
Politecnico di Torino

Master of Science in Computer Engineering

Tesi di Laurea Magistrale

# Rethinking Automotive Software Development: Exploring Software Defined Vehicle and its potential

**Supervisors**
prof. Danilo Bazzanella
dott.sa Piera Limonet

**Candidate**
Lorenzo Sciara

Anno Accademico 2023-2024

# Summary

This thesis project delves into the analysis of contemporary connected vehicle platforms, focusing on the benefits and challenges associated with these advanced solutions and emphasising aspects of safety and flexibility. A key trend in the current automotive sector is the prospect of transforming the car from a hardware-focused product to a software-driven device. The technology of choice for leading software development and production companies driving this change is the Software Defined Vehicle (SDV).

The primary objective of the thesis is to apply this paradigm to the development of a simulator for a vehicle control unit responsible for collecting telemetric data from the vehicle. The implementation of the simulator involves an in-depth analysis of the drawbacks of the automotive software production industry and the advantages of the Software Defined Vehicle solution. The simulator implementation also includes the creation of a scaled-down version of a connected vehicle platform, storage infrastructure and example application.

Using the Amazon Web Services (AWS), an environment in the cloud is established for the development of the necessary software for the operation of the vehicle control unit. Development of the vehicle control unit simulator is carried out, including client connectivity to interact with the cloud platform, telemetry generation, logic for remote operations, and optional applications. The final phase involves testing the simulator on compatible hardware to validate its functionality and performance.

The successful completion of this project in collaboration with Storm Reply, not only highlights the potential of the software-defined vehicle paradigm as a leading force in the future of the automotive sector, but also explores the economic, safety and security benefits associated with its adoption, paving the way for significant progress in the field and ensuring an advanced and safe end-user experience.

# Acknowledgements

Acknowledgement (optional)

# Contents

**List of Figures** 7

**List of Tables** 8

**Listings** 9

**1 Introduction** 11

1.1 Context . . . 11

1.2 Company . . . 12

1.3 Thesis Goal . . . 14

**2 State-of-the-Art Analysis** 16

2.1 Current Automotive Software Development . . . 16

2.1.1 difficulties . . . 17

2.2 Introduction to Software Defined Vehicle . . . 18

2.2.1 Enablers . . . 19

2.2.2 Benefits . . . 21

2.2.3 initiatives: SOAFEE . . . 23

**3 Proof Of Concept** 26

3.1 Amazon Web Services . . . 26

3.1.1 Introduction . . . 26

3.1.2 Used services . . . 26

3.2 Design . . . 26

3.2.1 Architecture . . . 26

3.2.2 Security . . . 26

3.3 Implementation . . . 26

3.3.1 Code . . . 26

3.3.2 Tools . . . 26

3.4 Test and Validation . . . 26

3.4.1 RPi demo . . . 26

**4 Concldng Remarks** 27
4.1 Contribution Recaps . . . . . 27
4.1.1 Have we meet the PoC goals? . . . . . 27
4.2 Future Works . . . . . 27
4.2.1 Transform the poc in a product . . . . . 27
4.2.2 Virtual workbenches . . . . . 27
4.2.3 Manage additional Use Cases (ML, Cockpit Apps, remote ECU etc..) . . . . . 27

**5 Conclusions** 28

**Bibliography** 29

# List of Figures


1.1 World automobile production in million vehicles [1] . . . . . 11

1.2 An incomplete overview of computers in a modern car [2] . . . . . 12

1.3 Logo of the partenr company of the project . . . . . 13

1.4 Here are a series of market research reports published by IT consulting firm Gartner that rely on proprietary qualitative data analysis methods to demonstrate market trends, such as direction, maturity and participants. [3] . . . . . 13

2.1 Cost of fixing errors increases in later phases of the life cycle [4] . . 17

2.2 A simple representation of communication using the MQTT protocol 21

2.3 Risks and time relationship in the various phases of a vulnerability lifecycle . . . . . 22

2.4 today development, integration, and validation workflows for embedded systems . . . . . 24

2.5 future development, integration, and validation workflows for embedded systems . . . . . 24

2.6 SOAFEE Architecture v1.0 [5] . . . . . 25

# List of Tables

# Listings

# Chapter 1

# Introduction

## 1.1 Context

The automotive industry stands out as one of the fastest-growing sectors, playing a significant role as both an employer and an investor in research and development; at the same time, it represents one of the most crucial domains for the European Union's economy. As reported in the article [1], in 2015, 21 million motor vehicles of all types were produced in Europe, representing a 23% share in the global production of more than 90 million units.

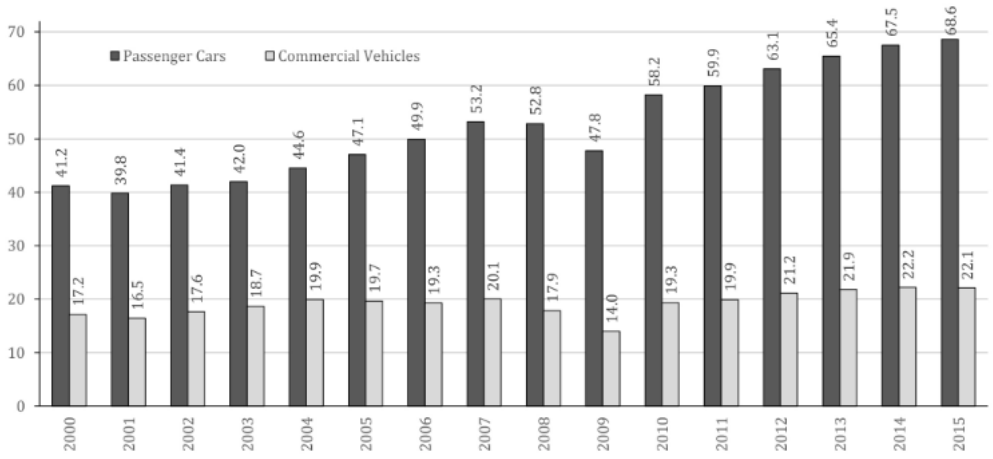


Figure 1.1. World automobile production in million vehicles [1]

