Development of Vehicle Control Software via Software Sharing Approach

Matheus Gomes Camargo Watermann
Jaqueline Bierende
Fernando de Souza Gonçalves Cassiano
Márcio José Moraes
Robert Bosch LTDA

ABSTRACT

Collaborative development of embedded software has gained relevance in the automotive industry, driven by approaches such as Software Sharing, which enable greater flexibility and efficiency in the process. Within this context, this work presents an application software for electric powertrain management, implemented in a vehicle control unit (VCU) as part of a know-how buildup project. The employed architecture allows Original Equipment Manufacturers (OEMs) and third parties to integrate their own functionalities onto a protected base, accessed via a documented Application Programming Interface (API). This approach promotes modularity, intellectual property protection, and process decentralization. The application includes functions such as pedal reading, driving mode selection, CAN communication implementation, and torque calculation, having been validated in dynamometer tests. The study highlights the benefits of this model for agile and flexible solutions in electric mobility.

INTRODUCTION

The transformation of mobility, driven by vehicle electrification, autonomous driving and connectivity, demands engineering approaches that enable modularity, reuse, and technical specialization. In this context, the development of automotive software aims to maximize synergy among different stakeholders — Original Equipment Manufacturer (OEMs), tier 1 suppliers, software developers, and system integrators.

Among the emerging strategies for automotive software, the concept of software sharing (SwSh) represents an effective response to the complexity of current E/E system. It is an approach that enables quicker development cycles and allows OEMs to customize final products by integrating their own software, building on shared models provided by suppliers [3],[4]. The way this sharing materializes can vary widely, depending on the responsibilities assigned to each partner and the stage of the project.

Tier 1 suppliers were key system integrators in previous automotive architectures, offering complete engine Electronic Control Units (ECUs) that included hardware components, necessary sensors and actuators, and fully functional embedded software [5]. Those “traditional business models ensure product-oriented workflow, clear supply chain and simple responsibility sharing for series product [6]”. However, vehicle manufacturers are now seeking to reduce costs by reusing software across multiple control units [5].

The goal of this project is to develop control application software in C language to be implemented in the Vehicle Control Unit (VCU) using the base software provided by Bosch. Our specific objectives include developing torque path functions to coordinate torque requests to the inverter(s) and establishing CAN communication to configure and operate the Inverter/Converter. By achieving these objectives, this article explores the benefits of the SwSh approach based on a practical application for an electric powertrain control system, highlighting the improvements in reliability, performance, and scalability of the developed solution.

SOFTWARE ARCHITECTURE

The ECU/VCU software architecture is structured into three layers that operate on the underlying hardware, as illustrated in Figure 1. This model organizes the system into distinct levels, enabling the independent development of hardware, basic software (BSW), and application software (ASW). The modular segmentation not only facilitates updates and maintenance without the need for extensive system restructuring but also allows teams to work simultaneously on each layer, minimizing interference, ensuring application portability across different hardware platforms with minimal adjustments.

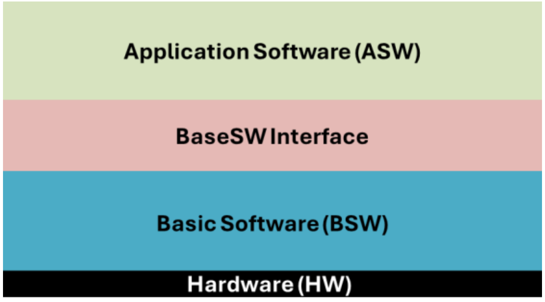

Figure 1 - Software architecture.
Adapted from [9].

ASW - The Application Software layer focuses on implementing higher-level functionalities tailored to the vehicle's specific functions and requirements [9]. It includes features such as powertrain coordination, torque management, thermal control, and driver assistance systems. This layer is highly specialized, reflecting the unique characteristics and intended use of each vehicle model, thereby directly impacting user experience and vehicle performance.

BASESW INTERFACES - This layer provides standardized interfaces that enable the upper layers to access the Base Software (BSW) and facilitate interaction between the Application Software (ASW) and the BSW [8].

BASE-SOFTWARE - provides essential functions for hardware abstraction, communication protocols, and low-level device drivers that ensure the integration of internal peripherals [7]. It also accelerates the development process and promotes code reuse across different applications.

SOFTWARE SHARING

The term *software sharing* encompasses a diverse set of technical collaboration models. In essence, it refers to the practice of sharing parts of the software development process among different stakeholders within the automotive ecosystem [4].

