COMPUTER ON WHEELS PART 4

JULY 2022

AUTHORS

WOLFGANG BERNHART
Senior Partner

MARKUS BAUM
Partner

FALK MEISSNER
Partner

KONSTANTIN SHIROKINSKIY
Partner

The future of the automotive software industry: Spend, trends and how to transform

The automotive software market is changing with the rapid growth of software-enabled features in cars. The software spend of OEMs and suppliers is set to skyrocket by 2030 with shifting proportions of different software cost categories. A new, software-defined approach to car design has emerged: cars are built around software platforms rather than the incumbent approach of integrating software into cars. The shift in design philosophy will enable the auto industry to become more innovative and avoid almost USD 16 billion in costs annually. Transforming to the new approach isn't straightforward. It requires new thinking, new technology and new business models.

Introduction

Today's automobile has become more than simply a way of getting from A to B. It has become a software-enabled feature-packed device – a computer on wheels. In Parts 1 and 2 of our Computer on Wheels series, we looked at how electronics and software are changing the automobile industry, turning OEMs into software players. In Part 3, we assessed the role of Tier 1 suppliers in the new landscape and how they need to adjust.

In this Part 4 we focus on the future software spend of automotive OEMs and their incumbent and new suppliers, and how they need to adapt. In particular, we look at how the industry's transition from incumbent thinking (integrating software into cars) to a software-defined approach (building cars around software) will develop, and what opportunities it will create.

Software first – The path to successful future car building

The value of cars and related services is increasingly dependent on software content, with the rollout of electric and ever-more autonomous vehicles driving this growth. More software means more complexity and challenges: new products, services and upgrades take a long time to develop and bring to market; they are hard to monetize after the initial sale; agile software development methods contradict traditional development cycles; testing, integration and maintenance costs are high; and it's difficult to define future requirements.

These problems are largely rooted in the auto industry's "Incumbent Approach" to building cars: fitting software into existing or new automotive models. This is increasingly complex, cumbersome, and therefore expensive, and hampers innovation. Most OEMs are now transitioning towards a "Software-Defined Vehicle (SDV) Approach", where the software characterizes the hardware and the car is built around software platforms. In the following chapters, we look at the three key aspects of this approach: a scenario showing the future software spend and potential cost avoidance of shifting to a purely SDV approach; the required technologies, such as microservices; and potential future business models.

FALK MEISSNER
Partner

"The value of cars and related services is increasingly dependent on software content. Automotive OEMs need to transition towards a Software-Defined Vehicle Approach, where the software shapes the hardware around it."

Spend and save: Software costs will rise, but the new approach offers huge efficiencies

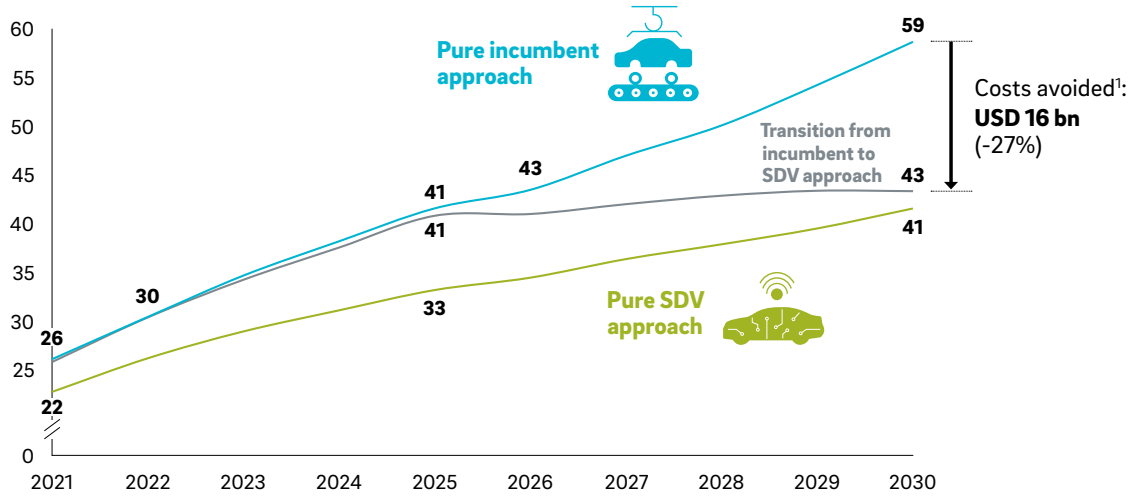
The transition to the SDV approach is a fundamental requirement for the automotive industry to meet complexity challenges and increase functional demand. It also represents a massive opportunity to boost value by generating efficiencies. Under the SDV approach, we expect the necessary software spend in the automotive industry[1] ' to rise by 70% – from USD 26 billion in 2021 to USD 43 billion in 2030[2] (USD 327 to USD 575 per vehicle). Under the incumbent approach, the software spend is expected to more than double – from USD 26 billion to USD 59 billion in 2030 (USD 419 per vehicle). This means the SDV approach avoids costs of almost USD 16 billion per year and enables the industry to meet the coming challenges.