In the evolving landscape of automotive technology, the imperative for automotive companies extends beyond the traditional realms of mechanical engineering to encompass a crucial reliance on both software and hardware components for vehicle construction. A glimpse into the intricate web of modern cars, as illustrated in Figure 1.2, reveals a mosaic of hundreds of distinct processors interfacing at various levels, earning contemporary vehicles the moniker of "Computers on wheels."

However, the proliferation of processors within vehicles, orchestrating communication to manage diverse components, presents a formidable challenge; each component often integrates a processor with unique logics, diverging from the logics

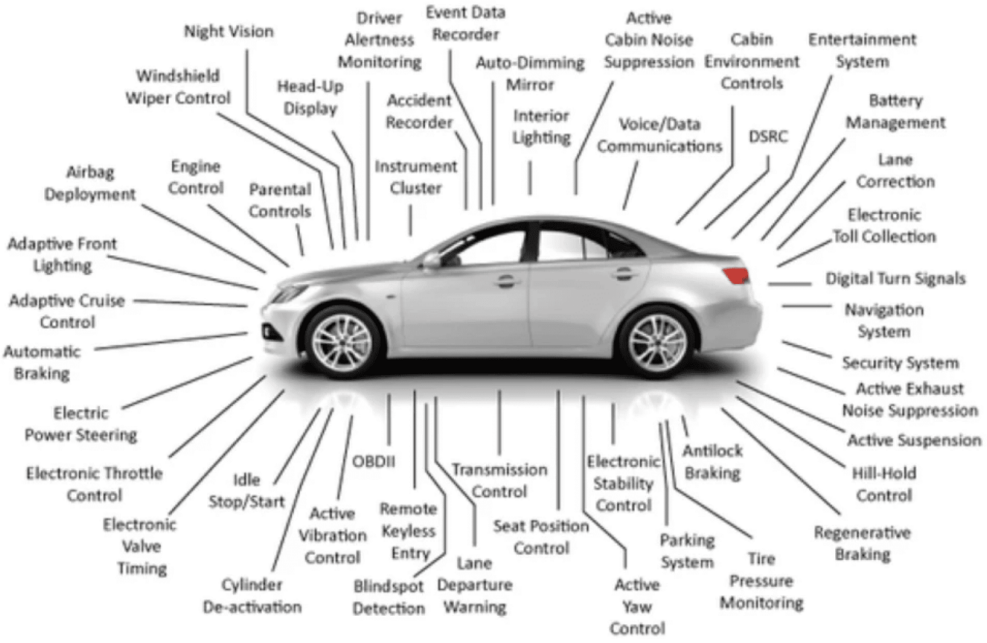


Figure 1.2. An incomplete overview of computers in a modern car [2]

embedded in processors of other components. Complicating matters further, these components are frequently supplied by companies with proprietary management logics, not readily accessible to the automotive companies themselves.

In addressing this intricate scenario, the transformative concept of a Software Defined Vehicle (SDV) comes to the forefront. Defined as "any vehicle that manages its operations, adds functionality, and enables new features primarily or entirely through software" [6], the notion of SDV offers a comprehensive solution to the challenges posed by the intricate interplay of software and hardware in modern vehicles.

Effectively navigating the development of SDV technology necessitates a collaborative approach across diverse companies, particularly in the realms of hardware and cloud computing. This collaborative synergy is exemplified in the realization of our project, made possible through the partnership with Storm Reply.

## 1.2 Company

Leveraging extensive experience in the cloud industry and fostering deep-rooted relationships within the automotive sector, Storm Reply stands out as the ideal choice to lead the project discussed in this thesis. A key player in the Reply group, Storm Reply specializes in designing and implementing innovative Cloud-based solutions and services [7].

With a diverse clientele spanning various sectors, notably the automotive industry, the company's expertise played a pivotal role in comprehensively understanding the project's context and internal dynamics. This profound knowledge served as the cornerstone for developing a tangible exemplification of the infrastructure.

Figure 1.3. Logo of the partenr company of the project

A point of pride for Storm Reply is its recognition as an Amazon Web Services (AWS) Premier Consulting Partner since 2014, ranking among the top Amazon Partners globally. This distinctive characteristic underscores the decision to develop the infrastructure using Amazon Web Services.

According to the official AWS description page [8] the AWS Cloud spans 102 Availability Zones within 32 geographic Regions around the world and servs 245 countries and territories. With millions of active customers and tens of thousands of partners globally, AWS has the largest and most dynamic ecosystem. AWS is evaluated as a Leader in the 2022 Gartner Magic Quadrant for Cloud Infrastructure and Platform Services, placed highest in Ability to Execute axis of measurement among the top 8 vendors named in the report.


Figure 1.4. Here are a series of market research reports published by IT consulting firm Gartner that rely on proprietary qualitative data analysis methods to demonstrate market trends, such as direction, maturity and participants. [3]

The infrastructure exhibits several key attributes contributing to its robustness and efficiency:

- **Security:** The infrastructure undergoes 24/7 monitoring to ensure the confidentiality, integrity, and availability of data. All data flowing across the AWS global network is automatically encrypted at the physical layer before leaving secured facilities.
- **Availability:** To ensure high availability and isolate potential issues, applications can be partitioned across multiple AZs (Availability Zones) within the same region, creating fully isolated infrastructure partitions.
- **Performance:** AWS Regions offer low latency, low packet loss, and high overall network quality. This is achieved through a fully redundant 100 GbE fiber network backbone, often providing terabits of capacity between Regions.

