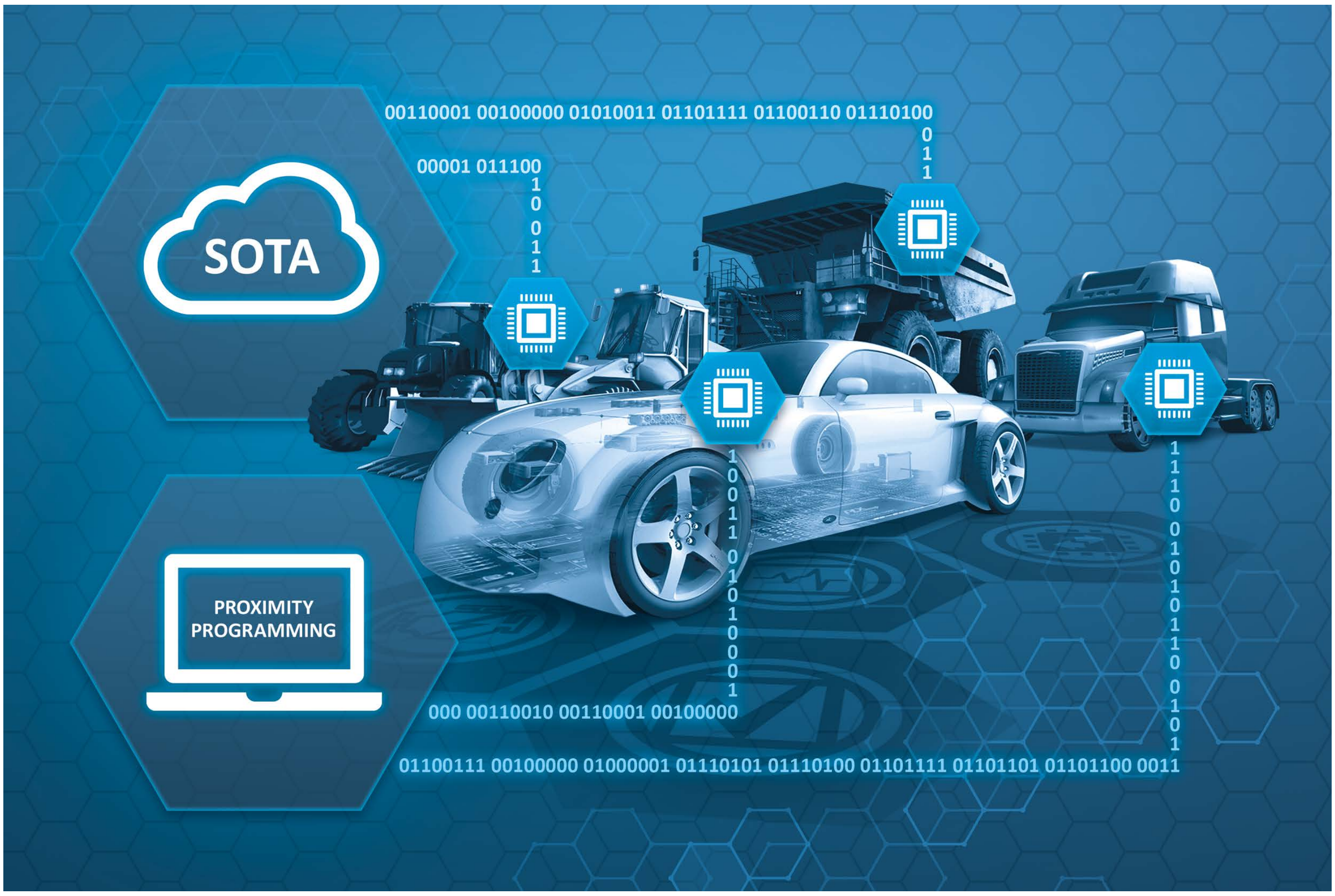
Originally published in German in the section "Engineering" of the technical magazine HANSER automotive 3/2021, published by Carl Hanser, Munich, Germany.



Mastering Flexibility in Flash Programming

Today, flash programming ensures tremendous flexibility – across all ECUs and throughout the entire vehicle life cycle. The challenges that arise in terms of performance, processes and the volume of data can be managed easily with the right system architectures.

