs and cause development and delivery delays.***

ON THE ROAD TO ZERO BUG VEHICLES

The 4 Main Challenges



1. The Autonomous Driving Challenge

Much of the continued increase in complexity will arise from advancements in autonomous driving, which is heavily dependent of software development.

"Growth in SW and sensors is largely driven by the development and adoption of AD, requiring advanced SW functionality (e.g., object detection and classification based on neural networks, raw data sensor fusion, and environmental modelling as well as algorithms for path planning), increased functional safety, and new sensor types (especially light detection and ranging, LiDAR)," says McKinsey. [10]

A large portion of this advanced AD SW functionality will be safety critical. This software will face greater scrutiny to ensure safety and compliance with regulations like ISO 26262. This, in turn will require efficient, effective defect detection and removal, as well as verification of strict adherence to requirements.

2. The Connectivity Challenge

Another major driver of software growth in the automotive industry will be the need for connectivity. Infotainment and other connected services, including the capability to update vehicle software components, will drive demand for greater cybersecurity in vehicles over the next ten years.

Therefore, any code related to connectivity must be free of software errors and other vulnerabilities. Such anomalies could allow hackers access to safety-critical functionality, exposing OEMs and suppliers to costly recalls, liability and reputational damage.

As with safety-critical software, automotive players in the connectivity domain will need to adopt technologies and methodologies that provide guarantees that their software is impervious to cyberattack and comply with regulations like ISO 21434.

3. The Verification and Validation Challenge

McKinsey projects validation and verification will constitute approximately 29 percent (USD 24 billion) of the total automotive SW market by 2030. [11] This sizeable market share will be driven primarily by the need to verify safety-critical software (AD, ADAS, security) and the operating systems deployed in the vehicles.

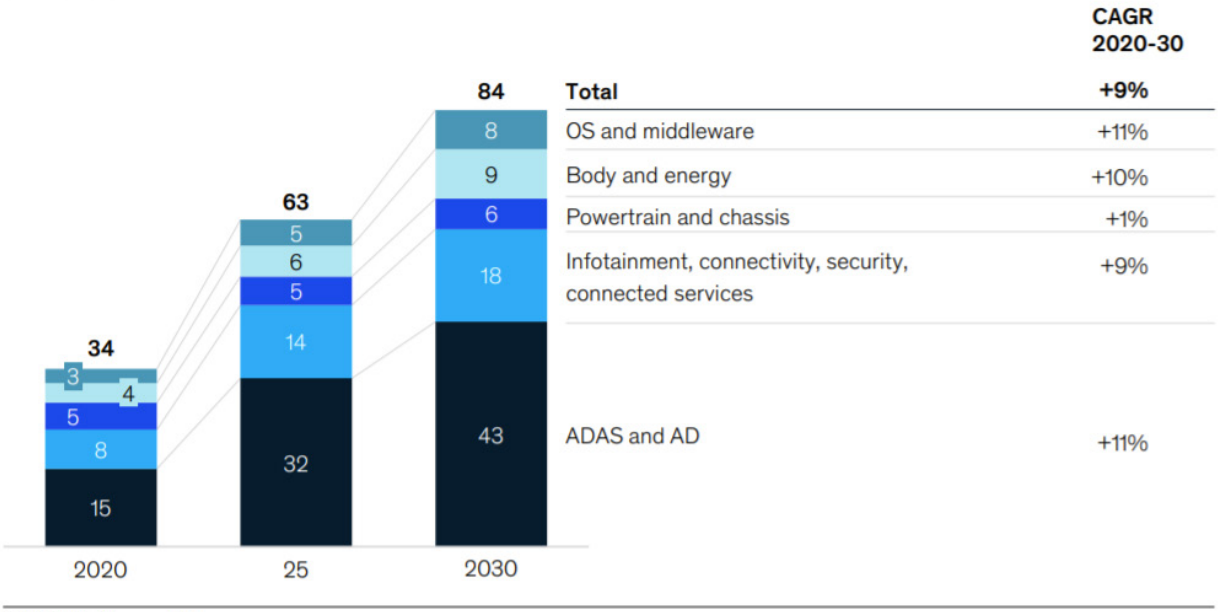
"The implication for automotive players is that they need to further invest in their capabilities to test and validate SW efficiently," says McKinsey. "This translates to adoption of more specialized tools and restructuring of teams to foster efficient validation and collaboration with HW teams." [12]

In other words, along with the need to provide the guarantees of safety and security mentioned earlier, automotive OEMs

and suppliers will be compelled to adopt tools that support modern methodologies like continuous integration and rapid debugging.