Bosch has identified many levels framework for model sharing, ranging from the use of ECU/VCU virtualization environments to full joint development. This framework helps define responsibilities and protect intellectual property in collaborative projects [3]. The following are the SwSh models and their main characteristics, highlighting the distribution of responsibilities between the supplier and OEM for hardware, BSW, and ASW components.

MODEL 0. COMPLETE SOLUTION – The partner develops and delivers a "turn-key" solution, including hardware, base software, and final application, according to client requirements. This reduces development time and ensures a comprehensive solution.

MODEL 1, 2, 3 and 4. OEM DEVELOPS PART OF ASW – The supplier delivers the ECU with validated hardware, configured BSW, and part of the ASW. The key difference in these models is that the OEM develops specific ASW components, with each model defining distinct interfaces between the supplier and OEM regarding software component responsibilities

MODEL 5. HARDWARE AND BSW PROVIDED BY THE PARTNER – The supplier delivers the ECU with validated hardware and configured base software. The OEM is responsible only for the application logic, allowing them to focus on their core competencies.

MODEL 6 AND 7. PARTIAL SOFTWARE ON THIRD-PARTY HARDWARE – The supplier contributes part of the software (e.g., diagnostics or torque control), but the hardware is provided by another integrator.

MODEL 8. ONLY HARDWARE PROVIDED – The partner supplies the validated hardware (ECU), and the OEM fully develops the software, including drivers and the operating system. This model allows OEMs greater control over software functionalities.

In summary, the software sharing models organize the different forms of collaboration between suppliers and OEMs, defining responsibilities in software development and ensuring that each stakeholder contributes according to their technical competencies and capabilities.

CUSTOMER INTEFACE LAYER

One of the key deliverables from suppliers is the BaseSW interface, commonly referred to as the Customer Interface Layer (CIL). The ASW leverages the services provided by the BSW through the CIL [9]. The following sections detail the CIL functions employed in ASW development, along with their specific implementations.

CIL FOR SCHEDULING- The control unit software operates on a Real-Time Operating System (RTOS), which is part of the BSW layer. A key category of CIL functions comprises scheduling interfaces (or OS interfaces), which are provided to the customer to facilitate the scheduling and execution of software tasks. These interfaces enable precise timing control, ensuring that tasks are executed according to their defined priority and timing requirements, thereby supporting deterministic behavior in critical control applications.

CIL FOR CAN COMMUNICATION - In the project, the supplier is responsible for delivering the CAN driver and CAN-related infrastructure software modules as part of the BSW. Additionally, should provide the customer with the API (application interface) to access these base software services. An example of an overview of CIL functions for

CAN message handling is depicted in the flowchart at Figure 2, followed by Table 1 with CIL functions. The CAN configuration is carried out by the supplier based on the customer's specifications.

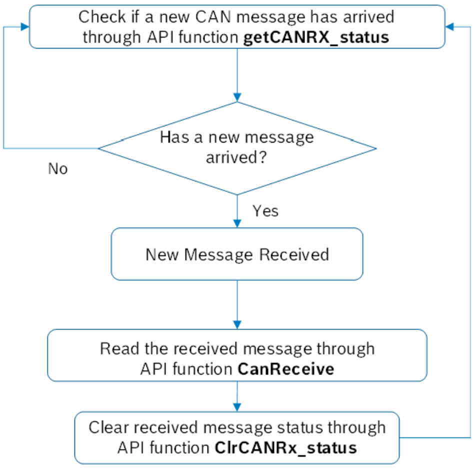

Figure 2 – CAN Message Reception Flow.
Source: Authors (2025)

CAN Interface function	Description
CanReceive	Interface function to read a CAN message
getCANRX_status	Interface function to read the status of Rx-message
ClrCANRX_status	Interface function to clear the status of Rx-message

Table 1 - CAN Interface Functions
Source: Authors (2025)

CIL FOR INPUT AND OUTPUT: The I/O interfaces enable customer ASW to access the ECU pins to read a sensor input or actuate an actuator like valves, switches and DC-motor. ECU specific analog, digital and PWM pins for the project and corresponding ECU pin enumeration are communicated with customer in an interface document. Examples of I/O interfaces are described in Table 2.

