McKinsey & Company

Advanced Electronics Practice and Automotive and Assembly Practice

# The case for an end-to-end automotive-software platform

Software is transforming car capabilities but also creating development challenges for automotive players. An end-to-end approach that integrates independent software elements into a comprehensive platform can improve functionality and decrease complexity.

*by Ryan Fletcher, Abhijit Mahindroo, Nick Santhanam, and Andreas Tschiesner*



January 2020

**Today's hardware-defined cars** are rapidly transforming into software-defined transportation platforms. The latest automotive innovations, including intuitive infotainment, self-driving abilities, and electrification, depend less on mechanical ingenuity than on software quality, execution, and integration. This change is happening so rapidly that automotive OEMs and other industry stakeholders are now struggling to keep pace. The enormous cost of integrating and upgrading the features that consumers increasingly expect, including high-end onboard assistants and advanced driver-assistance systems (ADAS), is also daunting.

## Automotive-software development: Trapped in a maze of complexity

Despite the clear importance of software to vehicle performance, the development of automotive-software modules frequently occurs in isolation. An OEM's in-house team may build some; others are purchased from suppliers or come out of strategic partnerships or joint ventures. Once the full set is available, OEMs or their tier-one suppliers try to stitch the modules together into a proprietary platform.

A typical new-generation vehicle likely has a software architecture composed of five or more domains, together comprising hundreds of functional components in the car and in the cloud. These cover everything from infotainment and ADAS to mapping, telematics, and third-party applications (Exhibit 1). Typical OEMs constructing this architecture interact with a multitude of software providers to build various capabilities; in the process, they fill their vehicles with a broad set of development languages, operating systems, and software structures. This piecemeal approach is common among industry leaders because no single software platform on the market can meet all cross-system needs.

Many automotive companies leverage the basic code for their software stacks, including those for operating systems and key middleware, from other industries. In doing so, they significantly reduce their development timelines and costs, since creating original code is much more difficult. For infotainment, the leading options for upcoming vehicles stem from the smartphone industry, with automotive variants of mobile operating systems becoming common. For ADAS, companies originally borrowed early software from aerospace applications and manufacturing automation; today, real-time operating systems from semiconductor players and embedded-software companies have become popular.

Software modules such as these are impressive in their own right and have enabled some of the most important automotive advances over the past ten years. But we are entering a new age in which automotive features increasingly rely on seamless integration among multiple vehicle subsystems. For instance, the active suspension on several new luxury vehicles requires real-time interaction among ADAS cameras, powertrain sensors, and chassis actuators—three discrete domains with separate software architectures, operating systems, and middleware.

Although integration among these vehicle systems is essential to unlock new use cases, companies are missing an end-to-end platform to connect everything together. OEMs and tier-one suppliers therefore face a daunting task of interface control and integration, creating major challenges in development, security, and performance.