Breakdown of SW development efforts into domains

USD billions



SOURCE: McKinsey analysis

4. The software skills and toolchains challenge

Unfortunately, the vast majority of automotive companies are not yet equipped to meet the challenges being wrought by these emerging trends. Most lag far behind the market leaders in the acquisition and development of strategic software skills. Plus, as the demand for those skills mushrooms within the industry, automotive companies will be forced to compete more aggressively for development talent with the software industry at large.

To remain competitive, all will have to cultivate in-house skills and strategic partnerships. They will need to build up competencies in coding, agile development, and specialty skills related to the ACES domains.

Finally, to unlock productivity, they will need to build up integrated toolchains that support state-of-the-art development, continuous integration, and test automation.

"Overall, the introduction of a standardized, state-of-the-art development toolchain is a key enabler to unlock 30 to 40 percent of productivity potentials from automated testing and agile methods," says McKinsey.

A strategic solution: exhaustive static analysis

Exhaustive static analysis tools can be a significant, strategic part of the next generation automotive software toolchain solution.

Exhaustive static analysis is a methodology that makes use of formal methods techniques to increase the operational efficiency of development and validation processes. Developed to guarantee the behavior of safety-critical systems, it expands its scope to guarantee software security as well as safety and reliability.

Plugs into your existing development process... and transforms it

What's more, exhaustive static analysis does all this in a way that does not disrupt the normal software development process. It brings the value of mathematical formal methods to software development in a way that's practically invisible to the developer.

With an exhaustive static analysis framework, all the aforementioned selection and application of formal methods are totally transparent to the developer. From a users' perspective, they are simply testing their code in a manner very similar to what they're already accustomed.

The framework expands and automates the testing process. For example, by adding just a couple of lines of code to your test script, you can expand a limited set of test cases to a complete set of test cases covering the complete range of possible values for all input variables.

Again, you don't need a PhD in formal methods to use these tools. Any developer can handle them. Users don't even need to be aware of the method the tool is using. They just express the problem and the tool does the analysis automatically.

Exhaustive static analysis can help automotive companies:

• Deal with the challenges of verifying and validating increasingly complex software

• Transition to and support automated testing and continuous integration practices

• Identify more bugs than with their current development and validation processes

• Guarantee their software is 100% bug-free

• Guarantee their software conforms exactly to its specification

• Provide proof that can be used in ISO 26262 and ISO 21434 certification

Exhaustive static analysis is also a framework where a broad range of formal methods collaborate together. It contains algorithms that choose the right formal method according to the question being asked. And it can switch seamlessly from one formal method to another, depending on the problem it has been asked to solve.

In other words, exhaustive static analysis is a holistic approach, not one of brute force. Rather than trying to solve every problem using a single formal method, the framework has been designed to determine which combination of formal methods is best suited to solving the problem at hand and to apply those methods in the best way possible.

The major benefits of exhaustive static analysis

Using exhaustive static analysis, you can guarantee the absence of coding flaws—like buffer overflows, for example—that hackers exploit. That means it's now possible to end the cat-and-mouse game of hackers finding vulnerabilities and developers scrambling to remove them. With exhaustive static analysis, you can eliminate all such flaws before hackers even have a chance to look for them. In fact, you'll eliminate many flaws conventional tools fail to catch.

It's extremely easy for users to find answers. The framework chooses the right tools for the problem and switches tools on the fly. For example, our own platform, TrustInSoft Analyzer for C/C++, offers several abstract memory models to analyze the software being validated.