The flash programming of electronic control units (ECUs) in vehicles has now become widely accepted: There are anywhere between 3 and 120 ECUs depending on the market segment. These can usually be supplied with new software directly or via a gateway using a tester. This capability is used in a number of ways, although a vehicle interface currently has to be plugged in to mediate between the programming application and the ECUs. In the future, over-the-air updates will open up further possibilities.

Freedom in the Life Cycle

An increasing proportion of the value added in today's vehicles is generated by software, and this will gain momentum as driver assistance functions become more widespread. This also has process advantages as software is easy to parameterize and can thus be easily adapted to different basic conditions. One example is variant coding, where hardware and software is parameterized to represent 150 hp and 190 hp as power output. In the same way, countless vehicle functions are parameterized via characteristic maps and individual adjustment values – and changed if necessary. This is used both at the beginning of the life cycle, when behavior is being coordinated, as well as in later stages. This allows changes in customer taste to be taken into account, for example by adapting the sound system to a driver's current listening habits by turning up the bass. However, unexpected aging behavior of sensors can also be counteracted and the corresponding characteristic curve can be adjusted. In the same way, entire control unit functions can be retrofitted. The prerequisite is, of course, that it is either a pure software function or that the required hardware is already installed. If this is the case, only the new software has to be programmed. Errors in the software can be dealt with in the same way. The error is put right, a new software release is made available and installed in good time.

This is used over the entire life cycle. In engineering, ECU functions are coded, variants created, functions parameterized, and errors corrected. In manufacturing, the latest software version must be programmed into the ECUs, and, after delivery, bug fixes, adjustments and new functions are integrated into customer vehicles in the repair shop.

The Challenge of Programming

The programming itself is by no means a trivial process. The vehicle is usually accessed via the OBD jack and a central gateway. Today, however, some ECUs can also be reached directly; others only via several gateways. The buses via which the ECUs are integrated in the vehicle network are also extremely heterogeneous in terms of their data rates and message lengths. Today there are already functions which are not limited to one ECU – a fact that must not be forgotten.

In this case, an update involves bringing several ECUs up to date at the same time. The data volumes also differ, sometimes considerably. Simple ECUs transfer kilobytes, others require several GBytes, e.g. navigation data. Not only the transmission to the ECU is a challenge, but also the distribution for worldwide use has to be mastered.

The process itself usually takes place in three stages.

- In the preparation phase, the ECUs to be programmed are identified and the available flash data is determined. Security algorithms are then executed – after all, not everyone should be allowed access to ECUs – and the environment is prepared. This includes establishing bus silence on the CAN bus in order to have the full bandwidth available and to inform the other ECUs about the programming process.

- The next step is the actual programming of the ECUs.

- Finally, successful transfer is validated and the environment is returned to its original state.

There are also many challenges from the process side. This starts with selecting the right flash data – after all, vehicles are usually only in a known state for a short time after manufacture because then some updates are applied and others are not. So first the current state must be identified. On this basis, the scope to be programmed must be determined – taking release states across multiple ECUs into account.