CAN Interface Function	Description
GetADC	Interface function for reading an analog ECU pin
GetDigIn	Interface function for reading a digital ECU pin
SetDigOut	Interface function to write into a digital ECU pin
SetPwmOut	Interface function to drive a PWM signal on an ECU pin

Table 2 - I/O Interface Functions
Source: Authors (2025)

SYSTEM OVERVIEW

The architecture of electric vehicle powertrains consists of various subsystems that work together to enable electric propulsion. These subsystems encompass functions such as energy storage and conversion, power control, and thermal management. Each subsystem has specific responsibilities that ensure the vehicle operates safely, reliably, and efficiently. Coordination among these elements, through a centralized control system, is essential for integrated operation and proper propulsion delivery [1],[2].

Figure 3 illustrates the topology of the electric propulsion system implemented in this work, consisting of power, control, and actuation units. The Rechargeable Energy Storage System (RESS) is responsible for storing the electrical energy used for vehicle propulsion and supplying it, at high voltage, to the inverter. The inverter, in turn, performs the controlled conversion of direct current to alternating current to power the electric motor.

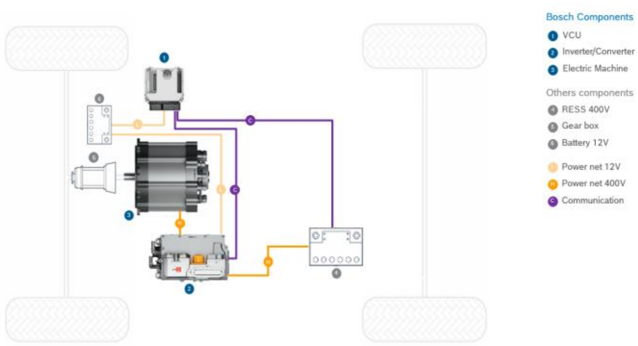

Figure 3 - System Topology.
Source: Adapted from [16]

The vehicle control unit serves as the management core of the system, communicating with the RESS, the inverter, and the display through the CAN network. Additionally, it is responsible for interpreting the driver's demands and sending the necessary control signals to the other powertrain components. The VCU has been specified as the central component of the E/E architecture, playing the role of integrating all elements of the electric powertrain.

INVERTER

The electric motor is powered by three-phase alternating current (AC), which is generated by converting direct current (DC) through an inverter. This conversion is achieved using a bridge circuit consisting of six Insulated-

Gate Bipolar Transistors (IGBTs), which function as high-speed, high-current controllable switches.

The inverter utilized in the project is a Bosch INVCON3.3 (Figure 4). It can operate in two modes: supplying AC current to the motor (inverter mode) or delivering DC current to the battery (rectifier mode).

The inverter, equipped with a high-level torque control function, receives the desired operating mode from the VCU via the CAN bus. Based on this communication, the inverter interprets the torque request as positive for acceleration or constant driving (inverter mode) or negative during braking (rectifier mode). Its internal software then adjusts the IGBT switching pattern to generate the required current profile accordingly.

Periodically, inverter sends messages to CAN bus in order to report its working state (actual torque, current, voltage, temperature, operating mode and diagnose messages) to the VCU. Therefore, there is a feedback for the demands requested by the VCU.

There is also a 2.7 kW DC/DC converter inside the inverter. This DC/DC reduces the voltage of the high voltage battery. Its output can be used to supply vehicle’s low voltage system and recharge low voltage battery (12 V).

Figure 4 - INVCON 3.3
Source: [14]

ELECTRIC MOTOR

The electric Motor is a three-phase alternating current electric machine, synchronous with permanent magnets. This kind of motor is recognized by its high torque capacity, with a high-power density and energetic efficiency.

The motor selected for the project is a Bosch SMG220 (Figure 5). It can deliver up to 350 Nm of torque, 140 kW of power, and reach a maximum speed of 15,500 rpm, while weighing only 70 kg (including the inverter).

Figure 5 - SMG 220
Source: [15]

VCU REQUIREMENTS ANALYSIS

From a functional perspective, a requirements analysis was conducted to establish the overall hardware specifications, including operational voltage and temperature range. Additionally, the basic software specifications were defined, focusing on the initialization and control of input and output (I/O) drivers, as well as the implementation of CAN communication protocols to ensure interoperability among subsystems.

Figure 6 illustrates the structure of input and output signals, along with the communication networks specified for the application. The configuration includes five analog inputs, nine digital inputs, ten digital outputs, and four CAN networks.

