WHITE PAPER

Multi-source Automotive Software Stacks

Understanding the software solution landscape for the software-defined vehicle

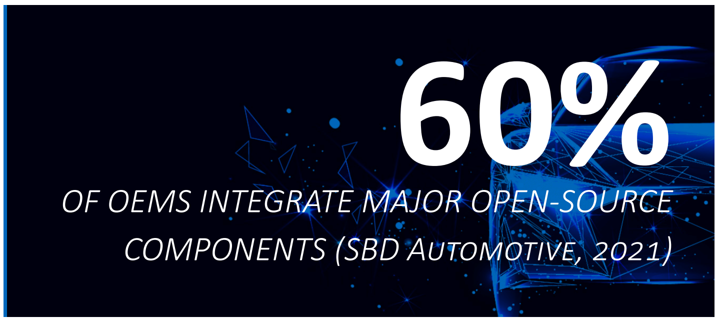
Third-party software plays a key role in the modern vehicle

The automotive software solution landscape is undergoing a fundamental shift as the industry moves towards software-defined vehicles. The amount of software integrated into vehicles is growing quickly as architectures move away from function-specific control units to central and zonal controllers that create virtual machines, using hypervisors and/or containers to combine the functionality of multiple ECUs. Virtualization in software is the key to the architectural enablement of the software-defined vehicle and enables mixed-criticality systems such as infotainment and safety cameras to share the same hardware.

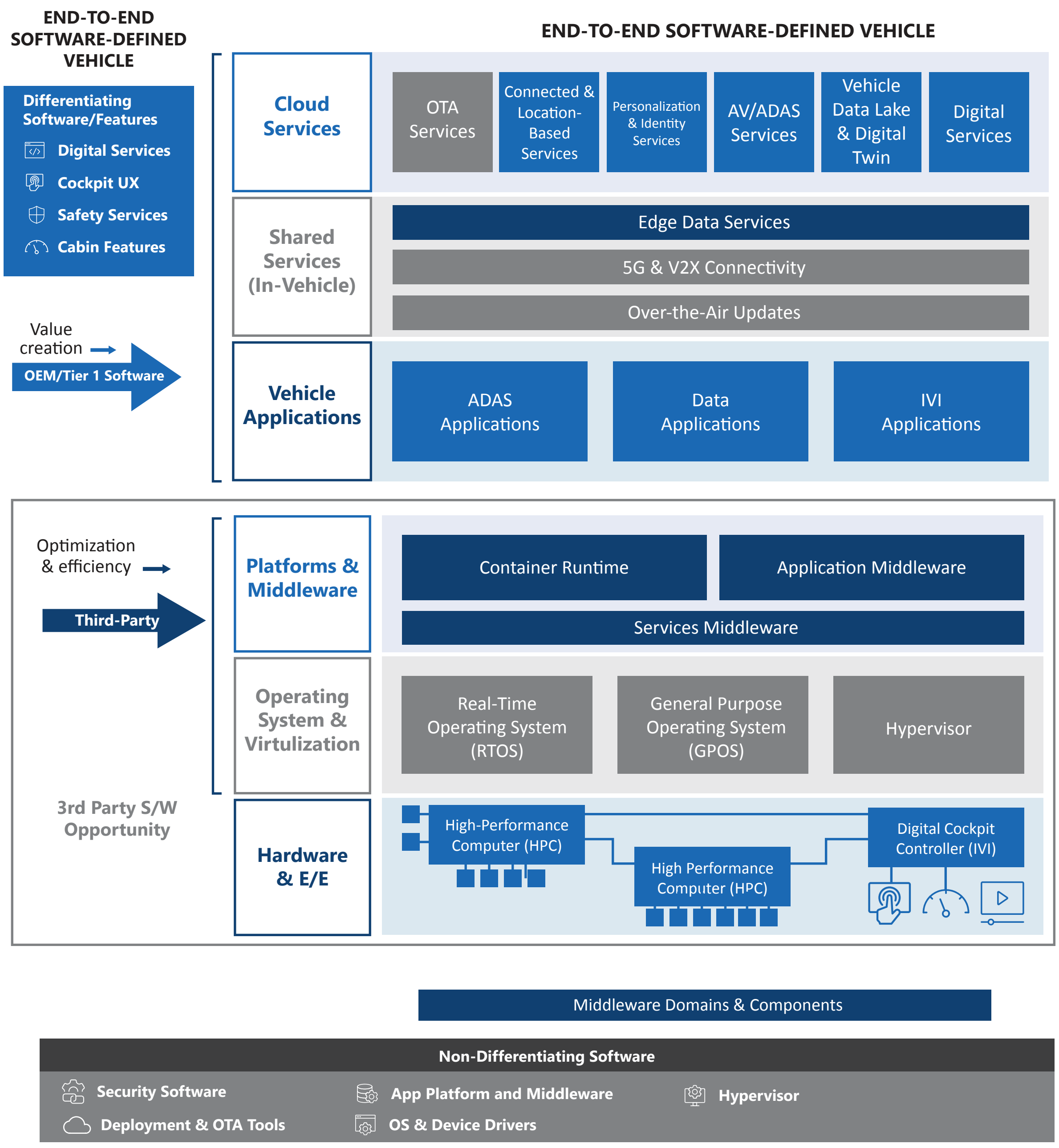
100%
OF OEMS INTEGRATE MAJOR VENDOR SOFTWARE COMPONENTS

OEMs such as Tesla, BMW, VW, Mercedes, Stellantis, and Volvo have all announced initiatives to develop their own software base as part of this shift. But do traditional vertical stack software development paradigms scale enough to allow OEMs to do what they want to do? Will integrating third-party software into those OEM stacks help make that scaling possible?