1 Excludes autonomous driving pure plays and electronic manufacturing services.

2 Assumes that the features currently expected will be delivered during this timeframe.

Our scenario

OEM automotive software spending forecast, 2021-2030 [USD bn]



[1] Assuming fixed software feature level in all scenarios

Source Expert interviews

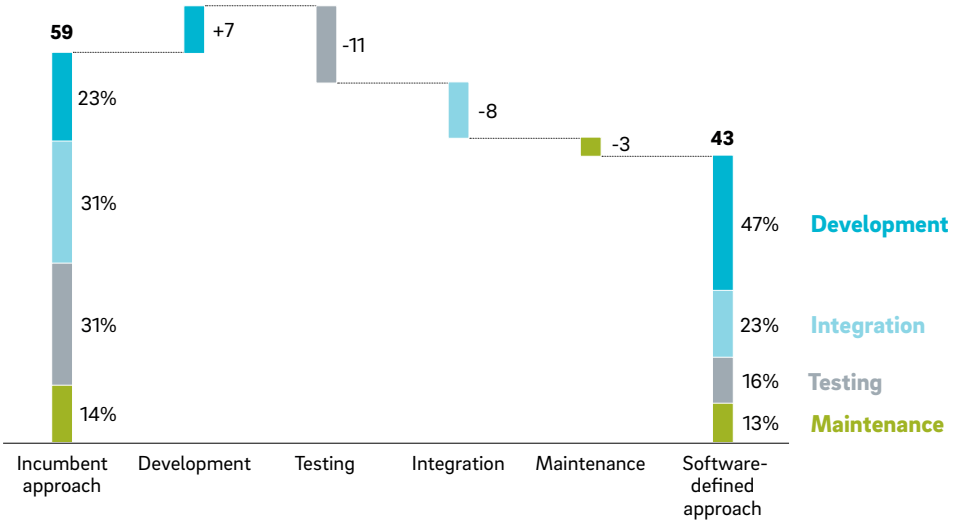
Roland Berger

In both cases, the increase in software spend is driven by the increasing sophistication of infotainment (IVI) and advanced driver-assistance systems/autonomous driving (ADAS/AD), increased maintenance costs, and a rise in application development. This results in an increase of automotive software spend of 6% CAGR until 2030. Put bluntly, only by avoiding costs can the industry afford the software required in the future.

Below we assess the USD 16 billion cost avoidance achieved by the SDV approach versus the incumbent approach. For a 360-degree view, we approached this from three different angles – the software stack level, the domain level and in terms of development steps (development, integration, maintenance and testing). For each, we discuss how the projected costs alter the make-up of the total automobile software spend in 2030 compared to 2021.

Costs by development step

Incumbent versus SDV approach, 2030 [USD bn]

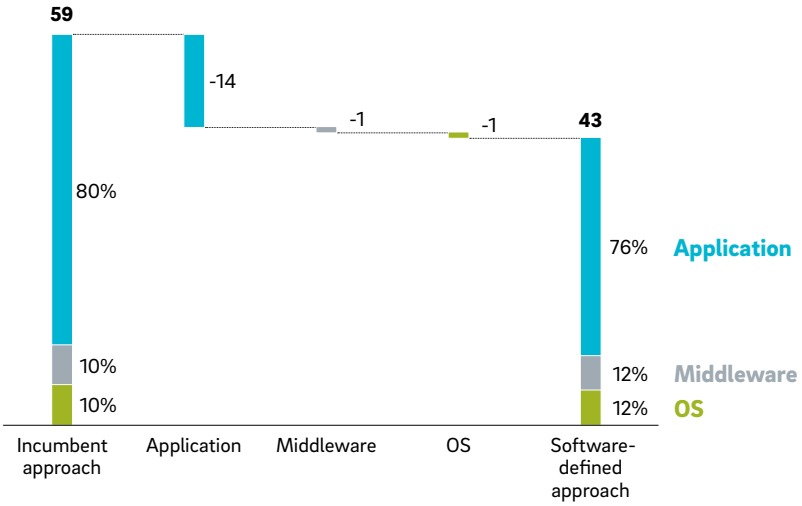


Source Expert interviews

Roland Berger

Costs by stack layer

Incumbent versus software-defined approach, 2030 [USD bn]