- **Scalability:** The AWS Global Infrastructure allows companies to take advantage of the virtually infinite scalability of the cloud. This enables customers to provision resources based on actual needs, with the ability to instantly scale up or down according to business requirements.
- **Flexibility:** The AWS Global Infrastructure provides flexibility in choosing where and how workloads are run, whether globally, with single-digit millisecond latencies, or on-premises.
- **Global Footprint:** AWS boasts the largest global infrastructure footprint, continually expanding at a significant rate.

## 1.3 Thesis Goal

In the automotive context, the use of Software Defined Vehicle (SDV) plays a crucial role in terms of costs, innovation, and safety. The goal of the thesis intertwine with the opportunities provided by Software Defined Vehicle technology, addressing the primary challenge of managing the current difficulties associated with the presence of various specialized hardware platforms on the same vehicle.

The central objective of this thesis is to propose a Software Defined Vehicle solution capable of eliminating various phases of the software production pipeline. This would result in significant time and cost savings, enabling the investment of these resources in other sectors. Since, by definition, a Software Defined Vehicle is characterized by the ability to undergo software updates dynamically and flexibly, this solution offers significant security advantages in various aspects:

1. Human Safety Critical Security: From the moment that a vehicle can be classified as safety critical (as it is reported in the standard ISO 26262-1:2018 of the ISO society where is said that "safety is one of the key issues in the development of road vehicles" [9]), the elimination of software vulnerabilities related to the vehicle's systems is crucial for the overall safety of the vehicle itself.
2. Intrinsic Software Security: This approach allows for the prevention and resolution of vulnerabilities unknown at the time of software design, contributing to ensuring a high standard of security.

Consequently, the use of Software Defined Vehicle aims to completely separate software and hardware, allowing the production of high-level software on entirely generalized hardware systems. This results in significant savings in terms of time and money for hardware production, along with providing an advantage in terms of security due to the simplification of software.

For example, as demonstrated by NIST in the research on the Analysis Of The Impact Of Software Complexity [10], the increase in software complexity in different cases results in less analyzable programs. In some instances, the same vulnerability analysis tool may detect vulnerabilities, while in others, analyzing the same code, it may not.

From a practical standpoint, the project's goal is to provide, through the use of AWS services, a cloud infrastructure capable of managing the Software Defined Vehicle both in terms of software production and data analysis.

# Chapter 2

# State-of-the-Art Analysis

The following chapter constitutes an in-depth exploration of current technologies and methodologies within the automotive industry, with a specific focus on the complexity of vehicular software development. Firstly, the current automotive landscape will be examined, providing a detailed insight into challenges associated with software development in vehicles.

Subsequently, through meticulous analysis of scientific publications, technical reports, and practical implementations, the chapter delves into the radical transformation of the automotive sector facilitated by the concept of Software Defined Vehicle (SDV). This technology, crucial for technological progress and vehicular safety, will be explored from various perspectives. Particularly, the synergy between Cloud, software, and hardware will be investigated, highlighting solutions proposed by major industry players and analyzing their applications, benefits, and limitations.

The objective is to offer a comprehensive overview of current dynamics, emphasizing the pivotal role of SDV in the evolution of the automotive industry.

## 2.1 Current Automotive Software Development

In the past, the automotive industry advanced primarily through the development of technologies in mechanical engineering, focusing on perfecting combustion engines. Nowadays, the paradigm has radically changed due to multiple factors, including electrification, automation, shared mobility, and connected mobility.

Software technology development in the automotive field can be metaphorically compared to what has happened in smartphone development, as highlighted in the manifesto document regarding Bosch's Software Defined Vehicle (SDV) [11].

The ultimate goal is to achieve simple and user-friendly devices that fully meet the user's needs. Currently, many customers express dissatisfaction because their cars do not offer the same functionality and ease of use common in smartphones. Many ask: Why can't my $50,000 car perform the same tasks as my $300 smartphone?

A key difference between the automotive and smartphone industries is the level of complexity, which brings with it a number of issues.

### 2.1.1 difficulties

We can analyse in depth the problems of the current automotive software that is being developed via 4 main difficulties:

- **Specialized Hardware:** Today's vehicles are still complex systems of systems. Each subsystem in a car, from brakes to transmission, is a complex entity, supplied by a different manufacturer and integrated with a unique software architecture. The level of complexity and the need for seamless interoperability between systems far exceeds that of today's smartphones.
- **Time:** The software production pipeline involves many development and testing steps with a not inconsiderable amount of time spent on each one. This is greatly increased by the presence of different components, so development time must be considered for each different unit of the system.
- **Cost:** The complexity of the software systems in vehicles entails very high costs, aggravated by the fact that the test phase is often carried out directly on the boards (for hardware requirements), which means a much longer production process, especially in the event of errors.

