A Project Report on

AUTOSAR and Adaptive AUTOSAR

Submitted by

C Praveen Kumar Reddy(R170128)

Submitted to

Under the supervision of

T Sandeep Kumar Reddy

Assistant Professor in Computer Science and Engineering Department
RGUKT ,RK Valley

as a part of Major Project in E4-SEM2

Table of Contents

Acknowledgement

We would like to express our sincere gratitude to **T Sandeep kumar Reddy**, our project internal guide for valuable suggestions and keen interest throughout the progress of our course of research .

We are grateful to **Mr N Satyanandaram** HOD CSE , for providing excellent computing facilities and congenial atmosphere for progressing with our project .

With sincere regards,
C Praveen Kumar Reddy(R170128)

Certificate

This is to certify that the report entitled "AUTOSAR and Adaptive AUTOSAR" submitted by C Praveen Kumar Reddy(R170128) in partial fulfillment of the requirement for the award of Bachelor of Technology in Computer Science Engineering is a bonafide work carried out by her under supervision and guidance.

The report hasn't been submitted previously in part or in full to this or any other university or institution for the award of any degree. Under the Guidance of **T Sandeep Kumar Reddy** (Assistant Professor, Computer Science &Engineering, RGUKT, R.K Valley).

Project Guide,

Mr T Sandeep Kumar Reddy,
Assistant Professor
CSE,
RGUKT,RK Valley.

Head of Department,

Mr N Satyanandaram,
Assistant Professor,
CSE,
RGUKT,RK Valley.

Declaration

I C Praveen Kumar Reddy (R170128) here by declare that this report entitled “AUTOSAR and Adaptive AUTOSAR” submitted by me under the guidance and supervision of Mr T Sandeep Kumar Reddy, is a bonafide work. I also declare that it has not been submitted previously in part or in full to this university or other university or institution for the award of any degree or diploma.

Date :01-05-2023

Place : RK Valley

Name & ID NO

C Praveen Kumar Reddy(R170128)

Abstract

AUTOSAR and Adaptive AUTOSAR are used in the development of automotive electronic systems, particularly in the area of software development for electronic control units (ECUs).

The AUTOSAR standard provides a framework for the development of automotive software that is scalable, portable, and reusable. It enables the development of software components that can be integrated into different vehicle models and across different manufacturers. AUTOSAR is used to develop software for a wide range of automotive applications, including powertrain, chassis, body electronics, and infotainment systems.

Adaptive AUTOSAR, on the other hand, is specifically designed for use in advanced automotive systems such as autonomous driving, connected cars, and advanced driver assistance systems (ADAS). These systems require a more flexible and scalable software architecture, which Adaptive AUTOSAR provides. Adaptive AUTOSAR is used to develop software for high-performance ECUs that require real-time processing, such as sensor fusion, perception, decision-making, and control.

AUTOSAR and Adaptive AUTOSAR are used in the development of software for a wide range of automotive applications, from traditional ECUs to more advanced and complex systems that require high-performance computing capabilities.

Introduction

AUTOSAR (AUTomotive Open System ARchitecture) is a global standard that aims to create a common software architecture for automotive electronic control units (ECUs). It provides a set of specifications, guidelines, and templates to enable the development of automotive software components and systems. The goal of AUTOSAR is to improve the quality and efficiency of automotive software development, reduce development costs, and simplify the integration of software components from different suppliers.

AUTOSAR is designed to be scalable, portable, and reusable. It enables the development of software components that can be integrated into different vehicle models and across different manufacturers. AUTOSAR also provides a framework for managing software complexity, including tools for configuration, calibration, and testing.

Adaptive AUTOSAR is an extension of the AUTOSAR standard that is specifically designed for use in advanced automotive systems such as autonomous driving, connected cars, and advanced driver assistance systems (ADAS). Adaptive AUTOSAR provides a software architecture that is more flexible and scalable than the original AUTOSAR standard, allowing for the integration of new functions and features as they become available. Adaptive AUTOSAR provides a framework for integrating software components from different sources, enabling the development of more complex systems. It also supports over-the-air updates, which allows for the remote updating of software components in a vehicle.