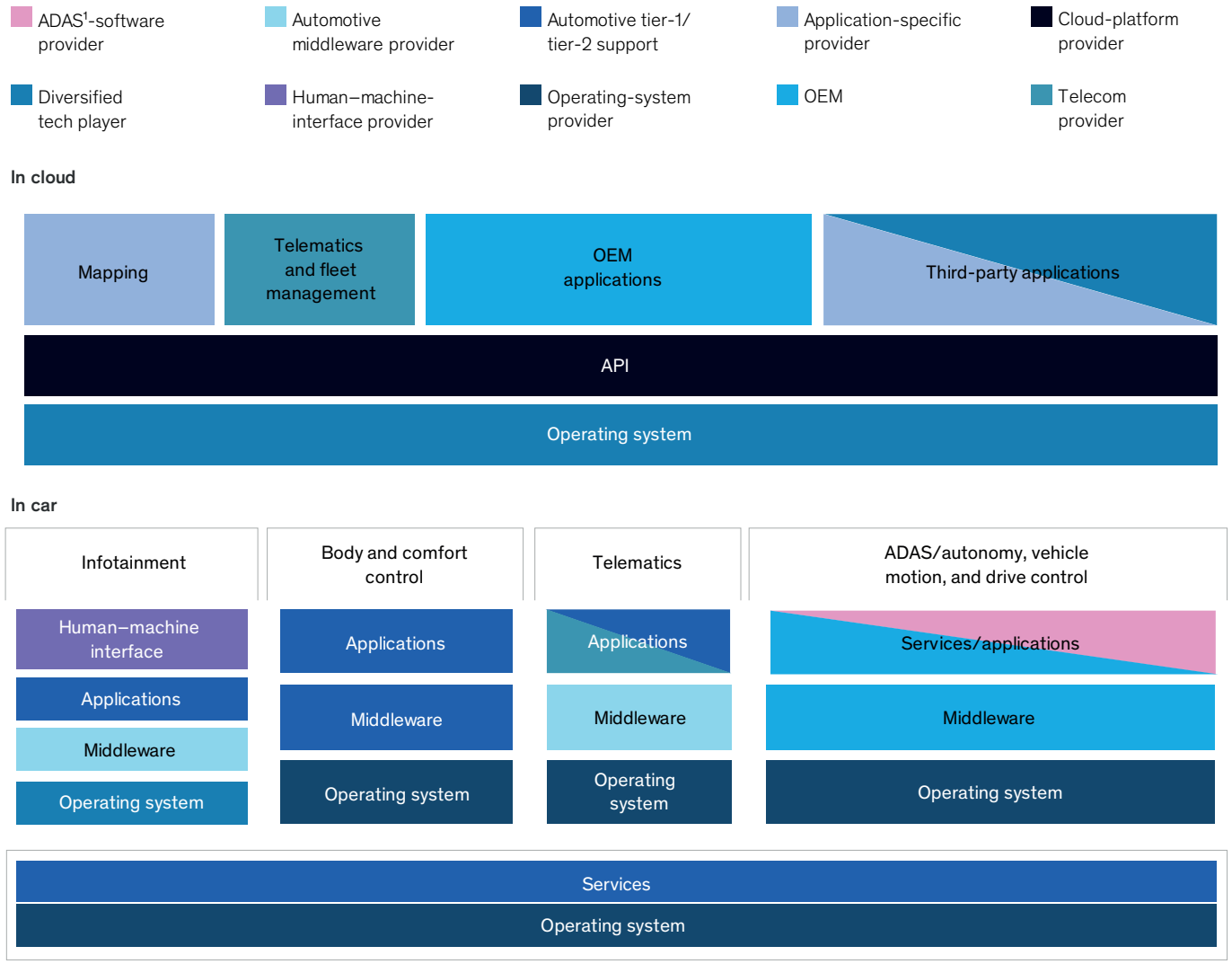
### Complexity is ballooning, while development productivity is stagnating

The overall development challenge is becoming very real. As software content rises in nearly every part of the vehicle, so does the effort required to make different systems work as a cohesive whole. Automotive players have an extremely tough time keeping up. For instance, modern infotainment systems now take upward of three years to develop, with several hundred software engineers contributing to each iteration. Of this effort, 30 to 50 percent is commonly dedicated to integration,

Exhibit 1

## Today, OEMs are stitching disparate software components together to build proprietary platforms.

**Vehicle-software components** (illustrative example, simplified)




[1] Advanced driver-assistance system.