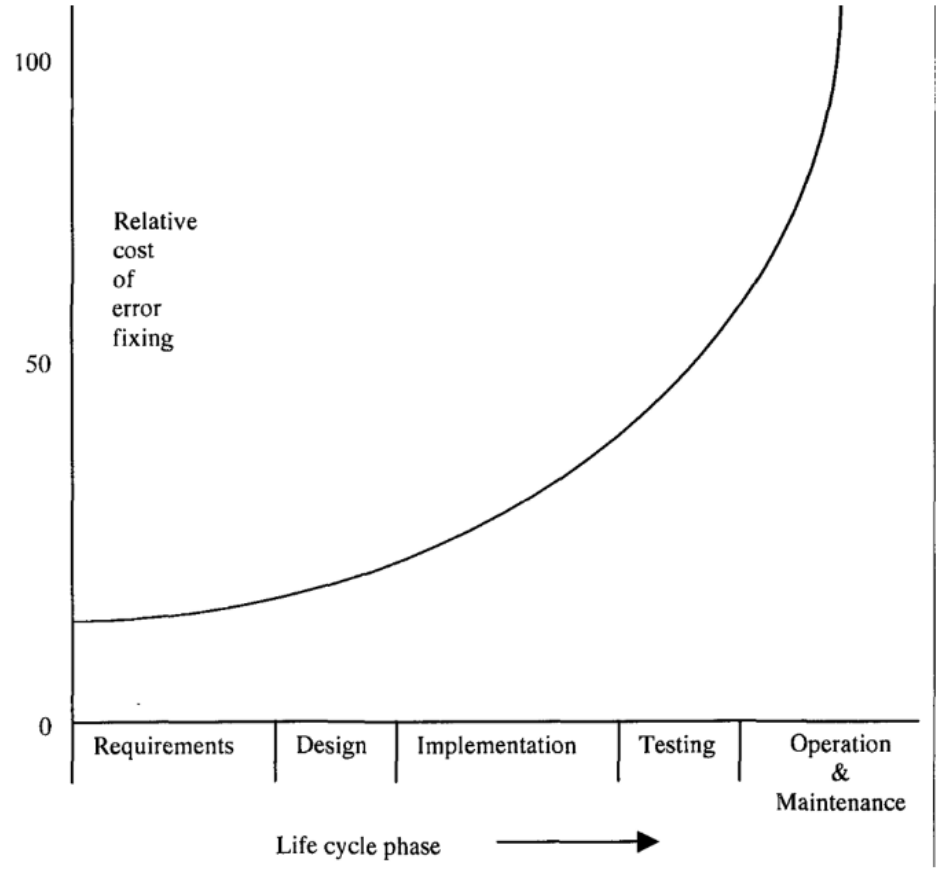


Figure 2.1. Cost of fixing errors increases in later phases of the life cycle [4]

- **Human Safety Security:** Automotive embedded software must meet stringent reliability and security requirements, while delivering performance and a reasonable memory footprint. To develop automotive embedded software, you need the right tools that meet safety and security standards to evaluate, prototype and test your software.

What lessons can be drawn from the study of barriers that can be applied to the vehicle lifecycle? Historically, the vehicle lifecycle has been characterised by the simultaneous production and deployment of tightly integrated hardware and software. Once the vehicle was in the hands of the consumer, its characteristics remained largely unchanged until the end of its life. However, the SDV paradigm introduces the possibility of decoupling hardware and software release dates, a prerequisite for adopting a digital-first approach. This approach brings the design and virtual validation of the digital vehicle experience to the forefront of the lifecycle. It also requires the application of the digital-first concept, which means that new ideas for the vehicle experience are first explored in virtual environments to ensure early user feedback, long before any custom hardware needs to be developed or a physical test vehicle is available. Digital first is the application of design thinking and lean startup principles, originally rooted in internet culture, to the tangible realm of automotive development.

## 2.2 Introduction to Software Defined Vehicle

The Software Defined Vehicle represents the new frontier of automotive manufacturing and is poised to completely change the paradigm of automotive production.

If we imagine bringing a feature update to one of today's vehicles, it will most likely take anywhere from one to seven years from the idea to when that feature is actually perceptible in the production vehicle; this takes so long because the vehicles produced up to this point have not been designed with frequent updates in mind [12]. Traditionally focused on physical functionality, the automotive industry has evolved from early electronic features such as airbags, vehicle stabilisation and braking systems to modern driver assistance and even automated driving. The current shift towards a digital experience is possible thanks to vehicle design that includes software integration as a fundamental part. Software should no longer be seen as an accessory to the vehicle, but as an integral part of the vehicle itself.

The simultaneous efforts of major automotive companies such as Bosch, Renault and Stellantis, in collaboration with leading computer developers such as Arm, BlackBerry and AWS, have given rise to the Software Defined Vehicle concept, which they define as "any vehicle that manages its own operations, adds functionality and enables new features primarily or entirely through software" [6].

The Software Defined Vehicle solution is nowadays being considered by several companies as the manifesto of a new era of vehicle development. An example is given by the Renault Group, which in an overview of its products describes: "Today, it is already possible to make remote updates of some vehicles via the Firmware Over The Air (FOTA) system. This keeps the vehicle safe by making it

easier and faster to improve the on-board system and apply patches. Tomorrow, the Software Defined Vehicle's flexible and scalable architecture will enable the faster development and integration of new features throughout the vehicle lifecycle, directly into the cloud, that is, in secure online servers accessible from anywhere and anytime" [13].

It is evident that Software Defined Vehicles represent the future of the automotive industry, promising an enriched and sustainable user experience as vehicle technologies evolve. This section further clarifies the current state of the industry, highlighting the key enablers that are allowing the development of the SDV paradigm and the benefits of this innovation.

### 2.2.1 Enablers

There are mainly three fundamental technologies that contribute to the realisation of the Software Defined Vehicle: standardized hardware, cloud and over-the-air (OTA) updates via OTA servers, all developed by leading companies in the computing industry. In this section, each technology will be analysed with reference to concrete examples from the current market.

- **Standardized hardware**

  One of the most important aspects of Software Defined Vehicle is the separation of software from hardware. To achieve this, it is essential to move away from the approach of using dedicated hardware for each vehicle component system, and instead favour an approach based on general purpose processors that are as centralised as possible. This transition not only promotes ease of software development and scalability, but also offers the opportunity to create parity between the virtual development and test environment and the real execution environment.