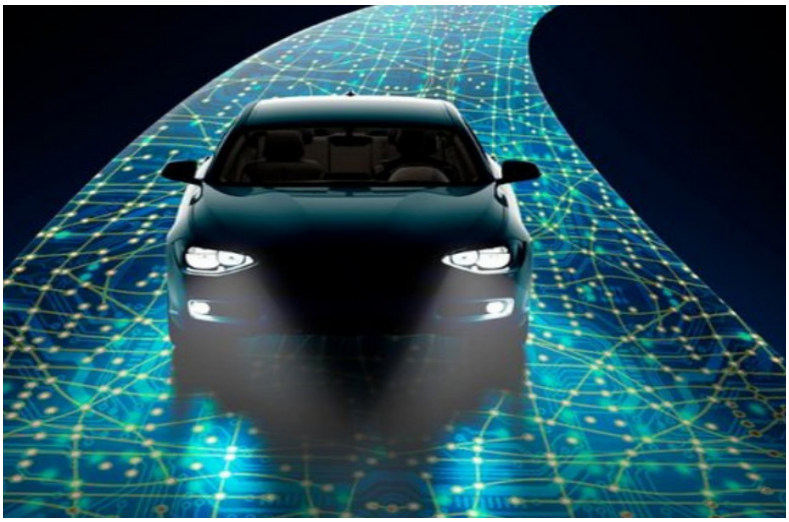
TrustInSoft Analyzer has been designed and developed so that no matter how a C/C++ developer programs, the tool will extract the meaning of the code. The user chooses and applies the best kind of formal method for their objectives and the nature of the software they are verifying.

TrustInSoft Analyzer was even cited in a National Institute of Standards and Technology report to the White House[13] for having demonstrated that it can be used to formally guarantee that no known undefined behaviors (bugs) are present in a system. It can also prove a system behaves exactly according to its specification.

Then, if you want to go a step further, you can guarantee the absence of entire families of vulnerabilities in your code—exactly what the NIST told the White House.

Exhaustive static analysis enables companies to guarantee to customers and regulatory bodies that their applications are completely free of coding flaws and conform exactly to their functional requirements. These guarantees are essential in safety-critical applications, like those for AD, and in security critical applications, like those for connectivity.

One other benefit of exhaustive static analysis: developer satisfaction. Exhaustive static analysis helps developers perfect the applications they've worked so hard to create. It helps them quickly find and eliminate those subtle coding errors that are so easy to make without even being aware of them… and that are so hard to track down after they've been made. It saves developers time. It makes them feel great about what they've built. All these factors can be advantageous in attracting and retaining the top-flight software development talent that will be harder and harder to come by over the next decade.



Code safety & security

Exhaustively find and fix all Undefined Behaviors including the most hidden ones

Ensure absence of crashes and deterministic behavior. Detect 0-days before they are known. Platform specific analysis without compiling.

Boost coverage. Perform quickly the equivalent of billions of tests with 1 generalized inputs test

Determines and propagates the superset of all possible code values in execution paths.

Get mathematical guarantees on software security/safety

Functional proof & absence of Undefined Behaviors (e.g. buffer overflow).

Plus, exhaustive static analysis does all this in an automated, fully-comprehensive manner. In short, exhaustive static analysis finds all the needles in your haystack, in a way that makes your V&V process more efficient.

ON THE ROAD TO ZERO BUG VEHICLES

Supports continuous integration while guaranteeing safety

Tracking down and eliminating undefined behaviors and guaranteeing the safety, security and reliability of an automotive software application is an iterative, cumulative process. The process benefits from taking a step-by-step hierarchical approach, stepping from a basic level of proof to more advanced levels depending on what a given application requires.

LEVEL 1

- Level 1 (testing) is the simplest to use. It is completely automated—as easy to use as ordering a compile of your code. The user doesn't even have to look at the code to use it. Yet, Level 1 will uncover a large portion of the bugs present in the code without yielding any false positives. Level 1 is ideal for use in a continuous integration process. New bugs can be stripped out on a daily basis before they accumulate, without developers having to look for them. Daily use of Level 1 analysis helps prevent verification backlog.

LEVEL 2

- (exhaustive testing) will guarantee that all coding defects have been eliminated from your code. In other words, when you complete Level 2 testing and correction, you'll have no bugs present that can be exploited by hackers or cause unexpected behaviors in your application. Use of Level 2 requires some operator intervention, but it has a very low false positive rate (<10/10,000 LOC on average). TrustInSoft Analyzer's Level 2 analysis provides you with a guarantee of the security of your system if all the confirmed defects are corrected.

LEVEL 3

- Finally, for those applications that need it, there is Level 3 (functional proof). TrustInSoft Analyzer's Level 3 analysis guarantees your code fulfills its specification exactly. It is your best guarantee of safety. Functional proof takes longer and is more costly than defect testing, but for code that needs to be perfect, the return is huge. You'll have a guarantee that your critical application works exactly as specified.

In summary, exhaustive static analysis helps you deal with software complexity by automating your software verification. It eliminates the cat-and-mouse process of finding and fixing bugs through conventional testing. It guarantees safety, reliability and cybersecurity through the use of mathematical formal methods. And finally, it simplifies compliance with ISO 26262 and ISO21434.