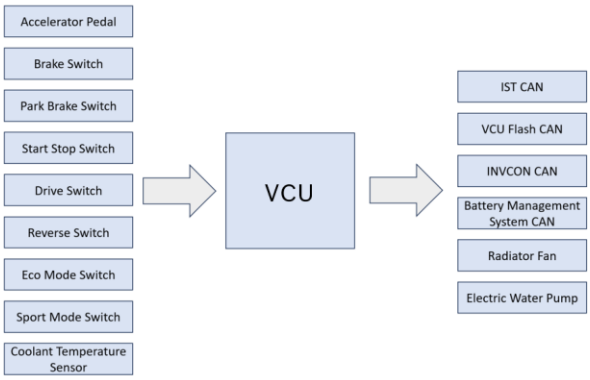

Figure 6 – VCU I/Os and CAN Networks
Source: Authors (2025)

The application software development was based on a hardware and software infrastructure provided by Bosch, aligned with Cooperation Model-5. CIL functions were utilized to access the hardware resources of the vehicle control unit.

ASW FOR TORQUE PATH FUNCTIONALITY

In electric powertrain management, the functions associated with the torque path coordinate the conversion of the driver's request into a torque demand value. This process involves analyzing the operating conditions of components such as the motor, transmission, and auxiliary systems, as well as interpreting the torque request based on inputs like accelerator, brake, and driving mode. Based on this information, the control functions determine the optimal torque to be delivered, ensuring a balance between performance, safety, and the integrity of the vehicle components.

Figure 7 illustrates the processing flow of torque requests in the VCU, from the acquisition of input signals to the generation of the torque demand signal. The process begins with reading signals from the accelerator pedal, brake pedal, transmission position (P, N, R, D), and electric machine speed (RPM). These signals, which can be analog, digital, or sourced from the CAN network, are processed by CIL functions that serve as access interfaces to the hardware resources and the basic software of the VCU.

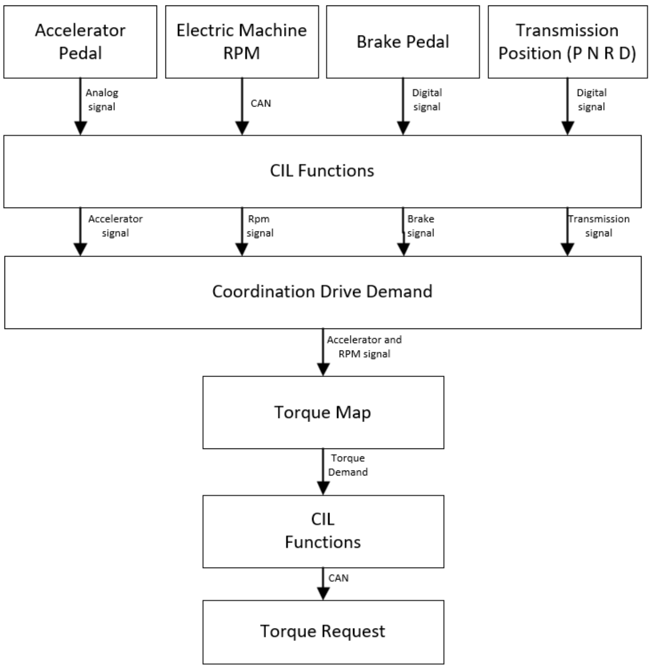

Figure 7 - Torque Path Flow
Source: Authors (2025)

Next, the signals are processed by the Coordination Drive Demand module, which verifies operating conditions to prevent inappropriate torque requests. This block ensures, for example, that no torque is applied when the vehicle is in neutral (N), safeguarding system integrity in situations where a torque request would be inappropriate. Subsequently, the Torque Map module defines the optimal torque value, considering the relationship between the accelerator position and the electric motor speed.

Finally, the determined torque signal is transmitted back to the CIL module, where it is formatted and sent via the CAN network as a Torque Request, directed to the inverter for electric motor control.

EV POWERTRAIN TEST IN DYNAMOETER

To validate the implemented torque management functions in the electric powertrain control architecture, a dynamometer test was conducted to monitor torque, power, and speed signals in real time, as shown in **Figure 8**.

The obtained torque curve reflects the expected vehicle behavior, as the calibration parameters of the torque map function were adjusted according to the operating limits of the integrated components. Consequently, the initial torque (when the vehicle is stationary and the accelerator is fully pressed) is set at 110 Nm. From this point, the torque is gradually reduced for higher RPM values based on the calibration strategy defined by the calibrator.

