Axel Schneider, Continental Engineering Services GmbH

Using a Modern Software Development Environment to Achieve Maximum Test Coverage

For this interview, we spoke with Axel Schneider from Continental Engineering Services GmbH located in Frankfurt. As Senior Engineer for Virtual Testing & Simulation, he works for the Business Center Driver Assistance and Autonomous Driving. We talked about how a modern development environment combined with virtual test driving helps to make vehicle development more efficient. At the same time, especially with regard to more and more complex software functions, the goal is to ensure the greatest test coverage possible.

We are very pleased that you are taking the time to talk with us. Could you introduce us to your field of work and present the goals your department pursues?

Schneider: Of course. Personally, over the past years, I have focused on the set up of SIL test systems and the integration of different algorithms for series products. The key topics here were continuous testing, automation and cloud simulation.

Our department offers outside and intragroup customers solutions for tests that are in part virtual. We build up test benches with different levels of integration, starting from SIL over HIL up to VIL. In many projects, we perform the tests as well – those are mainly projects in the field of advanced driver assistance and brake systems. But we are also responsible for simulation studies and support intragroup projects with our expertise in the area of test strategies. All projects of course have a virtual portion.

The software part in vehicles has rapidly increased over the past years. At Apply & Innovate 2022, you presented a project that Continental Engineering Services implemented with regard to the resulting challenges. What was the aim of this project?

Schneider: Indeed, at Apply & Innovate I presented a CI/CT/CD environment developed by us. That is short for continuous integration, testing and delivery, an automated development method that is well-established in the software industry.

This environment we developed also uses CarMaker as an open integration and test platform and allows to carry out different test steps. These steps include for example statistical code analysis, build and integration tests, unit tests and, last but not least, functional tests. The aim is to achieve a high code quality and to test the series algorithm components systematically.

This subsequently leads to a higher test coverage and prevents bugs. Or, you discover bugs at a much earlier point in time, which leads to a higher maturity in early software versions. The manual quality on code level is therefore significantly increased with a broader and more extensive test strategy – which then automatically leads to faster and better software development.

To achieve a higher quality, there are already many established processes used in software development. Can those be adapted for vehicle development?

Schneider: Definitely. The methods and processes that are used to develop any other software can, to a certain extent, also be applied to the development of automotive software. When developing open-source software, every programming language basically has its own ecosystem with tools accompanying the development, for example for the already mentioned statistical code analysis or functional tests. These tools help to develop powerful open-source libraries that are integrated in billions of devices.

These successful examples can be used as a blue print for company-specific methods and processes. CI/CT/CD is a long-established method in open-source software development processes and particularly supports developers by automating repetitive tasks such as code analysis and software tests.

In your opinion, which criteria are particularly important to successfully implement this kind of project?

Schneider: First of all, it needs to be said that every project or tool has different requirements and specific framework conditions. Hence there is no text-book approach to this. But in my opinion, there are success factors that should be taken into account for design and basic criteria.

The most important thing is that the users accept the toolchain as a means of support. It has to be clear to them that its purpose is not surveillance. Our experiences have shown that only accepted tools perceived as a support are actually actively used. What is also important is the period of time developers have to wait to get the results. They have to be able to know quickly how the analysis of their newly developed code went and if the tests were successful. It is all about keeping the timeframe short so that the focus stays on the developers.

Can you clarify which criteria are decisive for this kind of code analysis?

Schneider: Yes, of course. In this process, transparency as well as reproducibility of results matter most. If it is not known based on which criteria the code is analyzed, this can very quickly result in discouragement. Developers need to be able to reproduce all steps that are performed automatically in the toolchain one by one on their computers.

Personally, I also think it is very important that all proprietary tools in this context are developed with a CI/CT toolchain – not only the final product that is later send out to the customer. The proprietary tools supporting the developers in their work should also be of high quality.

There are other, non-measurable criteria that have an impact on it, too. In our department, introducing such a toolchain has for example somewhat changed our work culture. Due to the implementation of very objective criteria for the evaluation of the developed code, this has opened up the possibility for discussions. We have developed review processes that presume a cooperative exchange. Therefore, multiple employees are always involved in an implementation.

With your experience, would you say that objective and measurable criteria increase the trust in software development processes? Maybe also especially in the automotive and supplier industry, that has by tendency less of a software background?

Schneider: Definitely. Until recently, the automotive industry was just not an industry in which a lot of software was developed. In the past,

Interview partner Axel Schneider speaking with Henning Kemper from IPG Automotive

the business model was strictly limited to producing and selling vehicles – software was always considered like an accessory that had to be delivered with it. But the automotive industry has been changing for years now and software has an ever-growing share in the value creation.

This means that, step by step, OEMs and suppliers are becoming software companies – and in this type of company, there is no way around moving with the times and using modern development processes.

Which role does CarMaker play in the automated CI/CT/CD process chain?

Schneider: Because we are developing software for physical vehicles, we need a virtual counterpart in which we can integrate the driving functions under test – that is the virtual prototype. CarMaker as an open integration and test platform is a key element without which we would not be able to feed our software with valid data. Functional tests of the driving function would hardly be possible in a real-world environment or even not at all for some cases. With CarMaker, the high modularity and the option of extensive parameterizability are particularly helpful.

Could you please specify the measures that you are taking to achieve maximum test coverage?

Schneider: Sure – that is in principle quite simple. In our process, we have linked the requirements that need to be fulfilled to a pre-defined number of test cases. When designing the tests, the vehicle parameterization and test scenario can be varied and adapted at pre-defined positions via a set of parameters. With parameter variations we are then able to use a scaled single scenario with only one vehicle model in an extremely high number of test cases. When all tests are successfully carried out, the requirement is accomplished.

This method has proved to be very successful for us. In one project, we managed to generate 50,000 different tests from 24 maneuvers. Therefore, we only need to actively maintain a few maneuvers and the complexity stays manageable.

What do these parameter variations look like: Do you make small parameter changes on a regular basis, and generate many different tests accordingly on this basis?

Schneider: Correct. We can for example imagine a test case in which we want to test the detection of other road users around the own vehicle. Minimal changes in the movements of those road users alone result in a tremendous number of possible scenarios that all need to be validated. Therefore, we quickly reach the number of 50,000 tests.

Those are impressive numbers. How do you provide the necessary computing power for such a high number of tests, keeping a manageable time effort in mind?

Schneider: As to the technical implementation, we decided to scale the simulation with CarMaker in the cloud. This allows us to add or remove any number of resources to our liking – we can hence cover high load peaks at a low cost.

Can you already predict how this topic is going to evolve in the near future?

Schneider: My opinion aligns with the estimations of experts that I discuss the subject of virtual tests with: Virtual methods are quickly gaining ground. Especially when you look at SAE level 3 driving functions, the equivalent of automated driving functions, they are mainly released virtually and only isolated tests are performed in the physical vehicle. As a matter of fact, the number of the tests to be expected also does not allow any other conclusion either.

Thank you for taking the time and for this insightful interview.