One of the big challenges that comes with this seismic change is that the global industry is trying to access a very limited resource (developers). Going forward, all OEMs and Tier 1s cannot each develop their own unique software stack – it is just not a sustainable model for the industry as a whole. Instead, they are combining their own developments with code and artifacts from third parties – including commercial off-the-shelf products, open-source code, and commercial implementations of open standards. But how do they decide what kind of software to put where, and does it really matter whether software is commercially managed or community managed?


60%
OF OEMS INTEGRATE MAJOR OPEN-SOURCE COMPONENTS (SBD AUTOMOTIVE, 2021)

The continued move towards features defined by software rather than hardware will bring even more code into the car. A large part of that software, though, is not visibly differentiating to consumers. Software that provides the foundational layers of the software-defined vehicle, from drivers and board-support packages to operating systems, virtualization and middleware layers may differentiate **among themselves** for what they allow OEMs and Tier 1s to develop, as shown in Figure 1. This is the part of the software stack where third-party software, or software developed outside the OEM/Tier 1, offers the greatest value to the industry. Software that works across manufacturers and components maximize commonality at the enabler level, allowing the feature-focused software developers to concentrate their time on consumer-facing functionality and brand differentiation.



END-TO-END SOFTWARE-DEFINED VEHICLE
Differentiating Software/Features
Digital Services
Cockpit UX
Safety Services
Cabin Features
Value creation
OEM/Tier 1 Software
END-TO-END SOFTWARE-DEFINED VEHICLE
Cloud Services
OTA Services
Connected & Location-Based Services
Personalization & Identity Services
AV/ADAS Services
Vehicle Data Lake & Digital Twin
Digital Services
Shared Services (In-Vehicle)
Edge Data Services
5G & V2X Connectivity
Over-the-Air Updates
Vehicle Applications
ADAS Applications
Data Applications
IVI Applications
Optimization & efficiency
Third-Party
Platforms & Middleware
Container Runtime
Application Middleware
Services Middleware
Operating System & Virtulization
Real-Time Operating System (RTOS)
General Purpose Operating System (GPOS)
Hypervisor
3rd Party S/W Opportunity
Hardware & E/E
High-Performance Computer (HPC)
High Performance Computer (HPC)
Digital Cockpit Controller (IVI)
Middleware Domains & Components
Non-Differentiating Software
Security Software
Deployment & OTA Tools
App Platform and Middleware
OS & Device Drivers
Hypervisor

Figure 1: OEMs and Tier 1s have an opportunity to integrate a range of third-party software to improve their ability to scale their software stack

All OEMs integrate some level of vendor software into the lower half of their software stack, whether it is managing network traffic, enabling virtualization, at the OS level, or in middleware. At the same time, 60% of OEMs reviewed by SBD Automotive integrate significant community managed components at the OS level such as Linux® or Android™ – particularly within infotainment – and many integrate commercially managed implementations of open platforms and code (SBD Automotive, 2021).

Given the global ceiling on developer availability and the relatively high proportion of non-differentiating software in a vehicle, OEMs and Tier 1s can scale their software development most efficiently when they combine their internally developed differentiating software with a mixture of commercially managed and community managed software at the enabler levels.

Community managed versus commercially managed software – are they actually in competition?

The third-party software in cars today covers a variety of systems, mission criticality, and functional use cases. Third-party components range from cross-industry open-source products such as Android OS and Linux to automotive-specific open platforms implemented on a commercial basis, such as Classic and Adaptive AUTOSAR, to commercially managed, automotive-focused products such as the QNX® Neutrino® Real-Time Operating System and the QNX® Hypervisor from BlackBerry and Wind River VxWorks. By digging into automotive system and software architectures, we can see patterns emerging of how different third-party contributions complement each other to deliver software-defined functionality.