Dealing with ISO26262 and ISO21434

ISO 26262 is a functional safety standard used in the automotive industry, adapted from IEC 61508. It includes requirements for software development and design aimed at ensuring the safety of the vehicle at every stage in the vehicle's lifecycle. As mentioned earlier, the volume of software having a direct impact on passenger safety is expected to grow enormously over the next decade and beyond. It's important that automotive software complies with ISO 26262 and that automotive firms are able to provide guaranteed proof that their software does so. Exhaustive static analysis can provide that proof.

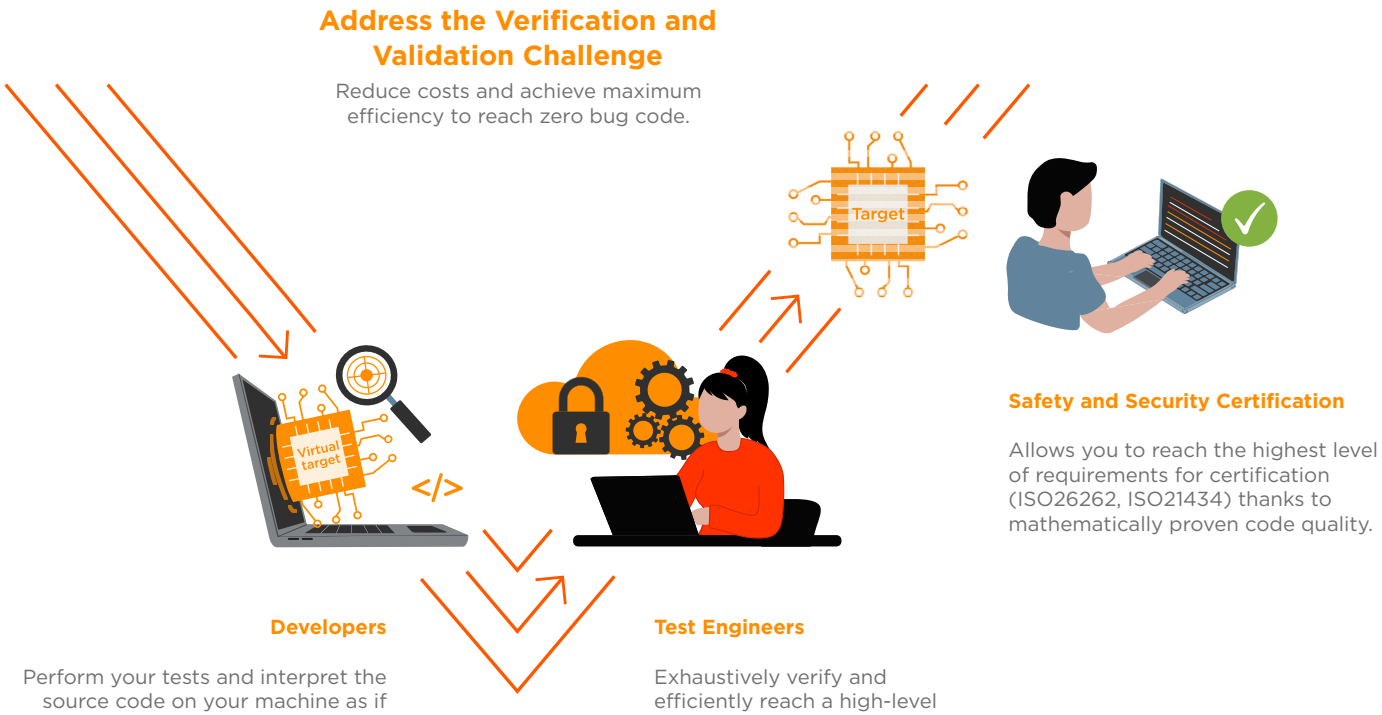
ISO21434, on the other hand, is a new standard (published in August 2021) that concerns the cybersecurity risk of the electronic systems of vehicles. As modern vehicles become more and more connected to the Internet, they also become more vulnerable to cybersecurity exploits.

ISO 21434 encompasses the entire vehicle production process, from design to software development. It introduces a structured approach to ensuring the security of vehicles. Developers of

connected vehicles will need to demonstrate that their vehicle was designed with cybersecurity in mind, from the beginning to the end.