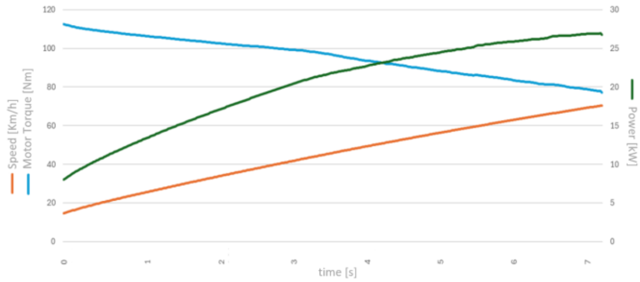

Figure 8 - Power, Torque and Speed Curve
Source: Authors (2025)

The maximum speed achieved in this application was approximately 75 km/h, with a peak power output of around 26 kW. This limitation was due to the maximum power capacity of the power source.

CONCLUSION

The study successfully demonstrates the feasibility of implementing torque demand management functions within the electric powertrain control architecture. The validation conducted on a dynamometer confirmed that the torque path processing functions accurately interpret driver inputs, converting them into torque requests that align with the operating limits of the electric machine and its components.

Furthermore, the concept of SwSh represents a substantial advancement in VCU software development, fostering faster, more flexible, and collaborative processes between OEMs and suppliers. By utilizing standardized frameworks and clearly defined contractual agreements, SwSh effectively addresses the limitations of conventional software development practices, promoting more efficient customer collaboration.

REFERENCES

[1] ANSYS, *What is an EV Powertrain?* Retrieved from https://www.ansys.com/simulation-topics/what-is-an-ev-powertrain.

[2] PEREIRA, Thiago M.; BOLSON, Marco A.; JUNGE, Thomas F.; PRESTES, Wagner de O. *High performance electric kart*. Division Bosch Engineering, Robert Bosch Ltda. Copyright © 2016. SAE International.

[3] Stéphane Louvet, Ulf Niebling, Mouham Tanimou. Model Sharing to leverage customer cooperation in the ECU software development. 7th European Congress on Embedded Real Time Software and Systems (ERTS 2014), Feb 2014, TOULOUSE, France.

[4] Stéphane Louvet, Mouham Tanimou. MDX and AUTOSAR Standards for Model Sharing to leverage Tier1 - OEM cooperation in the ECU software development. 8th European Congress on Embedded Real Time Software and Systems (ERTS 2016), Jan 2016, TOULOUSE, France.

[5] B. Boss (2012). Architectural aspects of software sharing and standardization: AUTOSAR for automotive domain. 2012 Second International Workshop on Software Engineering for Embedded Systems (SEES).

[6] Grote, W., Business Models for Software in the Automotive Industry. How to Retain Innovation? Software im Automobil, Talk, 2010-05-04.

[7] AUTOSAR, Layered Software Architecture, Release 19-11, AUTOSAR Core Partners, Nov. 2019. [Online]. Available: https://www.autosar.org/fileadmin/standards/R19-11/CP/AUTOSAR_EXP_LayeredSoftwareArchitecture.pdf.

[8] Zhang M, Teng Y, Kong H, et al. Automatic modelling and verification of AUTOSAR architectures. J Syst Softw. 2023; 201:111675.

[9] Vimal Sivashanmugam," Conversion of Control Unit Software towards AUTOSAR Compliance," Master's thesis in Information and Computer Engineering, Graz University of Technology, 2019

[10] the Automotive Industry. How to Retain Innovation? Software im Automobil, Talk, 2010-05-04.

[11] AUTOSAR, Layered Software Architecture, Release 19-11, AUTOSAR Core Partners, Nov. 2019. [Online]. Available: https://www.autosar.org/fileadmin/standards/R19-11/CP/AUTOSAR_EXP_LayeredSoftwareArchitecture.pdf.

[12] Zhang M, Teng Y, Kong H, et al. Automatic modelling and verification of AUTOSAR architectures. J Syst Softw. 2023; 201:111675.

[13] Vimal Sivashanmugam," Conversion of Control Unit Software towards AUTOSAR Compliance," Master's thesis in Information and Computer Engineering, Graz University of Technology, 2019.

[14] Bosch Mobility Solutions. Power Electronics INVCON3.3. Available at: https://www.bosch-mobility.com/en/solutions/power-electronics/power-electronics-invcon3-3-ohw/.

[15] Bosch Mobility Solutions. Separate Motor Generator SMG220. Available at: https://www.bosch-mobility.com/en/solutions/electric-motors/separate-motor-generator-ohw/.

[16] Bosch Mobility Solutions. High Voltage Hybrid Systems. Available at: https://www.bosch-mobility.com/en/solutions/powertrain/hybrid/high-voltage-hybrid-systems/.