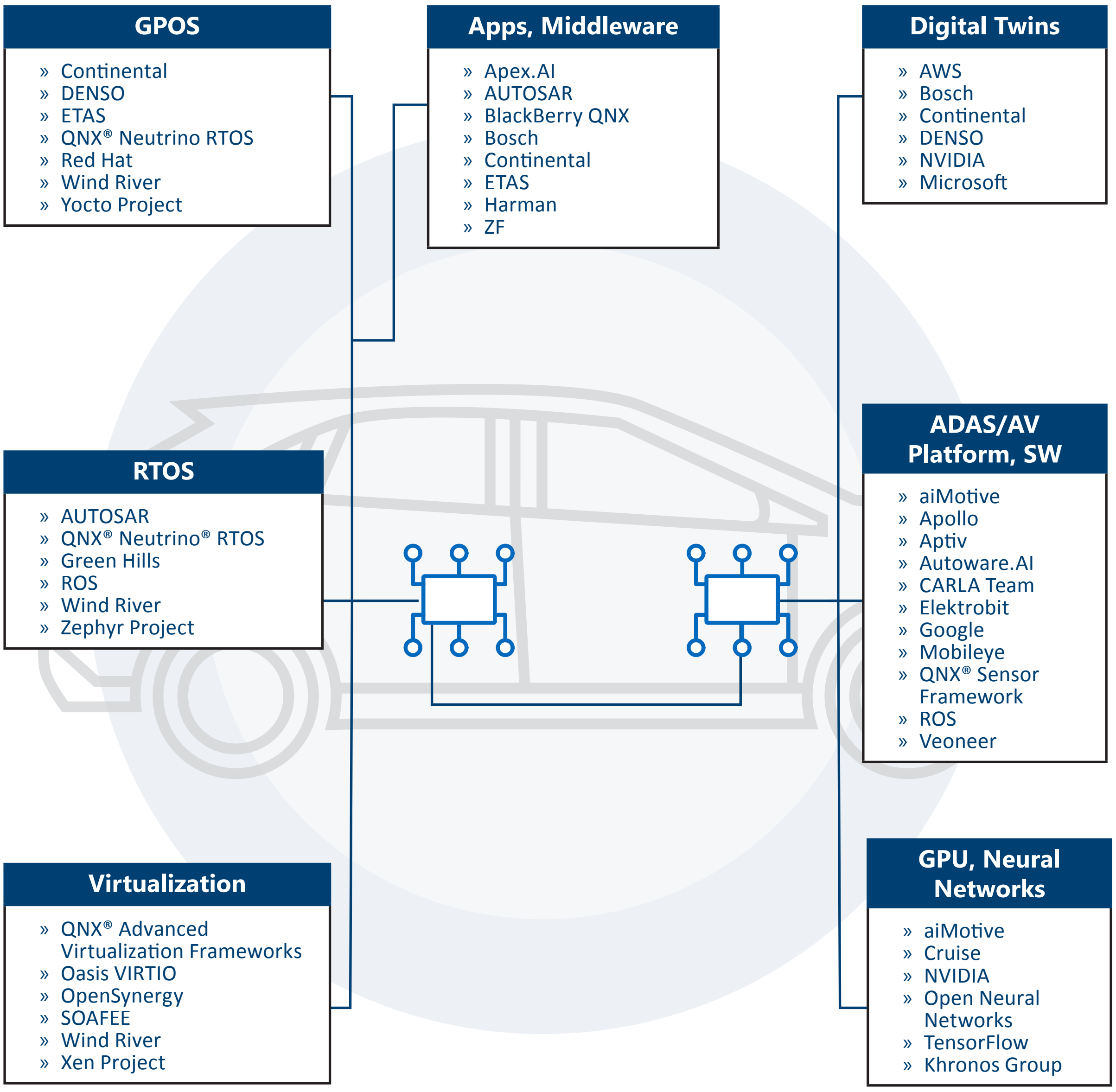


Figure 2: OEMs and Tier 1s will integrate many components from the range of available third-party software solution options

Different approaches to making the "where should I source my software" decision play out in today's market, exemplified by three broad company personae:

The "Integrators": OEMs or leading Tier 1s integrate a range of third-party platforms and software components with their own differentiating layers. These developments tend to rely heavily on commercially managed solutions to complement the internal development, such as those shown in Figure 2.

The "Slow-growers": These companies invest heavily and carefully in core technology enablers for some domains, taking time to develop their own "home-grown" technology stack. These companies, invest significant time and money in standards-based or open-source software that provide the majority of its non-differentiating layers, but still integrate a range of commercially managed software to get the best blend of functionality to suit their strategies.

The "Need-it-Nows": These manufacturers want a fast turnaround on their development effort. They are more likely to combine community managed software like Linux or Android for fast-change domains like infotainment with mature commercially managed solutions such as QNX for the slower-cycle and quality-critical areas such as safety/control systems.

All three manufacturer types show a tendency to integrate commercially managed software for safety-critical components and systems and as virtualization layers to enable mixed-criticality systems - particularly where developing an alternative to the existing solutions requires a long-term investment to create something a third party already has on the shelf. Open-source software, where integrated, is typically focused on non-safety systems such as infotainment or in a few cases as a foundation for own-built software.

All software needs to be managed

Given all the different software sources coming into modern vehicles, OEMs are now also faced with the challenge to manage and monitor their software inventory. With over-the-air updates allowing increasingly frequent software revisions to deliver both quality improvements and new functionality, vehicle lifecycles are quickly diverging from software lifecycles. Whether software is home-grown, commercially managed, or community managed, someone in the vehicle ecosystem has to look after the specific implementations that are on the roads for as long as they are active.