Although these standards cover a wide scope for developing safe and reliable code, exhaustive static analysis focuses on the most difficult part: identifying and helping developers eliminate undefined behaviors. It ensures that, whatever the inputs and the program conditions, the software will react in a deterministic way and is immune from security flaws.

TrustInSoft Analyzer's formal methods methodology verifies the semantics, or the meaning, of the source code, to be sure that no software defects exist. TrustInSoft's tools will reduce your efforts in achieving compliance with ISO 26262 and ISO 21434.



Thanks to TrustInSoft Analyzer, you can ensure software safety and security all throughout the V cycle

How did TrustInSoft help EasyMile secure its autonomous vehicles?

Don't just take our word for it: learn how autonomous vehicles company EasyMile used TrustInSoft Analyzer to develop safe and secure vehicle software.

Who is EasyMile?

EasyMile is a pioneer in driverless technology and smart mobility solutions. The fast-growing start-up, founded in 2014, develops software to automate transportation platforms.

EasyMile's technology is revolutionizing passenger and goods transportation, offering completely new mobility options. Since 2014, the company has developed and deployed more than 210 autonomous mobility projects in 24 countries.

ON THE ROAD TO ZERO BUG VEHICLES

In June 2019, EasyMile became the first autonomous vehicle company ever to be ISO 9001 certified. This certification recognizes the commitment of the company to deliver high quality-level services to partners and customers.

EasyMile's Need

Providing high-quality service is part of EasyMile's working culture. EasyMile requires correct and robust software to develop safety-first autonomous vehicles. To reach this level of exigence, EasyMile needed to formally verify and validate:

- The safety of its critical C embedded software
- The cybersecurity of its C++ telemetry, monitoring, and navigation software

TrustInSoft's Solution

To meet EasyMile's specific and rigorous needs, they used TrustInSoft Analyzer to mathematically guarantee the safety and the cybersecurity of EasyMile's software.

EasyMile used TrustInSoft Analyzer to perform test verification on C++ software, and to verify embedded C critical software. Since the license included 3-day on-premise training, EasyMile's engineers quickly became familiar with the analyzer.

Because EasyMile also wanted to reach the state-of-the-art in the application of formal methods in embedded software, TrustInSoft also provided specific deliverables to help EasyMile attain this level of robustness.

Within this collaboration, TrustInSoft's experts focused on EasyMile specific needs, questions, and issues.

> **"At EasyMile, we are fully aware of the responsibility we have for designing safe and secure autonomous vehicles. Part of this relies on making sure that the code base is bug-free and devoid of flaws used for several types of cyberattacks. TrustInSoft's solution is an obvious answer because it greatly increases the confidence in our code by providing necessary proofs." - Alexandre Hamez, Tech Lead**

Conclusion

Over the next ten years, the automotive industry will undergo a rapid transformation driven by trends in autonomous driving, connectivity, powertrain electrification, and shared mobility (ACES). The result will be rapid growth in the automotive software market, which will make software competencies the single biggest driver of success within the industry.