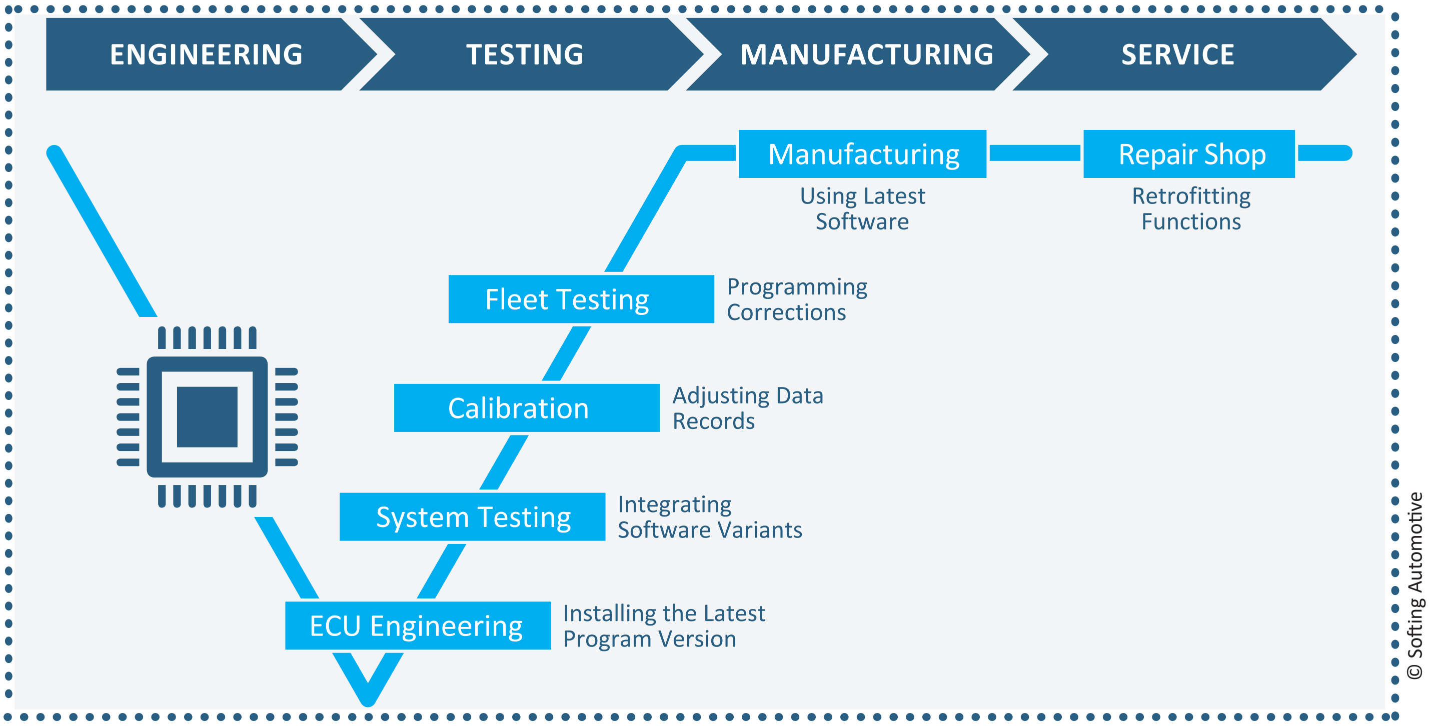


Figure 1: Use Cases in the Entire Process

The way back must also already be clear here (rollback), since in some cases the programming of a single ECU does not work, but in the end all ECUs must be in a permissible state – both in terms of function and with regard to the release. Robustness also includes an infrastructure that reduces such problems. This can include a stable Vehicle Communication Interface (VCI), the use of break-proof cable connections (e.g. Mag-Code) or the avoidance of wired solutions with a simultaneously stable WLAN coverage.

With regard to performance, which is crucial particularly with the high data volumes mentioned, it is necessary to take a holistic view of the process and architecture: Which data really needs to be changed, which ECUs can I program in parallel in a meaningful way, can I separate the data download to the vehicle from the actual programming. A combination of measures leads to the best results here.

Practical Solutions

To ensure that data can be used consistently throughout the life cycle, a three-part software architecture has become established for test systems. This consists of a uniform protocol implementation, a likewise uniform runtime environment for diagnostics and the applications. The latter are implemented specifically for the use case. In solutions, the software layers can basically be distributed variably on the hardware components used, i.e. the PCs and VCIs.

In ECU engineering, new code and new parameterizations are initially loaded into the ECU quickly and without major data logistics. This takes place either directly via the diagnostic engineering tester or via other engineering tools that have an integrated diagnostic runtime system. The latter is usually installed together with the application on the engineering computer. The connection to the ECU is made via a local VCI, which is connected to the PC via USB or WLAN.

In production lines, the focus is on performance and secure data transmission to meet cycle times and avoid rework. Since WLAN coverage in particular is a challenge here, splitting up the software structure has proven its worth. Data and the diagnostic runtime system are located directly on the VCI and are (almost) independent of the connection to the host computer; data is supplied at dedicated points, in some cases even via LAN cables.

In principle, this architecture can be adopted in repair shops. The latest vehicles go even one step further. The VCI is eliminated here and the diagnostic runtime system is integrated directly into the vehicle, for example into a TCU (Telematic Control Unit). The data is also stored on the vehicle and updated in parallel with the ECUs. The repair algorithms in the repair shop tester can thus be based on existing diagnostic sequences that match the vehicle.