  Several players in the semiconductor industry have stepped up to the challenge of realising this vision, including Arm. Through the development of energy-efficient processors, Arm is present in every part of the vehicle, from high-performance systems in advanced driver assistance systems (ADAS), automated driving (AD), in-vehicle infotainment (IVI) and digital cockpits, to gateway, body and microcontroller endpoints [14]. The aim is to create Arm-based MCUs that enable implementation of a common architecture, scalability between applications to meet processing requirements, software reuse and reduced development costs.

  Another major player is Qualcomm, which is being adopted by the Renault Group through its Snapdragon Digital Chassis vehicle architecture, a set of cloud-connected platforms for telematics and connectivity, digital cockpits, assistance and driver autonomy.

- **Cloud**

  Using a cloud platform that offers scalable and secure solutions for real-time application updates, increased connectivity and efficient data management is essential for SDV.

Well-known companies such as Amazon Web Services (AWS) and Google Cloud are already present in the automotive industry as partners of partner of many automotive companies. The AWS services and technologies will be in depth described in the futher chapters.

- **Over-The-Air updates**

  An Over-The-Air (OTA) update is the remote and wireless transfer of applications, services, firmware and configurations from a server to a target device. This process takes place over an available network, preferably the Internet. The main purposes of OTA are to remotely update software or firmware, provide power-safe procedures to ensure that the device will boot even if power is lost during the update process, maintain a robust implementation, ensure data protection and reduce overall maintenance costs [15].

  In the context of the thesis, it is crucial to acknowledge that the implementation of OTA updates may increase the vulnerability of automotive systems to hacking and other cyber attacks. These vulnerabilities could potentially be exploited by hackers to gain unauthorised access to private information, take remote control of the vehicle or even cause it to malfunction. Another significant issue is the leakage of information about updates and their sources. This can enable malicious actors to introduce viruses and malware, further exacerbating the security risks associated with OTA updates [16].

  To perform an OTA update, both a client on the vehicle, responsible for waiting and checking for incoming updates, and a server, facilitating the availability of the update broadcast to all connected devices, are essential. In this context, Autosar can be considered, as it represents a standard and open source architecture for intelligent mobility [17], which includes a dedicated platform for client and server management of OTA updates. Another notable example is Hawkbit, which serves as a backend framework for deploying software updates to edge devices and is being developed by the Eclipse Foundation; this tool will be discussed in more detail in later chapters as it will be used to create a proof of concept. The final tool of note is AWS Greengrass, an edge agent manager for managing software updates in edge IoT devices, provided by AWS; this tool will also be discussed in later chapters as an alternative solution to the client manager.

- **MQTT communication**

  The Message Queuing Telemetry Transport (MQTT) is a standardized protocol, specified by ISO/IEC 20922:2016 and developed by the Oaesis organization. It enables the exchange of Application Messages over a network connection, providing an ordered, lossless stream of bytes from the Client to Server and Server to Client without the need to support of a specific transport protocol.

  In an MQTT transport, an Application Message carries payload data, a Quality of Service (QoS), a collection of Properties, and a Topic Name. Clients, which can be programs or devices, perform various actions such as opening and closing network connections, publishing Application Messages, subscribing to requested Application Messages, and managing subscriptions [18].

On the Server side, it acts as an intermediary between publishing and subscribing Clients. The Server accepts network connections, processes Subscribe and Unsubscribe requests, and forwards Application Messages matching Client Subscriptions. The Server, also known as the Broker, essentially coordinates messages among various Clients. Its responsibilities extend to authorizing and authenticating MQTT Clients, transmitting messages to other systems for further analysis, and managing tasks such as handling missed messages and Client sessions [19].

Sessions, representing stateful interactions between Clients and Servers, can last for the duration of a Network Connection or span multiple consecutive connections.

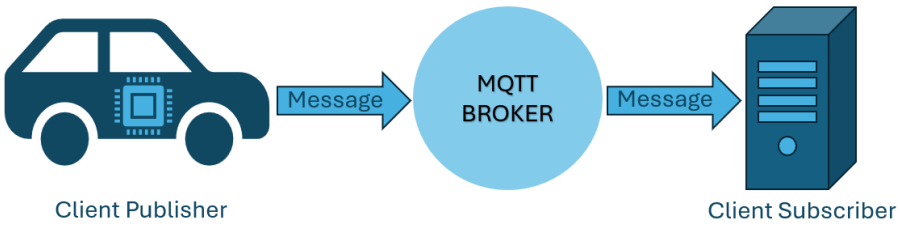


Figure 2.2. A simple representation of communication using the MQTT protocol

The MQTT protocol can be used in SDV, both for sending data produced by the vehicle to the cloud servers and for sending updates from the servers to the vehicle. This is because the MQTT protocol allows asynchronous and misaligned communication even in the presence of poor connectivity, a situation that cannot be underestimated in the automotive field.

The collaborative efforts of this technologies contribute to advancement of SDV for makeing vehicles not only defined by their physical attributes but also as dynamic entities that can be continuously updated through software.

### 2.2.2 Benefits

The Software Defined Vehicle, as introduced in the previous chapters, brings several benefits to both automotive companies and the end-user experience. These innovations are made possible by the fact that the vehicle becomes a device that can be constantly monitored and updated in real time via the cloud throughout its entire lifecycle. Let us now look at the key benefits.