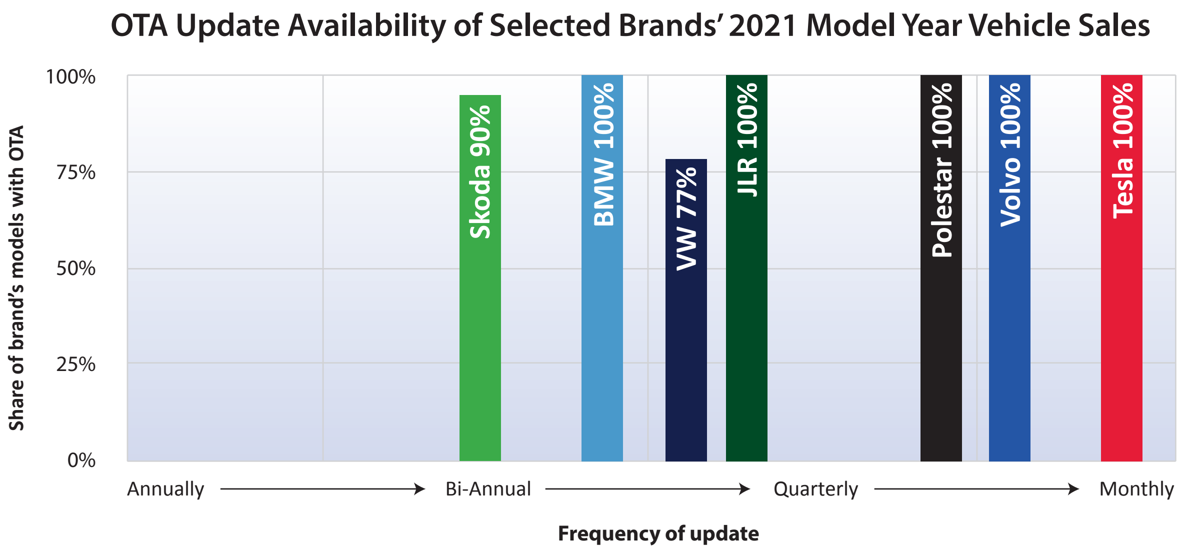


Figure 3: OTA updates enable an update cadence that will test automotive software management systems
Source: SBD Automotive OTA & SW-Delivered Features Guide EU: Q1 2022

Even updating annually presents OEMs with multiple third-party software components with traceability and maintainability issues. But as the industry moves towards monthly updates across different domains and functionalities or sees periods of high-frequency cybersecurity patching due to non-automotive factors, the ability to maintain a cohesive software integration will be tested even further.

Global automotive regulations such as UNECE R155 and China's GB/T series require manufacturers and suppliers to provide an extensive software bill of materials (SBOM) on new vehicle types (UNECE WP.29, 2020) (Biden, 2021), (General Office of the Ministry of Industry and Information Technology, 2022). So, regardless of persona or software sourcing strategy, manufacturers need more from their software enabler ecosystem than just the software. Integrators at OEMs and Tier 1s increasingly rely on software composition analysis tools such as BlackBerry® Jarvis® and Black Duck® to scan and understand the content of binary images going into cars.