Purpose

The purpose of this project report is to provide an overview of the benefits of using AUTOSAR and Adaptive AUTOSAR for automotive software development. This could include improved quality and efficiency, reduced development costs, and simplified integration of software components.

The primary goal of this is to describe the key features of AUTOSAR and Adaptive AUTOSAR, including their architecture, specifications, and guidelines. This could help readers understand how these standards work and how they can be applied in practice and to explain how AUTOSAR and Adaptive AUTOSAR can be used to develop software for different types of automotive applications, including traditional ECUs, autonomous driving systems, and advanced driver assistance systems (ADAS).

Overall, the purpose of this project is to provide case studies or examples of successful implementation of AUTOSAR and Adaptive AUTOSAR in real-world automotive projects. This could help readers understand the practical benefits of these standards and how they can be applied in specific contexts.

AUTOSAR(AUTomotive Open System ARchitecture)

The Automotive open system architecture (AUTOSAR) standard is a development partnership of companies and organizations from the automotive, electronics, semiconductor and software industries. AUTOSAR is a standard defining embedded software and a development flow that supports tasks surrounding basic automotive functions in the context of vehicle system development.

AUTOSAR Advantages

Since 2003, AUTOSAR has brought together the combined knowledge and experience of most major automotive manufacturers and suppliers.The middleware platforms specified by AUTOSAR are supported by workflow and exchange file formats, defined in the standards, supporting an open ecosystem, available from a selection of members. AUTOSAR supports a comprehensive set of automotive use cases. It integrates or interoperates with the technologies needed in vehicle ECU software and e/e architectures – e.g., SOME/IP, DDS other middleware types used as part of a heterogeneous software platform and standards such as those from ASAM, IEEE and more.

AUTOSAR Classic Platform

This software platform is suitable for a wide range of statically defined functions, used in traditional automotive ECUs with simple or complex functions. The AUTOSAR Classic Platform architecture supports a range of automotive network technologies, CAN, LIN, FlexRay and Ethernet and functional safety up to ASIL D. It can be extended to support cyber security. OSEK-based, API in C, application code – commonly C or C++ – is often generated from control engineering models.

The AUTOSAR classic platform is the standard for embedded real-time ECUs based on OSEK. Its main deliverable is specifications.

The architecture distinguishes between three software layers that run on a microcontroller: application, runtime environment (RTE) and basic software

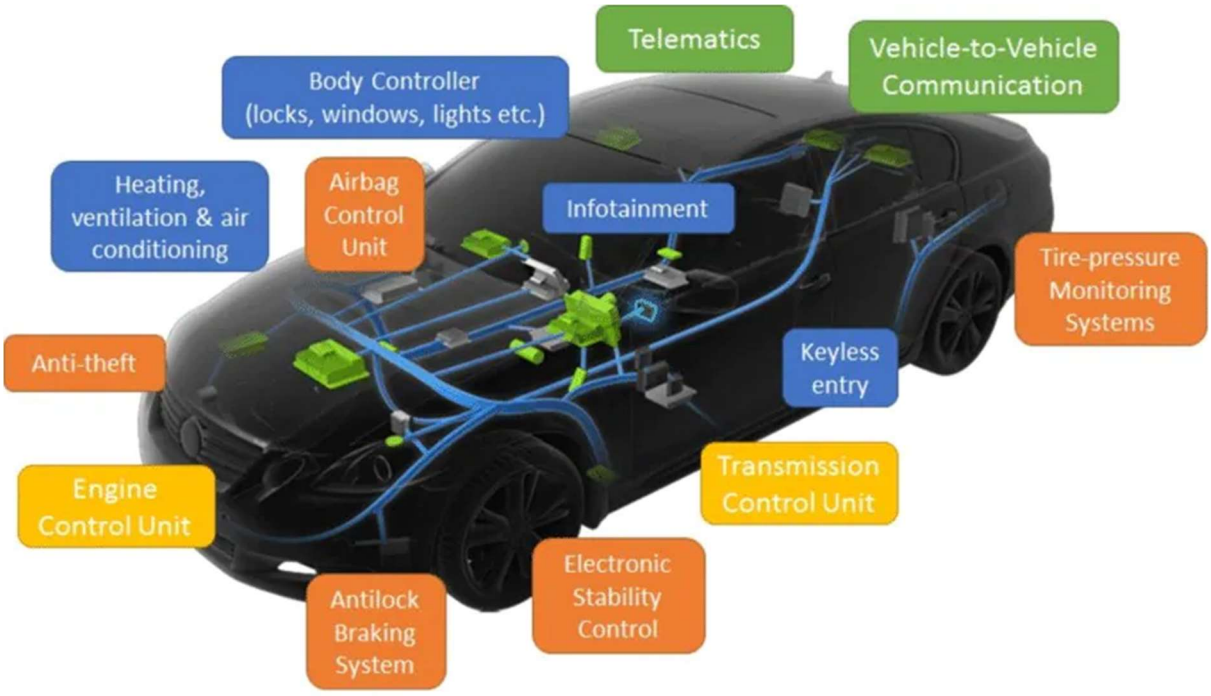
(BSW). The application software layer is mostly hardware independent. Communication between software components and access to BSW happens via RTE, which represents the full interface for applications.