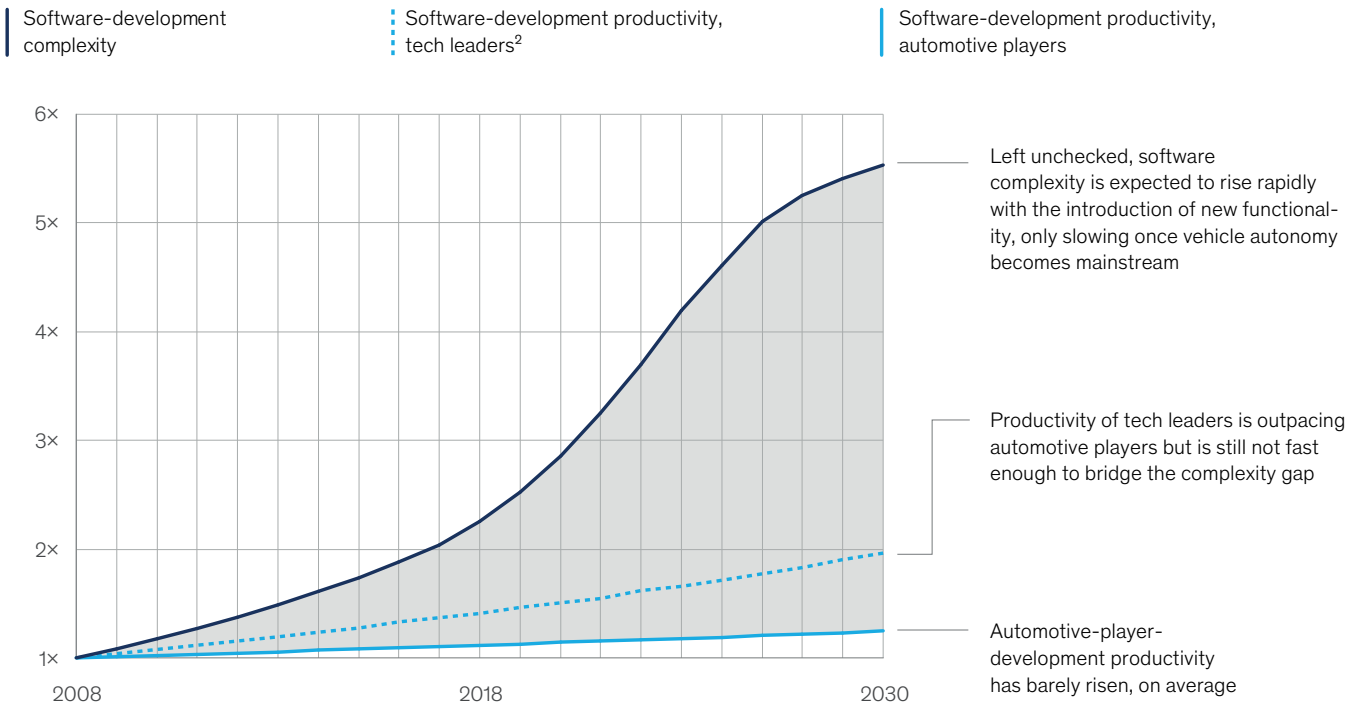
given the broad network of suppliers involved in development. Changes to any one software module often require extensive rework. These systems are not always backward compatible and thus require extensive redevelopment every few years to stay up to date with new features and performance.

Compounding the problem, software complexity is on track to nearly triple over the next ten years. OEMs and suppliers will have difficulty dealing with this complexity because their productivity is not increasing at the rate needed to sustain innovation (Exhibit 2). With the gap between complexity and

Exhibit 2

## The automotive industry is confronting a widening and unsustainable gap between software complexity and productivity levels.

**Relative growth over time, for automotive features,**[1] indexed, 1 = 2008




[1] Analysis of >200 software-development projects from OEMs and from tier-1 and tier-2 suppliers.
[2] Top-performing quartile of technology companies.

Source: Numetrics by McKinsey

productivity widening, OEMs and tier-one suppliers will soon face a massive talent shortage and a huge increase in development costs.

OEMs are awakening to the growing productivity problem and do not anticipate a quick fix. With talent in short supply, they cannot merely toss more resources at the problem. Instead, they will have to streamline software development by reducing complexity.

### Security concerns are becoming real

The complexity jump is also creating a host of new, difficult-to-trace security risks for connected vehicles. Much of the problem relates to advances to in-vehicle networking that create links among formerly independent electronic domains, such as infotainment, ADAS, and powertrain. These connections provide a conduit for attacks to spread through a car, since software vulnerabilities in one system can be exploited to provide access to other systems. Developers working on different software stacks across a vehicle seldom coordinate their activities, however. It is also difficult to align updates and patches across modules, particularly since the number of potential "attack surfaces" rises as the number of connected and autonomous driving systems increases.

Some top automakers have already experienced well-publicized security breaches. Hackers have proved their ability to access locks, brake systems,

and dashboard displays remotely via the cloud connections in several models. Security researchers demonstrated that they could connect to a vehicle's drive-control systems via the infotainment interface, gaining access to powertrain, infotainment, and climate functions. Most recently, other researchers demonstrated the ability to disable antitheft systems, doors, lights, and brakes through a Wi-Fi connection. OEMs rapidly addressed these vulnerabilities, but such incidents exposed the risks inherent in connecting software modules developed for formerly independent domains.

**Connectivity is driving performance**

Software has eclipsed hardware as the main performance driver in today's cars, with connectivity across domains a critical enabler for new features. For internal-combustion engines, software has enabled recent innovations, such as rapid stop–start technology to minimize idling and variable valve timing to improve efficiency. For electric vehicles (EVs), software is even more critical, particularly for trade-offs between performance and range. Take thermodynamics, for example. In gas-powered cars, engines generate excess heat and power that climate-control systems can use, but EVs do not have this capability. In fact, their range might fall by half or more in very cold or warm environments, when drivers are more likely to set air conditioning or heating on high. To manage these trade-offs, EVs will need to rely on efficient software systems to coordinate across domains.