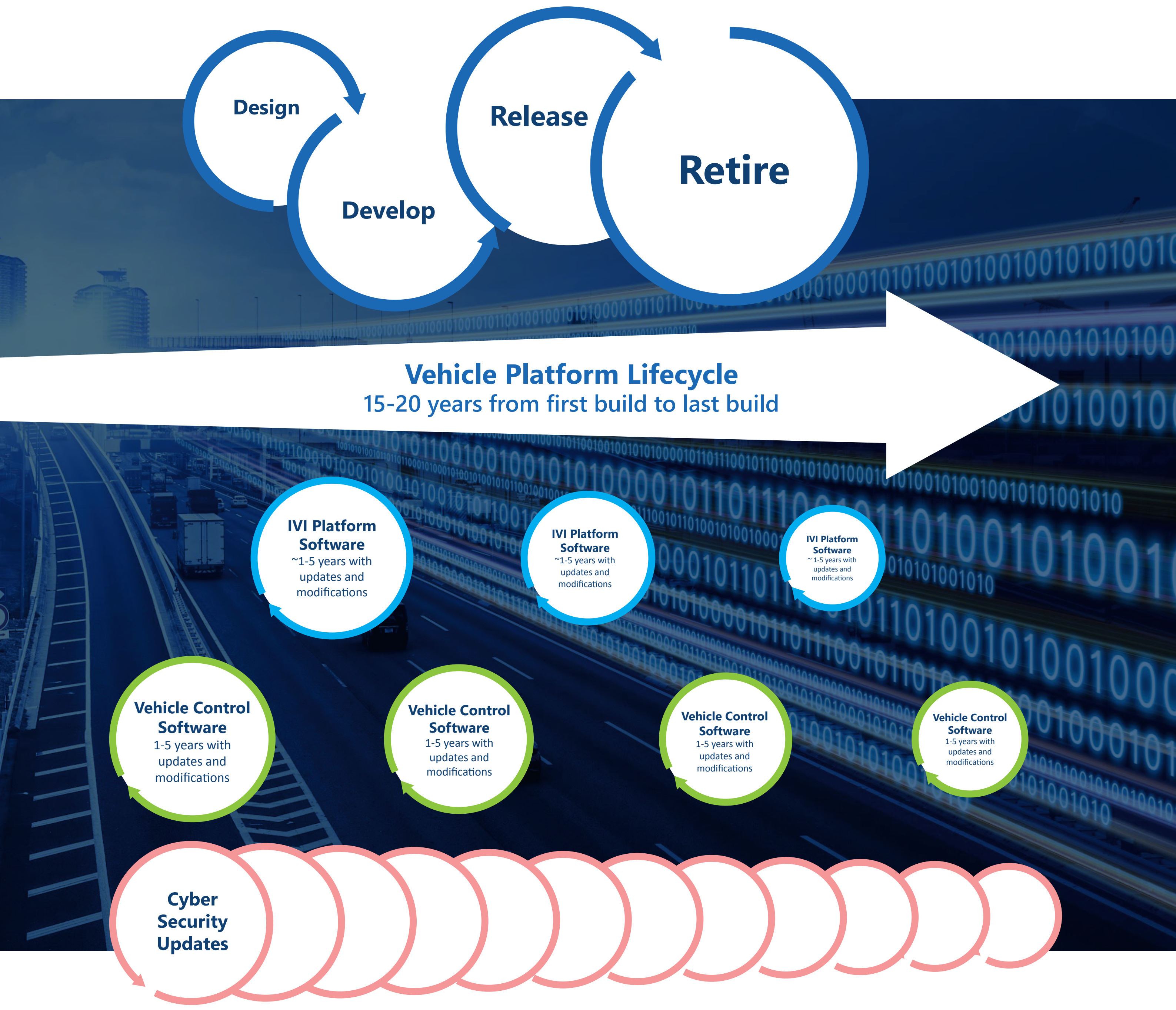

Design
Develop
Release
Retire
Vehicle Platform Lifecycle
15-20 years from first build to last build
IVI Platform Software ~1-5 years with updates and modifications
IVI Platform Software ~1-5 years with updates and modifications
IVI Platform Software ~ 1-5 years with updates and modifications
Vehicle Control Software 1-5 years with updates and modifications
Vehicle Control Software 1-5 years with updates and modifications
Vehicle Control Software 1-5 years with updates and modifications
Vehicle Control Software 1-5 years with updates and modifications
Cyber Security Updates

Figure 4: Vehicle platforms lifecycles encapsulate several different software lifecycles

The combination of commercially managed and community managed software means that even "proprietary" software can have significant levels of open-source components, and many open-source components are only actually viable in vehicles over a full lifecycle when managed either by the OEM, a Tier 1, or a commercial software partner. As they are blended into mixed-criticality systems, certainty of what is "in the box" becomes even more important.

The differentials among lifecycles amplify the complexity of understanding what is in the vehicle at any time and comparing that with what should be there. OEMs and their key software providers and integrators are implementing processes to manage this complexity. Software vendors, who generally have a clear product release model, are well-placed to support this non-functional regulatory requirement. For example, the OpenChain ISO / IEC 5230:2020 standard provides a self-certification framework to help companies (OEMs and suppliers) demonstrate that their license compliance programs include all relevant artefacts and materials governed by open-source licensing. BlackBerry was the first North American company to gain OpenChain accreditation across its entire product portfolio.

For open-source groups and developer communities, this combination of component traceability and different product lifecycles within a single vehicle offer a significant challenge. The number of software recalls issued by U.S. NHTSA continues to grow. Recalls require no-cost repair for vehicles up to 10 years old, but recall actions can extend to vehicles well beyond that age. Someone, somewhere, has to maintain those software components for much longer than the originating developers typically commit to doing.

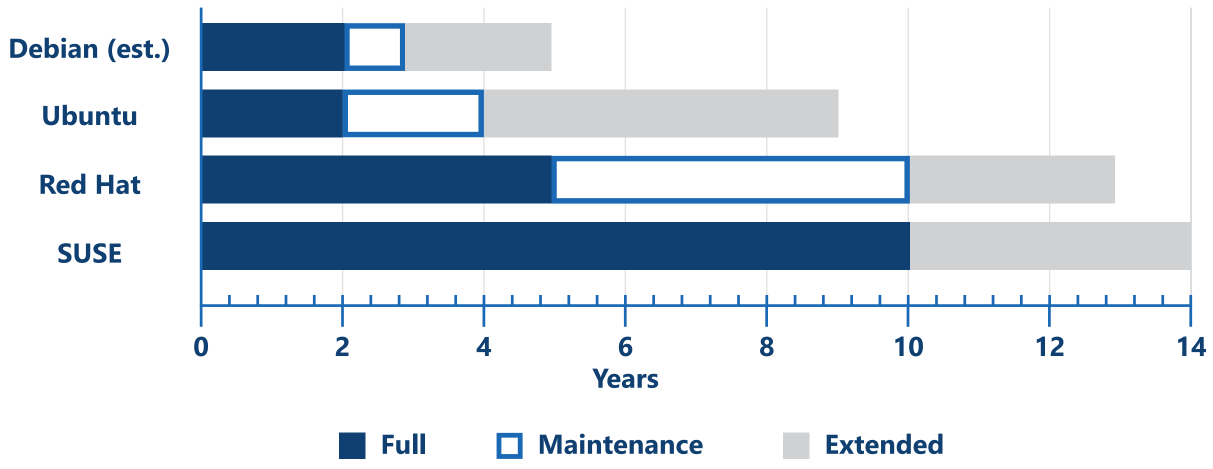


Figure 5: Long-term support among open-source OS may not always match automotive needs

To meet long vehicle platform lifecycles, software needs long-term support versions, which will receive updates beyond the 6-month to 2-year support typical in consumer electronics. The open-source community has established extended support for key software packages, such as the Civil Infrastructure Platform's goal of 10 years or more for the Linux kernel (The Civil Infrastructure Platform). However, such projects do not yet have the proven history that automotive-focused software vendors have provided since the 1990s. OEMs or Tier 1s who are blending software from different sources across mixed-criticality systems will pick up the long-term management of community managed software components, at least for the next several years.

Both commercially and community managed software will continue to adapt to market demands, fueled by the increased level of software both in vehicles and linked to them. Mixed-source software will continue to blend different strengths to provide solutions for vehicles, systems, and individual

ECUs. OEMs and Tier 1s will invest in their futures, selecting commercial partners and community managed code to deliver architectures with support from multiple third parties. Virtualization will become increasingly common to allow these software components to run together on mixed-criticality systems.