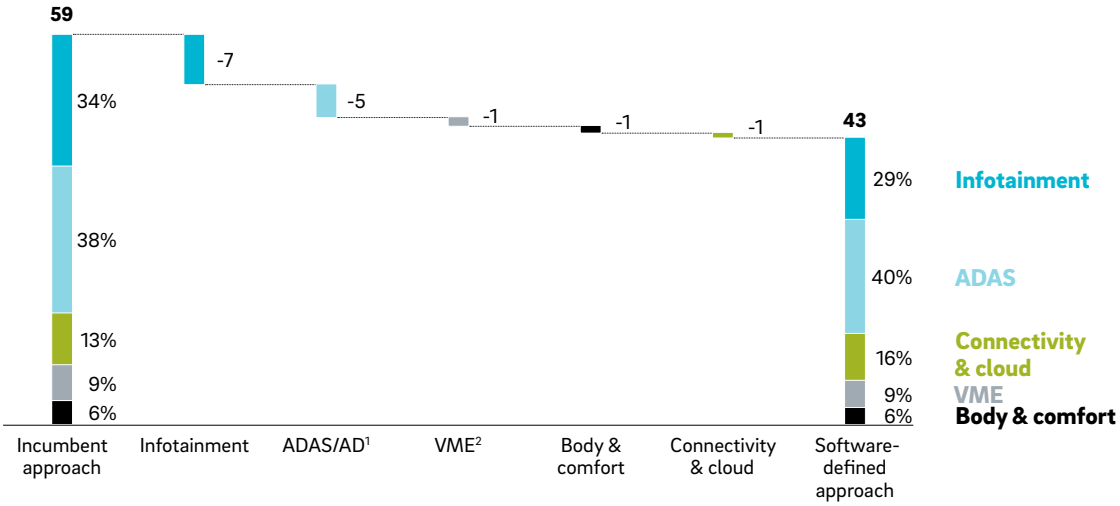


Source Expert interviews

Roland Berger

Costs by vehicle domain

Incumbent vs. software-defined approach, 2030 [USD bn]



1 Including autonomous driving (AD) systems up to SEA level 4 2 Vehicle motion and energy

Source Expert interviews

Roland Berger

Stack level

The classic software stack is applicable to cars – hardware layer at the bottom, followed by operating system, middleware and services, topped by the application layer. In the SDV approach, the application layer is the source of almost all the avoided costs. The SDV approach lowers the high application development costs of the incumbent approach as integration and testing efforts are greatly reduced. The application layer contributes USD 14 billion of the almost USD 16 billion total cost avoidance offered by SDV approach. Reinvestment in application development will result in a CAGR of 7% in average for the overall application stack across all domains.

Domain level

Infotainment and ADAS/AD are the two most complex vehicle domains, due to the complexity of the required signal processing, object detection and safety systems. They therefore benefit most from the SDV approach. Infotainment accounts for USD 7 billion in avoided costs versus the incumbent approach, and ADAS/AD slightly less at USD 5 billion, driven by the high costs of AI. This will drive the automotive software spend at a CAGR of about 8% for the AD/ADAS and IVI domains in particular.