The BSW is divided in three major layers and complex drivers:

- Services
- Electronic control unit (ECU) abstraction
- Microcontroller abstraction

Services are divided further, into functional groups representing the infrastructure for system, memory and communication services.

One essential concept of the Classic Platform is the Virtual Functional Bus (VFB). This virtual bus is an abstract set of RTEs that are not yet deployed to specific ECUs and decouples the applications from the infrastructure. It communicates via dedicated ports, which means that the communication interfaces of the application software must be mapped to these ports. The VFB handles communication within the individual ECU and between ECUs. From an application point of view, no detailed knowledge of lower-level technologies or dependencies is required. This supports hardware-independent development and usage of application software.



AUTOSAR partners

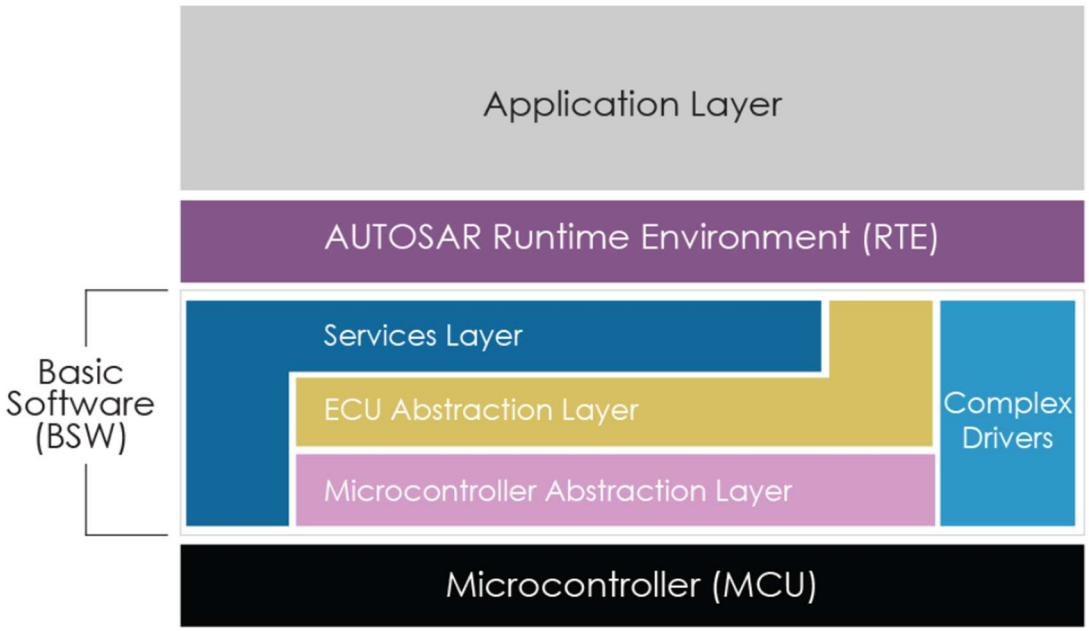
The AUTOSAR standard is developed and maintained by its partners, with consideration of the use cases needed to support the roadmaps of the users. Partners are categorized by their type of membership. Multiple categories exist for different levels of contribution, development, implementation and usage of the standard. The main categories are:

- Core partners – the founding members of the standard.
- Premium partners, including Siemens – typically actively involved in leading the development of the standard and its usage.
- Development partners – are typically users of the standard and contributors to the development of the standard.
- Associate partners – typically have a production usage of AUTOSAR planned or in place but take a limited role in developing the standard.
- Core Partners include the founding partners BMW, Bosch, Continental, Daimler AG, Ford, General Motors, PSA Peugeot Citroën, Toyota, and Volkswagen. These companies are responsible for organization, administration and control of the AUTOSAR development partnership. Within this core, the Executive Board defines the overall strategy and roadmap. The Steering Committee manages day-to-day non-technical operations and admission of partners, public relations and contractual issues. The Chairman and Deputy of Chairman, appointed for one year, represent the Steering Committee for that purpose. The AUTOSAR Spokesperson takes over the communication with the outside world.
- Strategic partners are appointed for a period of two years from the circle of Premium Partners and support the project leader team in the various technical, organizational and everyday processes. They also give new strategic inputs to the project leader round.
- Premium and Development members contribute to work packages coordinated and monitored by the Project Leader Team established by the Core Partners. Associate partners are making use of the standard documents AUTOSAR has already released. Attendees are currently participating with Academic collaboration and non-commercial projects.

AUTOSAR Architecture

AUTOSAR is an open system architecture for automotive software development and provides standards for developing common automotive software applications. It is a growing and evolving standard that defines a layered architecture for the software. The classic AUTOSAR platform runs on a microcontroller and is divided into 3 main layers:

- Basic Software Architecture- It is common to any AUTOSAR ECU.
- AUTOSAR Runtime Environment
- Application Layer



AUTOSAR Architecture

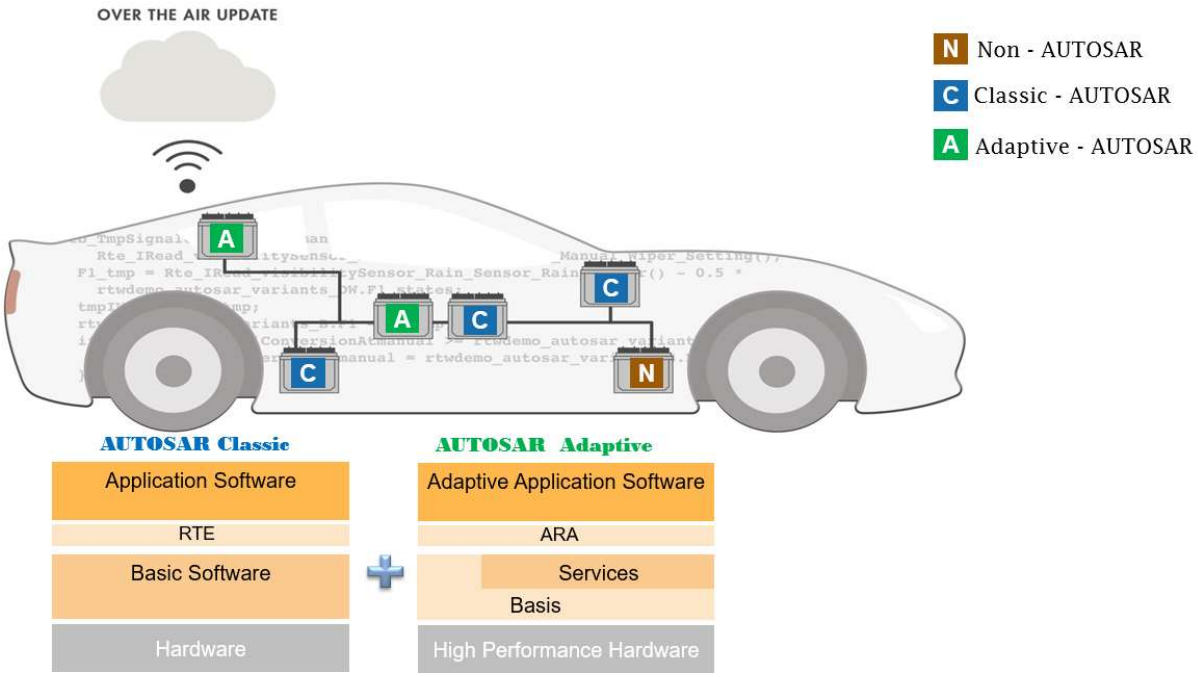
Basic Software Architecture (BSW)