Whether the full vehicle software stack is designed at an OEM or a trusted Tier 1 partner, best-in-class software partnerships across the third-party solution landscape will be needed to create, maintain and improve software for the life of the vehicle in a way that meets customer and regulatory expectations.

Case Study: BlackBerry QNX

QNX software has been a part of the embedded automotive ecosystem for more than 20 years, predating its ownership by BlackBerry. Over that time, the QNX Neutrino operating system has formed a key part of the software strategy for most OEMs at some point in their own journey towards a software-defined vehicle. The OS enables functionality across domains, and is well-known for use in telematics, infotainment, info-safety, and increasingly, ADAS. These products operate in over 195 million vehicles on the road today (BlackBerry QNX, 2021). This history and breadth of experience offers the company a good view of the challenges the automotive embedded software industry will face in the next 5-10 years.

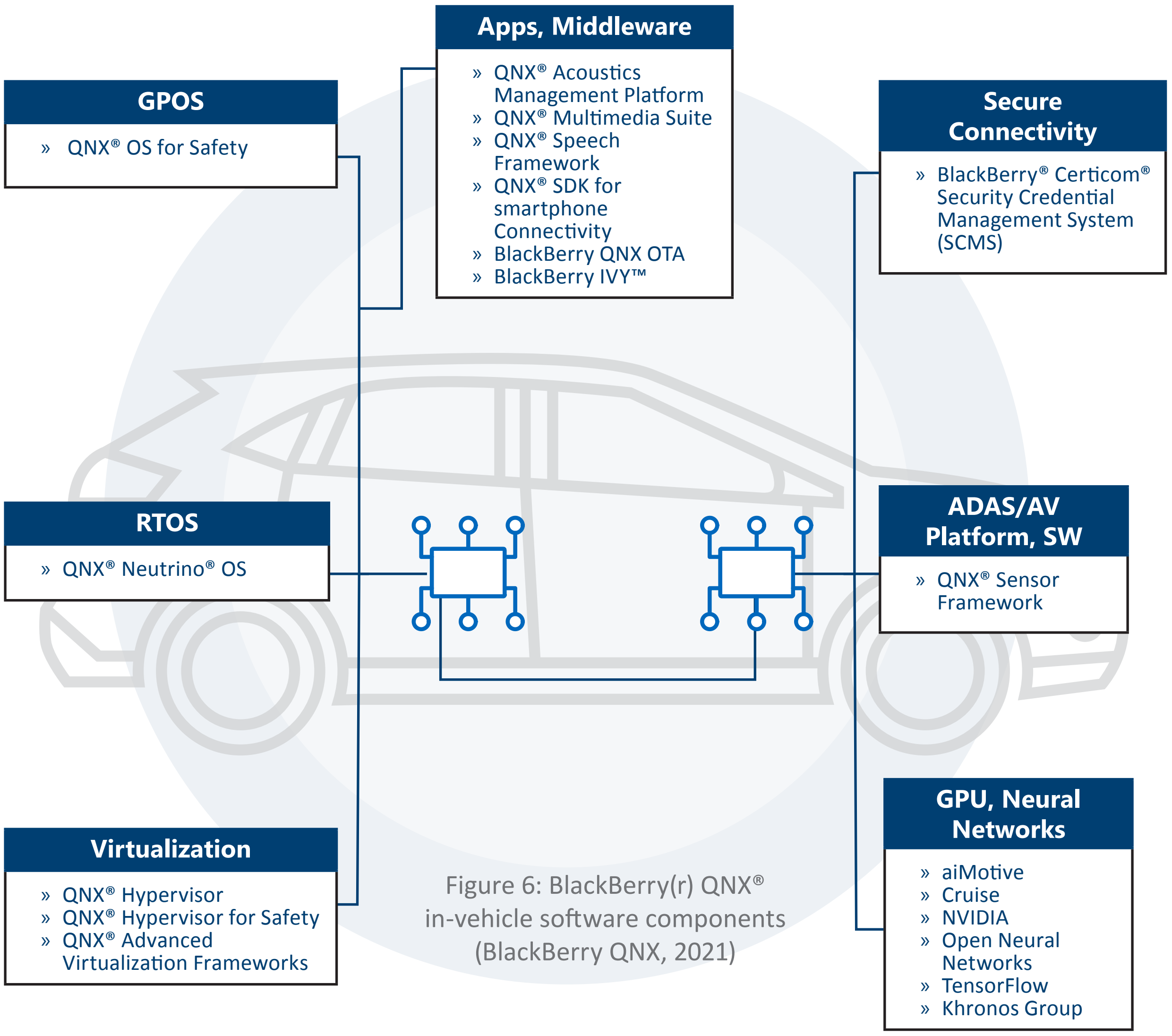
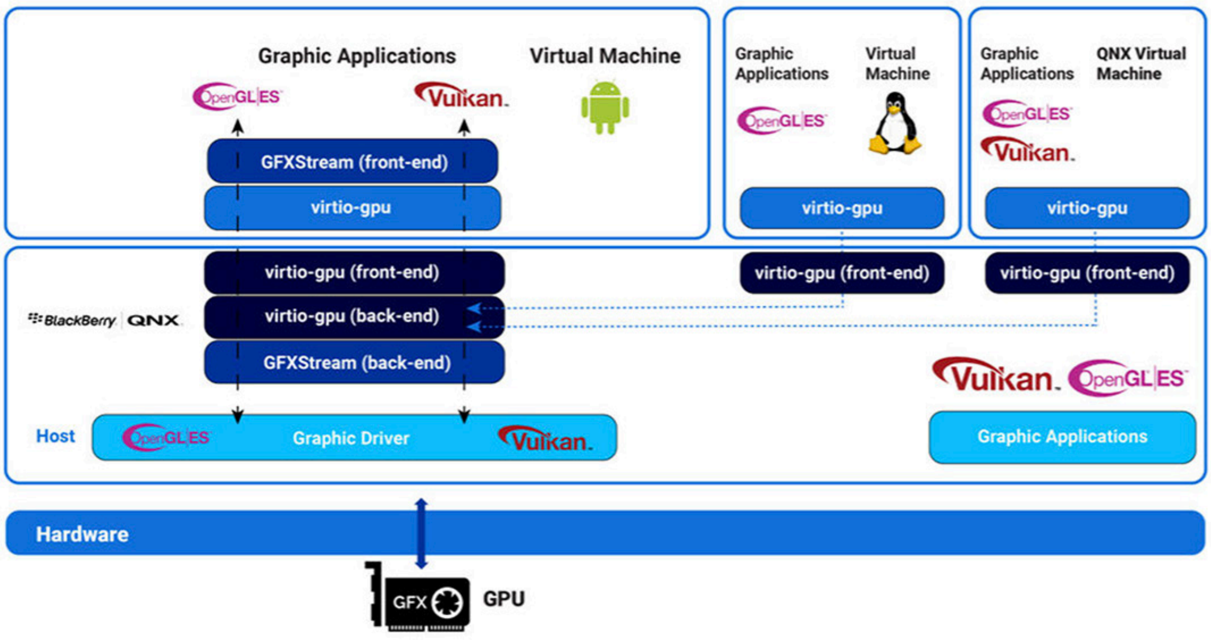