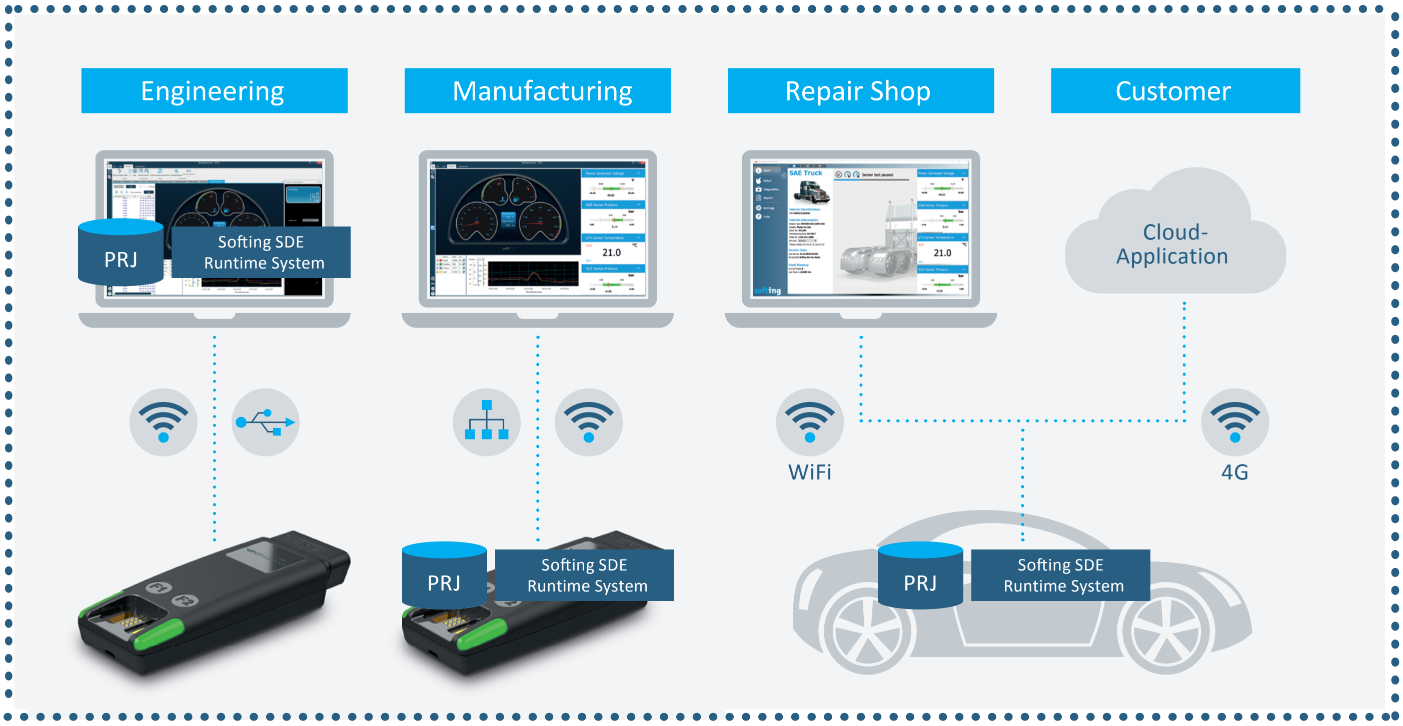

Engineering
Manufacturing
Repair Shop
Customer
PRJ
Softing SDE Runtime System
Cloud-Application
WiFi
4G
PRJ
Softing SDE Runtime System
PRJ
Softing SDE Runtime System

Figure 2: Smart Architecture Enables Scaling

Originally published in German in the section "Engineering" of the technical magazine HANSER automotive 3/2021, published by Carl Hanser, Munich, Germany.

Remote Increases Efficiency

Integrating the diagnostic runtime system in the vehicle also creates even more degrees of freedom: In combination with the TCU, simple remote diagnostics is possible. This opens up new possibilities as early as the engineering stage. An FMU (Functional Mock Up) can be supplied with new software directly from the office, and the same applies to vehicles in the prototype repair shop. And test benches, many of which are now operated in other regions, can also be safely updated.

After the vehicle is delivered to the customer, SOTA (software over the air) is the magic formula today. Critical errors in particular are easy to rectify as soon as the vehicle is in a safe condition and the driver agrees. This is beneficial for all parties if it prevents recalls.

We Can Do It!

With the increasing proportion of software in the vehicle and the growing criticality of software functions, the need to be able to adapt ECUs is growing. Flash programming offers many degrees of freedom while posing some challenges in terms of performance, security, and process definition. Suitable software architecture for the programming solution significantly supports efficient use in the field.

> > www. automotive.softing.com

Markus Steffelbauer heads up Product Management and Marketing at Softing Automotive and is a committed member of standardization bodies.