From the point of view of this project, the main innovation brought by this technology is the security of the device software. Since, as mentioned above [9],

vehicles are considered as safety elements critical to human life, the safety benefits can be analysed from two perspectives:

- **Human Safety Critical Security:** The ability of SDV to receive real-time data from the vehicle allows in-depth monitoring of all its components. Taking the influence of tyres as an example, it has been found that most road accidents are caused by tyre wear and lack of regular maintenance. It is therefore necessary to assess the health of tyres through continuous monitoring of physical parameters such as tyre thickness, temperature and pressure, as well as regular maintenance. This helps to eliminate or minimise the possibility of tyre bursts and subsequent accidents. It also improves the safety of people and vehicles [20]. These factors can be monitored either manually or automatically: manual predictive maintenance requires human intervention and can lead to some errors; automatic predictive maintenance using artificial intelligence can be more efficient [21]. Renault defines this work as "predictive maintenance" [13], stressing the importance of collecting and analysing data in a centralised system to anticipate and prevent potential failures, ensure the safety of people, reduce maintenance costs and improve the performance of the vehicle.

- **Intrinsic Software Security:** In the presence of bugs and vulnerabilities in the vehicle's software, SDV makes it possible to intervene promptly to resolve each problem and reduce the window of exposure.

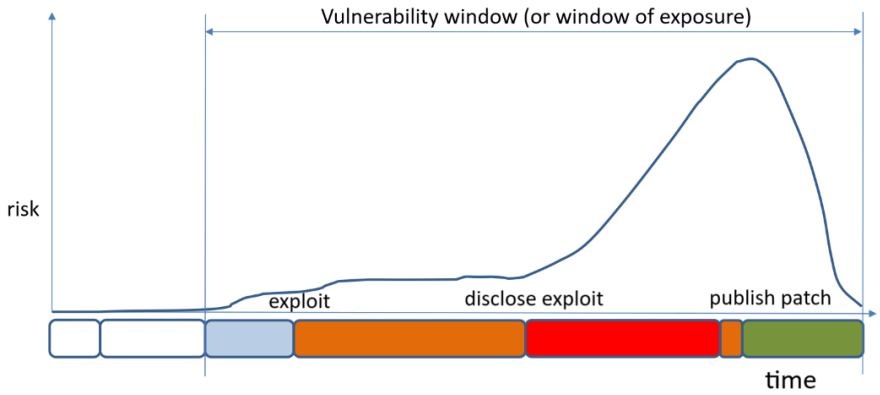


Figure 2.3. Risks and time relationship in the various phases of a vulnerability lifecycle

A crucial aspect of vehicle software security is the robustness of the algorithms, especially in the context of autonomous driving. In this context, a predictive algorithm responsible for vehicle safety decisions can be continuously improved and optimised. The SDV also introduces the concept of the 'digital twin', a platform that virtually replicates the functionality and behaviour of the vehicle. Thanks to this technology, predictive algorithms used in autonomous driving can be effectively tested on the cloud platform and, when ready, integrated directly into the vehicle.

From a user experience point of view, two other significant benefits can be identified: an increase in the value of the vehicle, which can be continuously upgraded over time, and the ability to enable additional vehicle functions via software. For example, the user can decide to activate a feature for a certain period of time and then deactivate it (paying only for the time it is used), or activate a new feature that was not available at the time of purchase. In essence, the vehicle becomes a dynamic platform that is constantly evolving and fully customisable through the software.

For automotive companies, the benefits mentioned so far can bring direct benefits to the industry. In support of this, Stellantis reports that: "the team in Poland will contribute to the global software creation network that is key to Stellantis' work in creating software-defined vehicles (SDVs) that offer customer-focused features throughout the vehicle's life span, including updates and features that will be added years after the vehicle is manufactured. "Creating an infrastructure inside our vehicles that easily and seamlessly adapts to meet driver expectations is a key element of Stellantis' global drive to deliver cutting edge mobility. Stellantis' software-driven strategy deploys next-generation tech platforms, building on existing connected vehicle capabilities to transform how customers interact with their vehicles and to generate €20 billion in incremental annual revenues by 2030".

In addition, the SDV paradigm brings an advantage from a software production pipeline perspective. In today's software production scenario, there can be two development mechanisms:

- A more traditional mode in which software is created directly on the system, hence on the processor itself. This is undoubtedly the most inconvenient solution, as it would require unnecessary overuse of processors, wasting resources, money and time.
- Alternatively, developers rely on cumbersome operating system emulation tools on the host machine and the cross-compilation process, which uses a dedicated compiler to produce executable code for the target system. Once the code is on the development system, a final integration and validation test can be performed, but scalability is limited to the number of physical hardware platforms.

Typical workflows for the development, integration and validation of embedded systems are as follows [22]:

As explained in the following chapters, by using the software defined vehicle, i.e. operating systems that rely on general purpose porpouse architectures to provide parity between cloud and edge systems, it is possible to reduce the embedded developer's workflow to remove many of the steps that are now no longer required, as shown in the diagram below [22]:

### 2.2.3 initiatives: SOAFEE

In 2021 Arm, AWS, and other founding members announced the Scalable Open Architecture for Embedded Edge (SOAFEE) Special Interest Group, which brings