AUTOSAR Basic Software Architecture consists of hundreds of software modules structured in different layers and is common to any AUTOSAR ECU.This means the supplier who has designed BSW can share it with other suppliers that are working on ECUs of engine, gearbox, etc.

Basic software architecture in AUTOSAR consists of three layers:

- **Microcontroller Abstraction Layer (MCAL):** MCAL is also known as a hardware abstraction layer and implements interface for the specific microcontroller. MCAL has layers of software, which are integrated with the microcontroller through registers, and offers drivers like system drivers, diagnostics drivers, memory drivers, communication drivers (CAN, LIN, Ethernet, etc.), I/O drivers and more.

- **ECU Abstraction Layer:** The prime task of the ECU abstraction layer is to deliver higher software layers ECU specific services. This layer and its drivers are independent of the microcontroller and dependent on the ECU hardware and provide access to all the peripherals and devices of ECU, which supports functionalities like communication, memory, I/O, etc.
- **Service Layer:** The service layer is the topmost layer of AUTOSAR Basic Software Architecture. The service layer constitutes an operating system, which runs from the application layer to the microcontroller at the bottom. The OS has an interface between the microcontroller and the application layer and can schedule application tasks. The service layer in BSW is responsible for services like network services, memory services, diagnostics service, communication service, ECU state management, and more.



AUTOSAR Runtime Environment (RTE Layer)

AUTOSAR Run-time Environment is a middleware layer of the AUTOSAR software architecture between the BSW and the application layer and provides communication services for the application software.

Application Layer

The application layer is the first layer of the AUTOSAR software architecture and supports custom functionalities implementation. This layer consists of the specific software components and many applications which perform specific tasks as per instructions.

The AUTOSAR application layer consists of three components which are: **application software components, ports of software components, and port interfaces.**

AUTOSAR ensures standardized interfaces for software components in the application layer and application software components help in generating simple applications to support the vehicle functions.

The communication between software components is enabled via specific ports using a virtual Function Bus. These ports also facilitate communication between software components and AUTOSAR Basic Software (BSW).

The above-explained architecture of AUTOSAR is its classic platform, which supports real-time requirements and safety constraints. Based on the microcontroller, the classic platform is capable of supporting applications in the field of networking and security by allowing ECUs to access vehicle sensors and actuators.