Connectivity inside a vehicle is also paramount. Although software has been the critical enabler of today's infotainment capabilities, new complications are arising. For instance, drivers must now deal with clunky, lag-filled interfaces that are slow to respond to their inputs, since infotainment systems must wait to receive information from other components. There are also disconnects (such as a lack of coordinated information between instrument clusters and center screens on some vehicle models) and an absence of visual consistency for indications and alerts. These issues are not only distracting but also disappointing to consumers accustomed to streamlined, user-friendly interfaces in mobile phones and tablets—devices that cost far less than a car does.

## A better path forward

Imagine a world in which software throughout a vehicle was truly integrated end to end. A primary operating system, robust and flexible enough to cover major systems throughout the vehicle, and software modules, developed on a common code base, could anchor this integration. Such a construct would provide a solution to many of the pain points present in today's fragmented ecosystem.

First, this construct would allow automotive players to address the performance issues stemming from disparate operating systems and disjointed sets of code. Cross-domain interfaces could become far simpler, since different systems could directly "talk" to each other without translation dragging down efficiency and introducing delays. Security could greatly expand, enabling one overarching solution to monitor the full code base—not just at a handful of interfaces. And development productivity could considerably improve, with OEMs adding new cross-domain features over time, without extensive software rewrites, by leveraging a common code base throughout the vehicle.

End-to-end software solutions could also lay the groundwork for a substantially enhanced human–machine interface. Fundamental user-experience constructs could easily be shared across vehicle domains, enabling output methods, such as the center screen, instrument cluster, and heads-up display, to have a consistent behavior, look, and feel. Automotive players could enhance safety, with real-time notifications triggering immediate visual, auditory, and tactile feedback, throughout the cockpit, that alerts drivers to act. As interior-design preferences evolve and user interfaces are repurposed for different vehicle models and tiers, the sensory experience could change with minimal development effort.

Finally, end-to-end software platforms could make dynamic resource sharing a reality—a shift that would reduce overall hardware costs while enabling the addition of new capabilities over time. Supporting this trend, OEMs appear to be moving toward an approach in which in-vehicle communications align with an Ethernet standard. This move will allow OEMs to create vehicles with a real-time, high-bandwidth information highway that links major systems together and opens the door for dynamic sharing of the full resource pool across a car. A unified operating system could then allow processing horsepower to be allocated on the fly, dynamically enhancing performance for critical systems when needed. For instance, more resources could be allocated to ADAS during "edge cases" for which complexity is high and response speed is paramount. Overall hardware costs could be optimized, and additional processing capacity could simply be "plugged in" to the architecture to support future applications.

## Taking the first steps: Options for automotive-industry stakeholders

The shift in software perspective, from discrete modules tailored for specific functions to a common architecture capable of effectively hosting code for each major system, has already occurred in other industries. For instance, several drone developers have moved from individual operating systems for flight control, wireless communication, and camera to a single platform that integrates code across the device. This change reduced development complexity and enabled new cross-domain functionality, such as flight-propulsion systems that use camera data for navigation and obstacle avoidance. Smartphones have undergone a similar shift, with leading players finding ways to improve functionality by more closely integrating code across existing hardware components. These players have also found ways to reuse a single software architecture across many types of devices, substantially reducing the need for redesign.

We are beginning to see similar shifts in the automotive industry, with a few tech-focused players recently taking the first steps at integrating software elements into combined platforms (Exhibit 3). Consider the following:

— A large smartphone-software player with an established position in infotainment is now moving into hypervisors to enable the safety-critical driver communication needed for today's ADAS.

— A provider of real-time ADAS operating systems is expanding into infotainment offerings, creating a safety-critical solution for driver notifications.

— A major tech player with a cloud-services platform is expanding into applications, bundling a tailored software suite for cars.

These efforts represent a step in the right direction, although they still fall short of the comprehensive integration required to make an end-to-end platform viable.