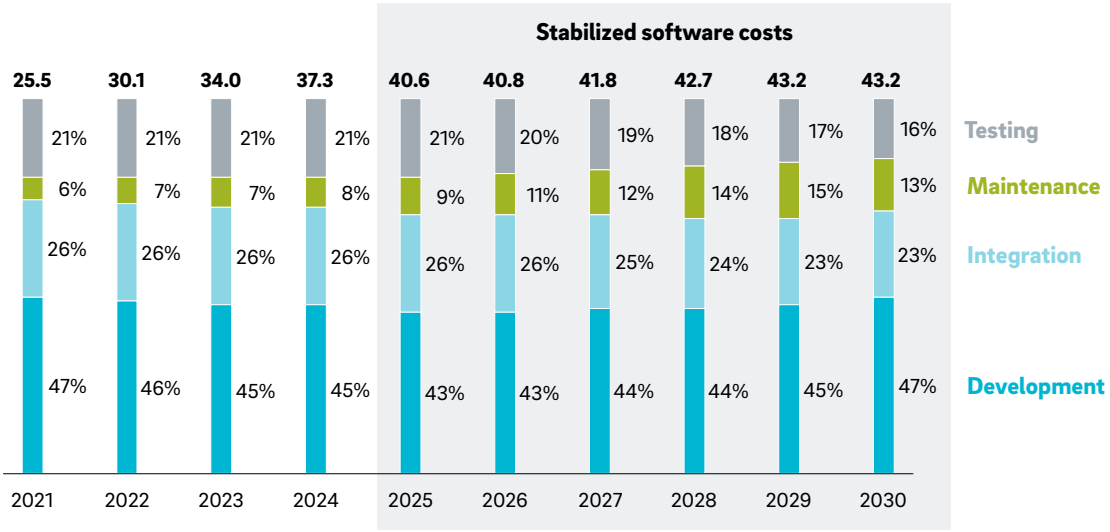
Development steps

Gauging costs by each step in the development cycle incorporates aspects of both the stack and domain levels. Initially, the SDV approach requires the setting up of more complex architectures (for example, microservice architectures – see box), resulting in increased development spending. This rise of around USD 7 billion is more than offset by big cost reductions in more agile software production: testing (USD 11 billion), integration (USD 8 billion) and maintenance costs (USD 3 billion). In addition, this frees up resources for content development.

In the SDV approach, the cost for maintaining and updating an increasing set of interdependent software over long periods after SOP, although lower than in the incumbent approach, remains a driver for growth, with a CAGR of 16%.

Development costs

OEM automotive software spending forecast by development step, 2021-2030 [USD bn]



Source Expert interviews

Roland Berger

What are microservice architectures?

Microservice architectures (MSAs) enable the rapid, frequent and reliable delivery of large, complex applications. Agile teams of developers independently develop, maintain and improve new services (microservices), each linking into the whole through a pre-defined API (application programming interface) gateway. The benefits, compared to conventional monolithic architectures, include reduced interdependencies, increased system stability, faster times to market and a more organized development process.

Key assumptions

The model outlined above is based on two core assumptions:

1. **Functional content will develop as expected (for example, in terms of autonomous driving):** Consumers will increasingly demand a digitally seamless experience, and OEMs will strive to generate additional revenue from software-enabled features.

2. **The shift from an incumbent to an SDV approach will be largely completed by 2030:** OEMs will increasingly roll out new software and electrical/electronic (E/E) architectures.

Ultimately, OEMs that fail to change their approach will have a substantial competitive disadvantage: higher costs, lower customer value and most importantly, outdated products. In the next chapter we look at the technology required to successfully achieve the transition.

The right tech: Only an SDV approach using new architectures can succeed

The incumbent approach aims to integrate software into a car, typically by adapting it to different models and configurations in a time-consuming process. This contradicts the lower effort, shorter time to market, efficient maintenance and reduced complexity requirements of software development. Therefore, the automotive industry needs to move away from today's technology to a software-centric approach.