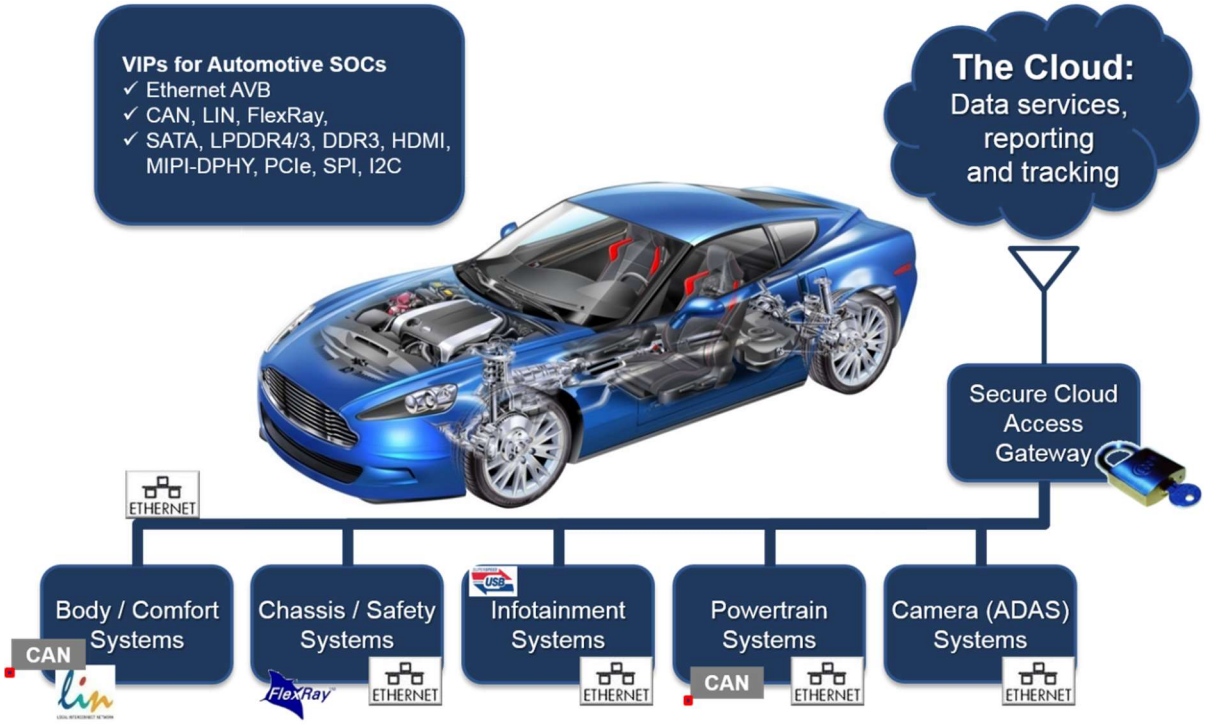
Adaptive AUTOSAR

Why Adaptive AUTOSAR?

From 2003 to 2015, Classic AUTOSAR became an established platform and was doing perfectly well to run 60-80 ECUs in a vehicle. With the evolution of IoT based automotive trends like V2X connectivity and automated driving, electrification skyrocketed, and as a result, a huge demand for supporting functions and devices was created in the market. It was discovered that the existing classic AUTOSAR platform was not suitable to support these mega-trends and new architecture with more powerful and flexible E/E architecture is required. Adaptive AUTOSAR, the new architecture was released to support these functions and 1st release of AUTOSAR adaptive platform was done in March 2017.

Adaptive AUTOSAR architecture comes with a central application server, which assists high-performance computing. Ethernet-based ECUs in this system supports real-time functionalities. Adaptive AUTOSAR is scalable and has dynamic architecture, in which applications can be updated over the vehicle's lifecycle. It enables OEMs to deploy high-tech software features in a vehicle and update them over-the-air whenever required.

AUTOSAR adaptive architecture supports all the futuristic automotive applications like infotainment, v2x, predictive maintenance, automotive apps, ADAS functions with a camera, RADAR and LIDAR sensors, map updates, electrification, and more.



The Outlook on Future Needs

New functionalities such as environment perception, autonomous driving, vehicle connectivity and Car as a Service (CaaS), have introduced a new set of problems to the electrical/electronic (E/E) panorama. These new functionalities differ from the ones ClassicAS was designed for. They no longer simply substitute electromechanical systems. They are, for lack of a better description, deeply autonomous and, dare I say it, *Intelligent*. As such, they are highly complex and have high processing needs. Furthermore, these new functionalities are expected to evolve during the vehicle's lifetime, in contrast to the not so significant changes that a traditional function could have. High complexity, high computing power, faster communications and continuous updates are, to name a few, the requirements set into stone by these new trends. The technology which enables the new functionalities is already here: ethernet and HPC.