Figure 6: BlackBerry(r) QNX® in-vehicle software components (BlackBerry QNX, 2021)

Today, as architectures move away from function-focused ECUs to high-performance centralized and/or zonal controllers, the company has identified a key need among OEMs for reliable software components and enablers off-the-shelf that can operate effectively in a mixed-criticality system, regardless of its source, as shown in Figure 6. Bringing together functionality for infotainment functions such as media streaming and navigation and info-safety displays such as clusters and head-up displays has challenged the embedded software ecosystem for some time. Within the ADAS domains, running low-criticality functions like lane-keeping assistance and high-criticality functions like piloted driving alongside each other can be equally complex.



Graphics Virtualization using virtio-gpu and GFXStream
Graphics Applications in Host are rendered immediately after boot

Figure 7: QNX Hypervisor with Advanced Virtualization Frameworks for Android, Linux and QNX guests supporting a range of third-party software components in an automotive software environment (BlackBerry QNX)

Pouring all of those into a single multi-domain, centralized compute resource offers significant potential savings by reducing the number of high-powered ECUs in the vehicle, but it also increases conceptual complexity in that the different domains and functions have very different reliability and performance requirements. The QNX virtualization capability, as shown in Figure 7, helps OEMs and Tier 1s approach the complexity of blending commercially and community managed software to enable mixed-criticality systems that share the same hardware components. For example, by following open standards like VIRTIO, infotainment services based on Android or Linux can share devices like displays and cameras with safety-focused functions like reversing cameras or lane-keeping warnings.

[1] OpenGL is a registered trademark, and the OpenGL ES logo is a trademark of Hewlett Packard Enterprise used by permission by Khronos.
[2] Vulkan and the Vulkan logo are registered trademarks of the Khronos Group Inc.
[3] The Android robot is reproduced or modified from work created and shared by Google and used according to terms described in the Creative Commons 3.0 Attribution License.

How to decide what type of third-party software is most appropriate for a specific component or function

OEMs and Tier 1s need to combine the best of both community managed and commercially managed software to create a full software stack (even if they're making their own).

To find your way through the options of what software to integrate and how you might need to blend commercially and community managed components, it can help to understand where you fit best among the three personae ("Integrator", "Slow-grower", or "Need-it-Now") and apply that approach to the following questions.

Reliability:

» Does the software component have the proofs/guarantees your organization or intended use requires?

» Does the software component meet internal and regulatory safety, stability, recoverability, and cybersecurity requirements in line with the relevant system criticality?

Financial:

» Does the software component meet both initial and long-term cost targets for usage, internal development and maintenance resources (are external companies looking after it for the long haul, or is that an internal cost)?

» Does the hardware implication of the software component match your hardware strategy and cost target?

Service and Support Outlook:

» Does the software component provider or another identified organization (internal or external) have an understood plan for updating, maintaining, and patching the software for its entire intended lifespan in the vehicle?

» Does the upgrade frequency and issue responsiveness meet your needs?

Time-to-market

» Does the software do what you need it to for your time-to-market targets?

Differentiation

» Does the software component offer you an opportunity to differentiate?

» Is the software component significantly different to other available alternatives?

If the answers to the above questions are typically yes, then the software component you are looking at is likely to be appropriate, regardless of its source. If it is no, then the persona alignment gives an indication of the right course of action:

Persona	Potential approach to turn “No” into “Yes”
Integrator	Look for commercially managed software to integrate
Slow-grower	Target internal + open-source / standardized development, and bridge any time gaps with off-the-shelf software from a commercial partner
Need-it-Now	Select the fastest turnaround / most flexible software source, for example, internal management of open-source components for IVI and non-critical functions and commercially managed software for safety, security, and critical functions.