## Routes to a common automotive-software platform

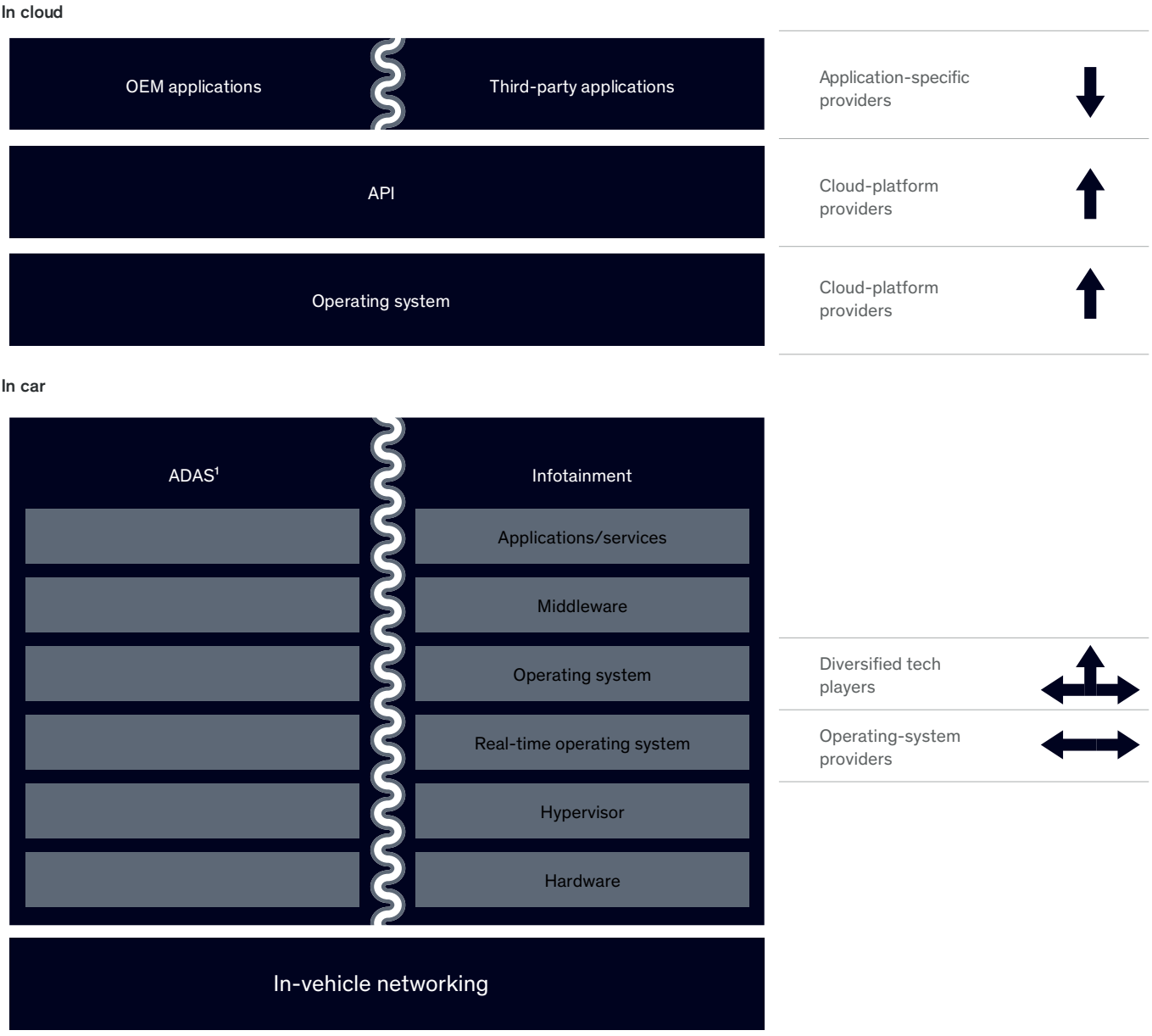
Each major player within the automotive space, from OEMs to established tier-one suppliers and tech companies, could potentially create its own end-to-end software platform in isolation. But there is also a good reason for each of them to avoid solo efforts. For OEMs, the issue is capability. Each company has certain software-development strengths—for instance, it might produce a strong ADAS—but no player is a standout expert in all vehicle domains. Building the required skills would be costly, difficult, and time consuming. Tech companies might be stronger in software, but they lack the hardware platform—in other words, vehicles—on which to deploy solutions. For established tier-one suppliers, the burdens fall in both areas: a lack of comprehensive software expertise and the absence of a hardware platform for deployment.

The industry has many technical and commercial dynamics, but early signs point to the following possible routes.

Exhibit 3

## Software providers are exploring moves across the stack to broaden offerings and gain the benefits of integration.

**Movement in the automotive stack** (illustrative example, not exhaustive)




[1] Advanced driver-assistance system.