HPC means the joint usage of not one or two, but many cores, processors, GPUs and other processing units. ClassicAS, already struggling with multi-core

ambients, cannot define a solid structural model for the new processing needs. Similarly, even though the higher bandwidth provided by ethernet is already in use by ClassicAS, this architecture cannot completely harness its power, and so its true potential in the automotive industry is yet to be discovered. Furthermore, the static nature of ClassicAS makes it inflexible in certain aspects. For example, its scheduling algorithm doesn't allow it to dynamically assign tasks to cores, meaning the usage at one core could be nearing 100%, while another one is barely used. The dynamicity offered by AdaptiveAS comes to the rescue and expands the solution to the many-cores architecture.

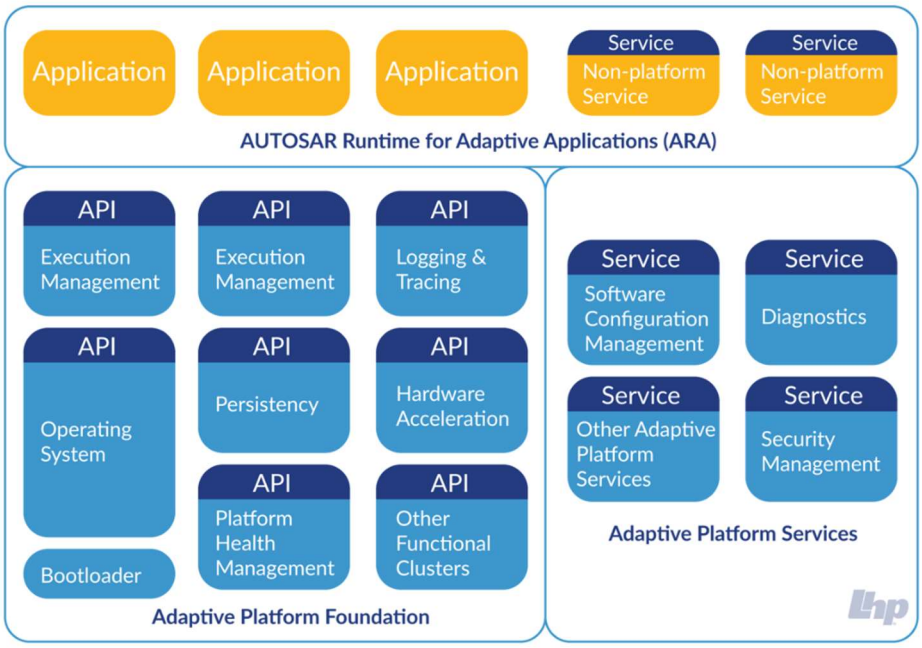
In simple terms, **Adaptive AUTOSAR** is a new standard made to cover the new needs and new resources previously mentioned. It does so by offering dynamicity at heart and by using the capabilities provided by high performance computing.

Adaptive AUTOSAR Architecture

AUTOSAR introduced a layered architecture, so it is only natural that **Adaptive AUTOSAR** also uses a well-defined one, albeit with not only small differences, but also unique characteristics that make it so perfectly suited for the problems and requirements of a new era. The AdaptiveAS architecture is based on key principles, such as C++ as the preferred language, Service Oriented Architecture (SOA), dynamicity and an inherent parallelism.

SOA can be defined as a model in which services are made available across a network. The services can reside either on the same ECU or externally in another ECU. Furthermore, the services are self-contained units; the clients (the users of services) are independent from the services. In addition, the coupling between services and clients is dynamic at runtime (in contrast to compile time from ClassicAS). Applications can consume services (i.e., be clients) or also offer services to other applications. In other words, SOA is a model in which applications openly present their utilities as services across a network. In this

network other applications can request these services in a dynamic and therefore “adaptive” manner.