While each organization makes choices differently based on a combination of short-term functional needs and long-term software support requirements, those choices will reflect their overall approach to software development and ownership. Whether the full vehicle software stack is designed at an OEM or by a trusted Tier 1 partner, best-in-class components from a range of third-party solutions are integral to creating, maintaining and improving software at scale across the automotive industry in all vehicle domains and for the full vehicle lifecycle. In today’s market, mixed-criticality systems rely on commercially managed virtualization solutions to ensure the blended software components run smoothly alongside each other. OEMs require the same long-term maintenance expectations for their hypervisors as for their other key software enablers. In a world of software-defined functionality and mixed-criticality systems, the software has to work like new for as long as the cars are on the road. So, unless an OEM or Tier 1 wants to maintain its virtualization solutions for 15+ years in the same way they will look after their application software, the commercially managed hypervisor route is a cost-effective way for carmakers to manage risk when deploying mixed-criticality vehicle systems.

References

Apex.AI. (2021). Toyota creates a visionary vehicle OS, Arene. Retrieved from Apex.AI: **https://www.apex.ai/toyota-woven-planet**

Automotive Grade Linux. (2018, January 18). Automotive Grade Linux Hits the Road Globally with Toyota. Retrieved from PR Newswire: **https://www.prnewswire.com/news-releases/automotive-grade-linux-hits-the-road-globally-with-toyota-amazon-alexa-joins-agl-to-support-voice-recognition-300580361.html**

Baker, E. (2022, February 8). High security: Interview with Adam Boulton, chief technology officer, BlackBerry Technology Solutions. Retrieved from Automotive Testing Technology International: **https://www.automotivetestingtechnologyinternational.com/features/high-security-interview-with-adam-boulton-chief-technology-officer-for-blackberry-technology-solutions.html**

Biden, J. R. (2021, May 12). Executive Order on Improving the Nation's Cybersecurity. The White House. Retrieved from **https://www.whitehouse.gov/briefing-room/presidential-actions/2021/05/12/executive-order-on-improving-the-nations-cybersecurity/**

BlackBerry. (2020, August 24). BlackBerry Teams Up with Desay SV Automotive to Enable an Intelligent and Safe Driving Experience for the new Xpeng P7- a Leading Level 3 Autonomous EV. Retrieved from BlackBerry: **https://www.blackberry.com/us/en/company/newsroom/press-releases/2020/blackberry-teams-up-with-desay-sv-automotive-to-enable-an-intelligent-and-safe-driving-experience-for-the-new-xpeng-p7-a-leading-level-3-autonomous-ev**

BlackBerry QNX. (2021, April 19). Foundational Software Solutions for the Modern Vehicle.

BlackBerry QNX. (n.d.). QNX Advanced Virtualization Frameworks. Retrieved from BlackBerry QNX: **https://blackberry.qnx.com/en/products/foundation-software/qnx-hypervisor/advanced-virtualization-frameworks**

General Office of the Ministry of Industry and Information Technology. (2022, March 7). Notice of the General Office of the Ministry of Industry and Information Technology on Issuing the Guidelines for the Construction of the Internet of Vehicles Network Security and Data Security Standard System.

Lambert, F. (2016, October 6). Tesla vehicles to get a much-needed updated browser with new Linux OS in December, says Musk. Retrieved from Elektrek: **https://electrek.co/2016/10/06/tesla-updated-browser-linux-os-december-says-musk**

Li Auto. (2021, August). FORM 6-K/A. Retrieved from Li Auto: **https://ir.lixiang.com/static-files/b184f5fe-c7dd-4815-85c2-f3d75d45cfa4**

Man, H. (2020, June 22). Volkswagen to develop in-house software for next-gen cars. Retrieved from CarExpert: **https://www.carexpert.com.au/car-news/volkswagen-to-develop-in-house-infotainment-software**

SBD Automotive. (2021). The business, technology and supply chain of The Software-defined Vehicle. SBD Automotive.

SBD Automotive. (2022). SBD Automotive OTA & SW-Delivered Features Guide EU: Q1 2022.

Stellantis. (2021, December 7). Stellantis Software Day. Retrieved from Stellantis: **https://www.stellantis.com/content/dam/stellantis-corporate/investors/events/stellantis-sw-day/Software_Day_2021_Presentation_final.pdf**

Stellantis. (2022, January 5). Amazon and Stellantis Collaborate to Introduce Customer-Centric Connected Experiences Across Millions of Vehicles, Helping Accelerate Stellantis' Software Transformation. Retrieved from Stellantis: **https://www.stellantis.com/en/news/press-releases/2022/january/amazon-stellantis-collaborate-on-software-solutions**

The Civil Infrastructure Platform. (n.d.). The Civil Infrastructure Platform FAQ. Retrieved from The Linux Foundatino: **https://www.cip-project.org/faq**

UNECE WP.29. (2020, March 3). Draft new UN Regulation on uniform provisions concerning the approval of vehicles with regard to cyber security and of their cybersecurity management systems. ECE/TRANS/WP.29/GRVA. Retrieved from **https://unece.org/fileadmin/DAM/trans/doc/2020/wp29grva/GRVA-06-19r1e.pdf**

Volkswagen AG. (2021, January). Transformers. Retrieved from Volkswagen Aktiengesellschaft: **https://www.volkswagenag.com/en/news/fleet-customer/2021/01/transformers.html**