Benefits of the new approach

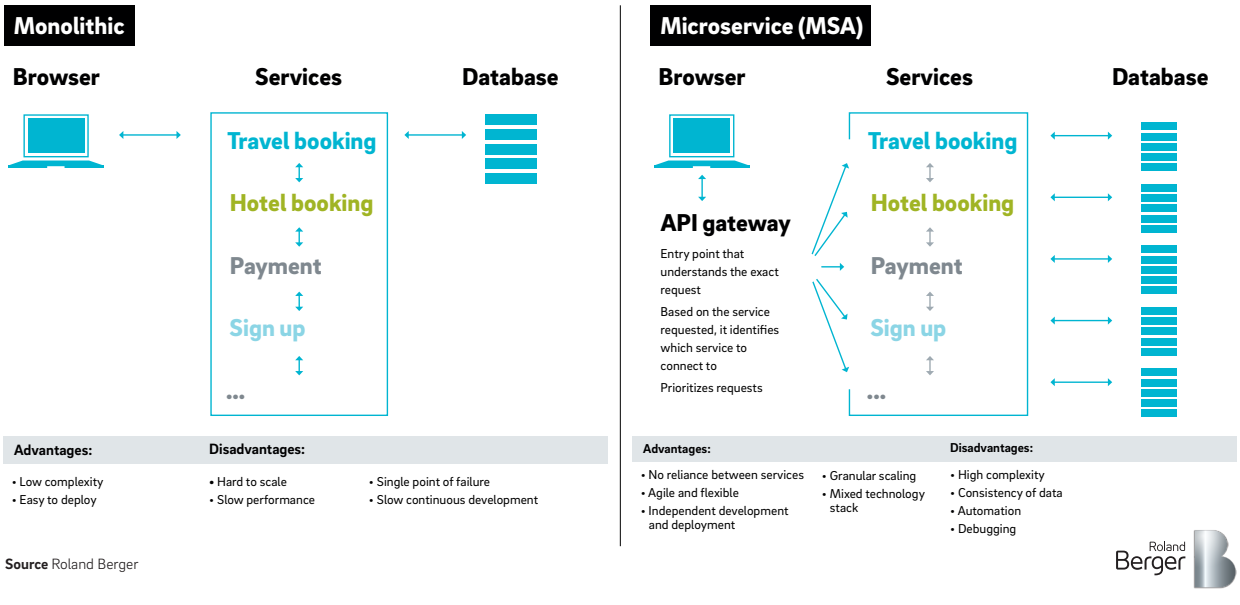
Today's technology is characterized by a monolithic software architecture and a distributed E/E architecture, resulting in the following:

- High testing, integration and maintenance costs, driven by complexity and the effort of providing new functions, upgrades and updates
- Long development cycles and no continuous deployment, driven by complexity of making changes to monolithic code.

In contrast, the SDV approach is characterized by new service-oriented or microservice-based architectures (MSAs) and a central, zonal E/E architecture. Its key advantages stem from breaking down functions into independent, more manageable parts, enabling faster testing, validation and releases.

Mono versus micro

Comparison of monolithic and microservice architectures



Source Roland Berger

"A Software-Defined Vehicle approach offers multiple advantages across all development steps and thus becomes a key success factor for automotive OEMs."

WOLFGANG BERNHART
Senior Partner

Below we outline these advantages by development step, highlighting the drivers of avoided costs.

Development
Incumbent monolithic architectures rely on a single continuous string of code, which must be updated in its entirety even if only a small change needs to be made to a service. In new architectures, each service is independently developed and deployed, meaning changes only require updates to individual service-specific code within the parameters of the defined interface. In microservices, the use of individual microservice APIs and an API gateway allows services to communicate directly, independent of their chosen implementation. The definition of the APIs and designing services in such a highly independent structure requires high levels of effort, and therefore drives initial development costs.

Testing
The single binary file of monolithic architectures hampers testing, too. Whenever an upgrade or update is introduced, the entire application needs to be re-built and validated. This is not the case in MSAs: upgrades affect only individual microservices, reducing the need to test the overall application. The reduction in testing is the largest avoided cost of the SDV approach.

Integration
A key benefit of the SDV approach is the drastically reduced system integration efforts enabled by hardware abstraction and MSA. This is in contrast to monolithic applications, which must be completely re-built and taken offline to implement code changes.

Maintenance
The main advantage here is the ability to reuse code and incrementally adopt it (for example, in agile development and code maintenance) without having to integrate and test a whole new function or system. In addition, single points of failure have a far smaller impact in an MSA as they do not bring down the entire system right away.

So, in contrast to monolithic architectures, the overall effects of new architectures on software development are:

- Lower testing, integration and maintenance costs, driven by the impact of microservices
- Short and continuous development cycles, enabled by, for example, avoidance of downtime and agile development.

It is important to note that microservices are currently used only for non-safety critical functionality. However, we expect this to change in the medium term. As the ability to scale in-vehicle computing power increases and the availability of safety-certified container technologies widens, microservices are expected to be applied throughout all vehicle domains in the future.

System upgrade: Successful transformation requires innovative new business models

The shift towards an SDV approach will have significant implications for the automobile industry. We find that the major automotive players already try to move away from monolithic architectures to multi-component or microservice architectures with a high degree of reuse. The major obstacle on their journey is, however, the management challenge to oversee and steer the vast amount of software developers working across millions of lines of code, says software development process expert **Johannes Bohnet, CEO of Seerene**. OEMs and suppliers need to transform to compete in the new market environment, meaning they must rethink their business models and role in the value chain – and they must be able to systematically drive the necessary transformation processes in their software development units.