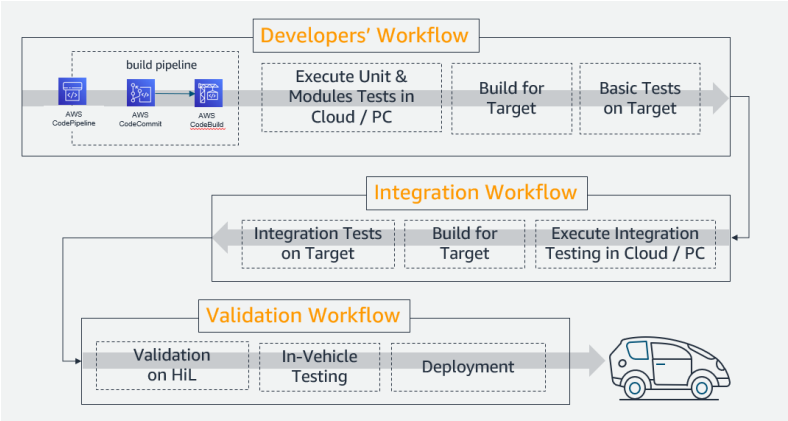


Figure 2.4. today development, integration, and validation workflows for embedded systems

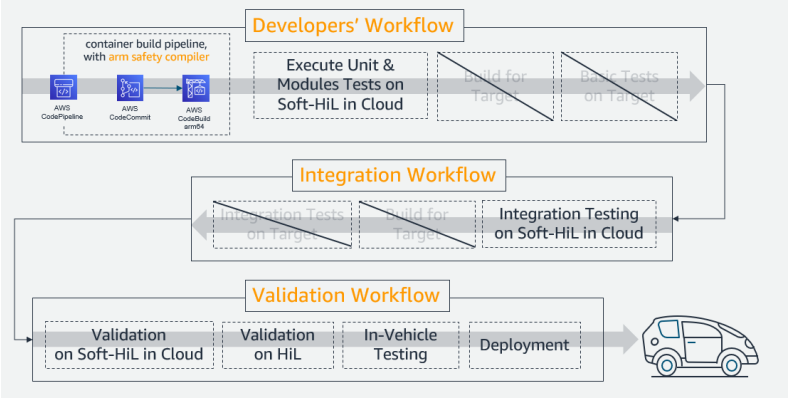


Figure 2.5. future development, integration, and validation workflows for embedded systems

together automakers, semiconductor, and cloud technology leaders to define a new open-standards based architecture to implement the lowest levels of a software-defined vehicle stack. [22]

SOAFE is created to achive Software Defined Vehicle, and for doing that four-pillar principle are used [23]:

1. Standards: standardization ensures interoperability and compatibility among various software components, fostering a cohesive ecosystem for Software Defined Vehicles.

2. New software architecture and methodologies: this involves transitioning from traditional monolithic architectures to more modular and scalable designs; the incorporation of agile development practices and DevOps methodologies ensures efficient and continuous software evolution.

3. Industry collaboration: Fostering partnerships, knowledge sharing and collaboration among key stakeholders, including automakers, technology companies and regulators, is essential.

4. Vehicle simulation: simulated environments allow in-depth testing and refinement of software functionality to ensure optimal performance and security under a variety of conditions.

SOAFEE aims to adopt and enhance current standards used in today's cloud-native world to help manage the software and hardware complexity of the automotive Software Defined Vehicle architecture.

The core principles of safety, security, and real time are inherent in each pillar. It is fully expected that the SOAFEE architecture will support use-cases that execute safety-critical services alongside non-safety-critical ones. It is fully expected that the SOAFEE architecture will support use cases that execute safety-critical services alongside non-safety-critical services. As it is not reasonable to develop the whole platform according to one safety standard, the strategy is to develop only safety-critical elements according to ISO 26262 and to isolate them from the non-safety-critical elements in order to ensure spatial, temporal and communication isolation. All implementations pass security checks and follow a set of best practices /citeSoafeeArchitectureOverview.

The SOAFEE paradigm is based on a very sophisticated architecture becouse it should work in the same way in the vehicle and in the cloud and follow cloud native technologies while considering the automotive specific needs for safety and limited resource footprints [5].

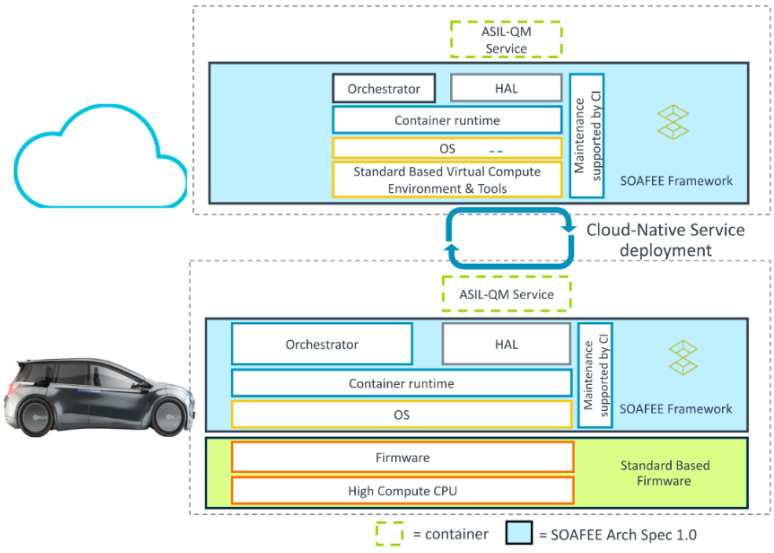


Figure 2.6. SOAFEE Architecture v1.0 [5]

# Chapter 3

# Proof Of Concept

## 3.1 Amazon Web Services

### 3.1.1 Introduction

### 3.1.2 Used services

## 3.2 Design

### 3.2.1 Architecture

### 3.2.2 Security

## 3.3 Implementation

### 3.3.1 Code

### 3.3.2 Tools

## 3.4 Test and Validation

### 3.4.1 RPi demo

# Chapter 4

# Conclding Remarks

## 4.1 Contribution Recaps

### 4.1.1 Have we meet the PoC goals?

## 4.2 Future Works

### 4.2.1 Transform the poc in a product

### 4.2.2 Virtual workbenches

### 4.2.3 Manage additional Use Cases (ML, Cockpit Apps, remote ECU etc..)

# Chapter 5

# Conclusions

# Bibliography


[1] Vošta and Kocourek, "Competitiveness of the european automobile industry in the global context." *Politics in Central Europe*, vol. 13, no. 1, pp. 69–89, 2017. [Online]. Available: https://www.politicsincentraleurope.eu/documents/file/PCE_2017_1_13.pdf#page=71