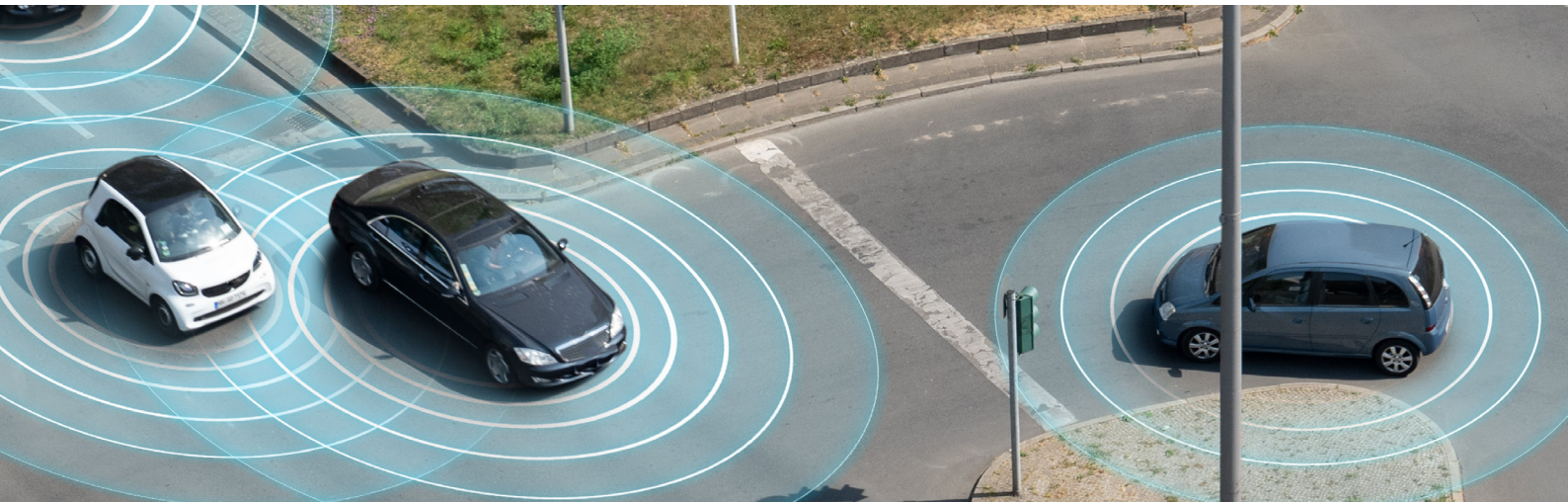
Automotive OEMs and their suppliers will be under extreme pressure to adopt new software development tools and methodologies to improve their productivity. Increased reliance on software within the industry will make it necessary to abandon traditional software testing methods and adopt automated verification strategies and tools that can guarantee safety, security and reliability.

One such methodology, exhaustive static analysis, can provide guarantees that your automotive software application is 100% free of coding defects, impervious to cyberattack, and behaves exactly as specified. Exhaustive static analysis can help automotive industry players adapt to agile development and continuous integration practices, and ensure compliance with ISO 26262, ISO 21434 and other safety and security regulations.

ON THE ROAD TO ZERO BUG VEHICLES

References

1. Burkacky, Ondrej, et al, *When code is king: Mastering automotive software excellence*, McKinsey & Co., October 2021.
2. Burkacky, Ondrej, et al, *Automotive software and electronics 2030: Mapping the sector's future landscape,* McKinsey & Co., July 9, 2019.
3. Ibid.
4. Ibid.
5. Burkacky, Ondrej, et al, *When code is king: Mastering automotive software excellence*, McKinsey & Co., October 2021.
6. Doughty-White, Pearl and Quick, Miriam, *Codebases – Millions of lines of Code*, informationisbeautiful.net. September 2015.
7. Schmitt, Bertel, *Flash! Armies of Microcontrollers in Our Cars Hunger for Memory, and Renesas Feeds Them, thedrive.com*, June 2019.
8. Burkacky, Ondrej, et al, *Automotive software and electronics 2030: Mapping the sector's future landscape*, McKinsey & Co., July 9, 2019.
9. Burkacky, Ondrej, et al, *When code is king: Mastering automotive software excellence*, McKinsey & Co., October 2021.
10. Burkacky, Ondrej, et al, *Automotive software and electronics 2030: Mapping the sector's future landscape*, McKinsey & Co., July 9, 2019.
11. Ibid.
12. Ibid.
13. Black, P.; Badger, L.; Guttman, B.; Fong, E.; *Dramatically Reducing Software Vulnerabilities: Report to the White House Office of Science and Technology Policy; National Institute of Science and Technology (NIST)*, November 2016



TRUST IN SOFT

To learn more about TrustInSoft Analyzer, visit ***trust-in-soft.com/product-c-and-c-source-code-analyzer/.***

If you'd like to speak with a TrustInSoft technical representative about how TrustInSoft Analyzer can meet your organization's specific needs, contact us by email or by phone at:

contact@trust-in-soft.com
Phone : +33 1 84 06 43 91 or +1 (408) 829-5882

About TrustInSoft

TrustInSoft is a French software publisher that provides a C & C++ source code analyzer founded on formal methods, to formally verify software is free from code defects that could affect the safety and security of software. It is headquartered in Paris and serves an international clientele. TrustInSoft currently works with software developers in various market sectors. Leading companies worldwide in the automotive, IoT, telecoms, semiconductors, aeronautics, and defense industries trust our tool to ensure the quality, security and safety of their code.

We empower software developers and testers to guarantee that software behaves in a deterministic way, is resilient to crashes whatever the inputs, is immune from security flaws, while reducing software verification efforts. The company received awards and recognition from the NIST, RSA Conference and the Linux Foundation.