How to remodel

Today, OEM software requirements are highly specific to their particular vehicle platform and architecture. Tier-1s and software suppliers provide customized solutions to individual OEMs, who usually own the IP. This creates inefficiencies as it drives-up integration and testing costs, prevents reselling of software, and therefore limits economies of scale.

The automotive industry needs to establish four key tenets of software-driven business model innovation: common standards, an open-source approach, IP trading and efficient IP management. Below we look at each.

1. **Common standards**
   One result of the industry's transition to an SDV approach will be the emergence of industry standards based around the new architectures, such as MSAs. We can expect different standards for safety critical and non-safety critical functions, helping to harmonize all OEM software requirements. We often experience a different understanding of software criteria between suppliers and self-developed components at our customers. This leads to poor controllability and manageability. Independent, uniform benchmarks must become standard to create true reusability – says software process mining expert Dr. Johannes Bohnet, CEO of Seerene.

2. **Open-source approach**
   Common standards enable the development of open-source software. It maximizes the re-use and/or even trading of software IP among OEMs and Tier-1s and the safety of software content, creating economies of scale and driving down costs.

3. **IP trading**
   OEMs, suppliers and tech firms are expected to increasingly engage in the buying and selling of code, enabled by the low testing/integration costs of already-proven software based on common requirements. By offering software as a product, firms leverage their IP assets, create economies of scale and monetize their investments.

While remarketing IP is already a core business for suppliers, it will be new ground for OEMs. It will require additional resources, new processes, boosting partnerships with suppliers and the emergence of software marketplaces.

4 IP management

Software IP represents a growing share of company value and is increasingly a competitive factor. As discussed, automotive players currently miss out on its monetization potential due to heavy customization and the lack of industry standards. To realize the additional commercial potential of IP trading, automotive players need to establish a proactive and holistic IP management covering four core areas: a comprehensive IP strategy (closely linked to the corporate strategy); a suitable organizational form; IP processes (purchasing, monetization, defend and protect); and IP tools and systems (for example, IP management systems).

Key recommendations

The SDV approach enables OEMs and suppliers to update and extend their business models, as well as focus on economies of scale and state-of-the-art software. Our key recommendations are:

"To enable an SDV approach, the automotive industry needs to establish common standards, an open-source approach, IP trading and efficient IP management."

KONSTANTIN SHIROKINSKIY
Partner

Microservice software architectures

Deploy new software architectures, such as microservice architectures, to reduce development complexity, integration, testing and maintenance efforts, as well as make use of continuous deployment.

Containerized middleware and applications

Implement containerized middleware to allow for continuous deployment and cloud-native design principles to run applications on the car and in the cloud.

Central + zonal E/E architecture

Develop central + zonal E/E architecture to provide a suitable hardware environment for the software-defined approach.

IP management & economies of scale

Set up a dedicated IP management unit incl. sufficient tools and processes to make use of economies of scale from trading IP/software with other players to maximize value generated by software functions.

Software-as-a-product

Suppliers should invest in in-house (platform) development, moving towards SaaP, and therefore bringing to market their own solutions while owning the software IP.

Software licensing

OEMs should monetize software architectures by selling/licensing them to other manufacturers, eventually partnering with OESs to leverage their distribution channels.

Further reading

ADVANCED TECHNOLOGY
➔ rb.digital/advanced_technology

COMPUTER ON WHEELS: A NEW ROLE FOR TIER 1 SUPPLIERS
➔ https://rb.digital/Computer-on-wheels-3

COMPUTERS ON WHEELS: TURNING OEMS INTO SOFTWARE-ENABLED COMPANIES
➔ https://rb.digital/Computer-on-wheels-2

THE CAR WILL BECOME A COMPUTER ON WHEELS
➔ rb.digital/Computer-on-wheels-1

WOLFGANG BERNHART
Senior Partner
Stuttgart Office +49 711 3275 7421
wolfgang.bernhart@rolandberger.com

MARKUS BAUM
Partner
Stuttgart Office +49 711 3275 7121
markus.baum@rolandberger.com

FALK MEISSNER
Partner
Chicago Office +1 510798 7550
falk.meissner@rolandberger.com

KONSTANTIN SHIROKINSKIY
Partner
Detroit Office +1 248 729 5132
konstantin.shirokinskiy@rolandberger.com

We thank Patrick Kazmaier and Constantin Lang, consultants in Roland Berger's Automotive Practice, for their contribution.