When software writes software

04/08/2021 Software is increasingly becoming the determining factor in automotive development. Marius Mihailovici, Managing Director of Porsche Engineering Romania, looks into the future of software development and explains why the job of a programmer could change completely over the next 20 years.

Even today, cars are very much rolling computers. They contain a network of electronic control units (ECUs), with between 70 and 100 being installed in every modern vehicle. These computing units control fuel injection, regulate braking behaviour and monitor the air conditioning system. The next step will be HCPs (High Performance Computing Platforms), which will enable significantly more computing power to be integrated into an ECU.

The higher computing power and integration are necessary because the number of lines of code and the complexity of the functions in the vehicle are increasing year by year. One number may make this clear: 100 million. This is how many lines of code feature in today's car. By comparison, a Boeing 787 Dreamliner only has 14 million.

There are also many lines of code behind the entertainment system and navigation. Added to this is the possibility of connecting smartphones and other devices to the car, which is also only possible with complex software. And it doesn't stop there: software is taking over more and more important in-car tasks. The most important functions today and in the future include data exchange with other road users and the infrastructure, updating vehicles from the cloud, and eventually even autonomous driving.

The development of automotive software has become a rather tricky balancing act. Safety regulations and customer requirements in the form of voluminous specifications need to be fulfilled. The conventional development processes in the industry are usually time-driven: there is a predefined schedule that defines certain milestones. Our clients expect regular results at predetermined dates.

Added to this are the official approval processes known as homologation. For example, vehicles cannot be brought to market until they have been built and accepted in a certain quantity. In the end, the industry is always about moving from one 'finished' state to the next. All these different challenges mean that fast and flexible work is required. Targets are often set at short notice and change quickly. One might say that software is developed in a results-driven process.

In Cluj, in Romania, our software developers work with tools that have proven themselves in software development for 20 years. We use agile methods wherever possible. They are based on small development steps whose results are checked in daily feedback rounds. The individual teams have a great deal of freedom and work closely, supporting each other with their respective capabilities. They flexibly set their own goals each day. Team leaders often have only a moderating function here and keep an eye on the big picture.

Another current paradigm of software development is 'continuous integration'. This is a highly automated process in which software elements are checked at the end of a working day to see if they are ready to run and be integrated into the overall system. In this way, we can quickly identify and eliminate errors and problems. The use of agile methods and continuous integration ensures greater efficiency and also offers our customers added value: it makes it easier to present interim results and gives them the opportunity for quick feedback.

For the next five years, I expect to see an increasing use of the methods described above. Our work will be even more results-driven than time-driven. Team hierarchies are likely to become less important, and I foresee a higher degree of flat and self-organising teams with clear responsibilities. Moreover, it will become much more prevalent to work flexibly from different locations, based on the idea of putting people to work where the expertise exists. We will also see increasing automation in the coming years. More and more often, software is tested by software and not by people. Manual testing will disappear completely.

The boundaries between car and environment are blurring

Developments outside the automotive industry will also force us to rethink things, as cars are increasingly integrated into the digital lives of their drivers. For example, the smartphone automatically connects to the vehicle when getting in. Media use, navigation and communication merge seamlessly.

Use scenarios like these will change the content of our software developers' work – for the simple reason that the boundaries between pure 'automotive software' and other applications are becoming blurred. Incidentally, this also requires a certain mindset on the part of our developers: we employ people who live the digital lifestyle themselves. They not only know what our clients demand, but also what vehicle users expect from their cars.

ECUs of the future influence software development

New ECU architectures are also changing the way automotive software is developed. I assume that in the car of the future there will be a few central, very powerful computers of the HCP type which, together with subordinate, simpler units, will control the entire vehicle. These central units will also run all applications that go beyond basic functions, such as entertainment, data traffic, or passenger communication applications. These central computers with a real operating system truly turn the car into a PC on four wheels.

For developers, this means that the methods of their work do not change that much. What will change are the systems they are dealing with: they are more hierarchical, have fewer components, and are controlled by one overall software. And this – like any software – will receive regular updates. So the software of a car is not developed and installed once, but is constantly being developed further, even when the car is in the customer's hands.

A future without software developers?

The question is by no means absurd: will there still be software developers in the future? Some experts assume that artificial intelligence (AI) will take over software development completely in the next two decades – even in the automotive industry. Opinions differ on this, however. Personally, I can imagine that we will continue to set the framework and that AI will then implement it. We will still need software architects, requirements engineers, and software engineers to define what AI or neural networks will do: generate software functions based on the defined requirements, test them automatically, and correct them continuously until the software quality is at the expected level.

The developers of the future must therefore be able to think in terms of complete systems. They must know how end customers use their vehicles. Their work is not only determined by the specifications of the OEMs, but increasingly by the consumers. There is one thing, however, that I cannot imagine: a software completely without bugs. There will always be mistakes. We just don't have to find and fix them ourselves; the system of the future will do that, monitored by human software experts.

Marius Mihailovici

Marius Mihailovici has been Managing Director of Porsche Engineering Romania since 2016. He previously headed research and development at Alcatel-Lucent S.A./Nokia Oyj, where he also worked as a software manager for 2G and 3G projects. At World Telecom he worked as a communications engineer.

Info

Text: Marius Mihailovici

Text first published in the Porsche Engineering Magazine, issue 1/2021

MEDIA ENQUIRIES

Frederic Damköhler

Senior Manager Corporate Communications Porsche Engineering
+49 (0) 711 / 911 16361
frederic.damkoehler@porsche.de

Consumption data

Taycan Turbo S (2023)
Fuel consumption / Emissions

WLTP*
Electric power consumption* combined (WLTP) 23.4 – 22.0 kWh/100 km
CO emissions* combined (WLTP) 0 g/km
CO2 class A Class

*Further information on the official fuel consumption and the official specific CO emissions of new passenger cars can be found in the "Leitfaden über den Kraftstoffverbrauch, die CO-Emissionen und den Stromverbrauch neuer Personenkraftwagen" (Fuel Consumption, COEmissions and Electricity Consumption Guide for New Passenger Cars), which is available free of charge at all sales outlets and from DAT (Deutsche Automobil Treuhand GmbH, Helmuth-Hirth-Str. 1, 73760 Ostfildern-Scharnhausen, www.dat.de).

Link Collection

Link to this article
https://newsroom.porsche.com/en_US/2021/technology/porsche-engineering-when-software-writes-software-25367.html

External Links
https://www.porscheengineering.com/peg/en/

Dr. Ing. h.c. F. Porsche Aktiengesellschaft
Porscheplatz 1
70435 Stuttgart