**OEMs band together for joint development**

OEMs might compete for market share, but partnerships among traditional rivals are well known in the space, especially when companies have a common interest. For instance, three German OEMs formed a consortium to acquire a mapping business, then worked together to improve the system for automotive use. Each company occasionally added proprietary mapping features, such as improved user interfaces, to its own cars to provide a competitive advantage. A similar collaboration framework could work when developing an end-to-end software platform. Partners would develop a core operating system and middleware set, but each OEM could add proprietary code for its own vehicles via well-defined interfaces.

Additionally, OEM collaborations would decrease reliance on tech powerhouses for software. That is an important benefit, since OEMs fear becoming low-margin hardware players—a fate that has befallen many mobile companies that adopted outside software platforms. Quality is also an issue, since even the strongest platforms from tech players might have features that disappoint or annoy customers, such as the presence of advertisements on infotainment systems. OEMs might have little leeway to ask for changes in the platform, especially if the tech company is providing it for free.

OEM collaborations will not automatically produce excellent software, even if the companies involved have complementary skills. To succeed, the participating players must follow strong software-design principles. For instance, they should allow multiple developers to contribute simultaneously and make rapid improvements. They will also need agile development frameworks, top talent, and efficient governance systems.

Most collaborations will probably be small at first, involving only two or three OEMs, but other players might become interested if an alliance creates core components with attractive features. The same pattern occurred with the German OEM collaboration, with other major OEMs licensing mapping data from the original three partners after their initial success. We may also see a bifurcated approach, with premium OEMs taking an active role in shaping platform features, while volume OEMs rely more on commonality and cost-sharing models.

**Leading tech players make strong acquisitions in the automotive space**

With most OEMs preferring to remain independent from tech players, alliances between these two groups are likely to remain tepid, at least over the short term. Even if a leading technology company develops an innovative end-to-end software platform, OEMs might be reluctant to deploy it across their fleets. But it is possible that a tech player with a diversified interest in automotive components could instead acquire an OEM, or even a leading tier-one supplier, creating a new entity that aligns the interests of both players.

Such acquisitions could give a tech company the ability to infuse its technology through a player already well established in the automotive value chain. In addition to enabling the deployment of an end-to-end software platform, this partnership could provide the tech company with complete control over hardware and software integration—a common pain point that often results in delays or suboptimal performance in the current development process.

While the acquisition of an OEM or leading tier-one supplier could be a radical move in the tech industry, such bold deals are not unprecedented. A major tech player recently acquired a tier-one infotainment provider, for instance. And two tech companies undertook a similar approach with smartphones over the past decade, acquiring handset players and leveraging their hardware to stimulate market demand for new operating systems and applications.

**Disruptive OEMs take a software-based approach to car design**

Although established players have strong incentives to maximize reuse of current electronics architectures, new entrants can take a ground-up approach to vehicle development while reshaping the construct for supplier engagement. They would deploy a unified software architecture from the start,

providing the foundation for new cross-domain features and updates. This approach would also reduce overall system complexity and development costs, with suppliers providing native compatibility with the architecture base across each component they provide.

A leading EV OEM has embarked on this path, designing its electronic systems to share a common software foundation. Nearly all the systems can seamlessly communicate with each other and receive updates over the air. New software-defined features are deployed as they become ready and can be further improved over time. Security and safety vulnerabilities can be rapidly identified and addressed. Development efforts can be streamlined over the long term, with teams efficiently executing improvements rather than battling a maze of system interfaces for minor changes. Performance and usability data are even collected across the fleet and fed back into R&D to inform future features and help prioritize projects.

Established OEMs are likely to take note of this proof point and could follow a similar model for their next vehicle platforms. But they need not bring all development in house. Shifting to a software-centric mind-set could simply mean defining and maintaining the core end-to-end foundation and insisting upon adherence as part of the supplier-relationship construct. Although this represents a fundamental shift in approach, it is likely to prove to be a critical lever against the threat of disruptive OEMs.

---

The path to an end-to-end platform will be complex, with some stakeholders assuming new roles or venturing into new software areas, and there is still much uncertainty ahead. Over the next five to seven years, consolidation of today's complex vehicle architecture into even two or three leading platforms could represent a viable first step. But one fact is clear: the potential value that increased integration could unlock for OEMs and suppliers is immense, proving the adage that the whole is greater than the sum of its parts.

**Ryan Fletcher** is an associate partner in McKinsey's Southern California office, where **Abhijit Mahindroo** is a partner; **Nick Santhanam** is a senior partner in the Silicon Valley office; and **Andreas Tschiesner** is a senior partner in the Munich office.

Designed by Global Editorial Services