[2] R. Saracco, "Sdv: Software defined vehicles," 2021. [Online]. Available: https://cmte.ieee.org/futuredirections/2022/11/01/sdv-software-defined-vehicles/

[3] J. Scheibmeir, S. Sicular, A. Batchu, M. Fang, V. Baker, and F. O'Connor, "Magic quadrant for cloud ai developer services," *Gartner*, 2023. [Online]. Available: https://pages.awscloud.com/Gartner-Magic-Quadrant-for-Cloud-AI-Developer-Services.html?trk=d59e704f-4f30-4d43-8902-eb63c3692af4&sc_channel=el

[4] M. KARLSSON and L. SCHÖNBECK, "Department of technology management and economics," *CHALMERS UNIVERSITY OF TECHNOLOGY*, p. 11, 2018. [Online]. Available: https://odr.chalmers.se/server/api/core/bitstreams/077c3440-c033-418e-92ed-eda5dd638c5f/content

[5] S. project and M. Spencer, "Architecture," 2023. [Online]. Available: https://architecture.docs.soafee.io/en/latest/contents/architecture.html

[6] B. QNX, "What is a software-defined vehicle?" *Software-Defined Vehicles*, 2024. [Online]. Available: https://blackberry.qnx.com/en/ultimate-guides/software-defined-vehicle

[7] S. Reply, "Who we are," 2024. [Online]. Available: https://www.reply.com/storm-reply/en/

[8] AWS, "Aws global infrastructure," *About-aws*, 2024. [Online]. Available: https://aws.amazon.com/about-aws/global-infrastructure/

[9] I. Society, "Functional safety," *ISO 26262-1:2018 Road vehicles*, no. 2, 2018. [Online]. Available: https://www.iso.org/obp/ui/en/#iso:std:68383:en

[10] C. D. D. O. E. N. Fong and P. E. Black, "Impact of code complexity on software analysis," *NIST IR*, vol. 8165, no. upd1, pp. 1–12, 2017. [Online]. Available: https://nvlpubs.nist.gov/nistpubs/ir/2017/NIST.IR.8165.pdf

[11] D. Slama, A. Nonnenmacher, and T. Irawan, "The software-defined vehicle," *A Digital-First Approach to Creating Next-Generation Experiences*, pp. 1–6, 2023. [Online]. Available: https://www.bosch-mobility.com/media/global/mobility-topics/mobility-topics/software-defined-vehicle/

[12] A. Nonnenmacher and L. Product, "What is a software-defined vehicle in your opinion?" 2024. [Online]. Available: https://www.bosch-mobility.com/en/mobility-topics/software-defined-vehicle/?gad_source=1

[13] R. Group, "What is software defined vehicle?" 2024. [Online]. Available: https://www.renaultgroup.com/en/news-on-air/news/

all-about-software-defined-vehicle/
[14] Arm, "Do you have the right tools?" 2024. [Online]. Available: https://www.arm.com/developer-hub/embedded-systems/automotive-tools
[15] A. K. Srivastava, K. CS, D. Lilaramani, R. R, and K. Sree, "An open-source swupdate and hawkbit framework for ota updates of risc-v based resource constrained devices," *2nd International Conference on Communication, Computing and Industry 4.0*, p. 1, 2022. [Online]. Available: https://ieeexplore.ieee.org/document/9689433
[16] M. Helmy and M. Mahmoud, "Enhanced multi-level secure over-the-air update system using adaptive autosar," *International Conference on Computer and Applications*, p. 1, 2023. [Online]. Available: https://ieeexplore.ieee.org/document/10401797
[17] Autosar, "Vision," 2024. [Online]. Available: https://www.autosar.org/about
[18] A. Banks, E. Briggs, K. Borgendale, and R. Gupta, "Mqtt version 5.0," *OASIS Standard*, pp. 10 – 13, 2020. [Online]. Available: https://docs.oasis-open.org/mqtt/mqtt/v5.0/os/mqtt-v5.0-os.pdf
[19] AWS, "What are mqtt components?" 2024. [Online]. Available: https://aws.amazon.com/it/what-is/mqtt/
[20] B. J. R. G and R. P, "Iot based system to predict the defects of tires in heavy vehicle," *International Conference on Sustainable Communication Networks and Application (ICSCNA)*, p. 1, 2023. [Online]. Available: https://ieeexplore.ieee.org/document/10370342
[21] K. T. Selvi, N. Praveena, K. Pratheksha, S. Ragunanthan, and R. Thamilselvan, "Air pressure system failure prediction and classification in scania trucks using machine learning," *Second International Conference on Artificial Intelligence and Smart Energy (ICAIS)*, p. 1, 2022. [Online]. Available: https://ieeexplore.ieee.org/document/9742716
[22] L. H. M. da Ros Gomes and S. Goscik, "Building an automotive embedded linux image for edge and cloud using arm-based graviton instances, yocto project, and soafee," 2022. [Online]. Available: https://aws.amazon.com/blogs/industries/building-an-automotive-embedded-linux-image-for-edge-using-arm-graviton-yocto-project-
[23] S. project, "Achieving software-defined vehicles," 2023. [Online]. Available: https://www.soafee.io/