Adaptive AUTOSAR Applications

At the top layer we can find the new Adaptive Applications (AA). AAs, in contrast to software components (SWCs), are now processes which may be single or multithreaded. Another important characteristic of them is they are now able to be started or stopped at any time. Even though a similar scenario can be done in ClassicAS using partitions, the key aspect lies once again in the static (ClassicAS) vs. dynamic (AdaptiveAS) behavior. AAs can be added (downloaded) on their own, while adding a new partition would require us to reflash the whole software. In addition, AAs are now held together by a familiar, yet completely new entity, the **AUTOSAR Runtime for Adaptive Applications (ARA)**. This new ARA is made up of application interfaces coming from the building blocks of the next layer, that is, the functional clusters. Moreover, such functional clusters can be either from the Adaptive Platform Foundation or the Adaptive Platform Services. Another key aspect to understanding the ARA is that, in contrast to the **runtime environment (RTE)** and therefore with the ClassicAS, in AdaptiveAS the clients and services are dynamically linked at runtime. However, AdaptiveAS

also allows options to limit the dynamic behavior in order to decrease the risk of undesired effects. For instance, some of these options include fair scheduling, dynamic memory allocation only in the startup phase, and even fixed allocation of processes to CPU cores. In other words, AdaptiveAS doesn't forget its target audience and offers planned dynamics.

Functional Clusters

The software of the **Adaptive Foundation** and **Adaptive Services** is presented in the form of functional clusters. Simply put, a functional cluster is a set of requirements grouped by the aspect they refer to. These functional clusters are, in a manner, similar to the concept of the basic software (BSW) in the Classic Platform. Their main purpose is to offer functionalities in the form of services to the application. However, being part of the new Adaptive Platform, and in contrast to the BSW, the new functional clusters are now processes, which can be single threaded and multithreaded. Another key feature is the dynamicity vs. the traditional static nature of the BSW. Just as in the ClassicAS, there are needs for non-volatile storage, communication and diagnostics, to name a few, which are taken care of by **non-volatile memory (NVM)**, Com, **Diagnostic Communications Manager (DCM)** and **Diagnostic Event Manager (DEM)**. These needs are catered in the AdaptiveAS through functional clusters, such as Persistency, Communication Management and Diagnostic Management. Although similar in purpose, they differ in nature. For instance, Persistency is based on key-value and stream storages, and Communication Management drops the signal-based approach in favor of a service-oriented strategy.

Machine

The new term, "Machine," given by the AdaptiveAS is, in simple terms, the entity where the software runs. It is understood that this Machine can be of virtual nature. The actual hardware can in turn host one or multiple Machines.

Operative System

The new resources like HPC and SOA also call for new prerequisites. One of them is an operating system (OS) capable of meeting their needs (dynamic environment, mix of computing resources). Therefore, the static nature of AUTOSAR OS and OSEK is not well suited for the job. However, instead of defining a new OS, the solution here is to define an Operative System Interface (OSI). The OS must be Portable Operating System Interface (POSIX) compliant. AAs have been given a more constrained interface – PSE51 (some characteristics include: no use of file system, single process) – while the Foundation and Services are allowed to use full POSIX. One may ask, "Why select PSE51?" The answer is simple: to achieve freedom on interference, and to offer portability for existing POSIX applications.

But what about the so needed real-time organization in embedded systems? After all, predictability is of the essence. Luckily for us, scheduling policies for real-time applications are available in the POSIX standard, analogous to the priority-based scheduling from AUTOSAR OS, SCHED_FIFO, SCHED_RR and SCHED_DEADLINE are present and allowed.

Following the architectural definitions of adaptive applications and functional clusters, one can easily make the analogous comparison to the software components and the basic software of the ClassicAS. A following point to ask would be, "How do they communicate with each other?" In the ClassicAS all communications among SWCs must go through the RTE. This is accomplished using ports and connectors. In a similar manner, the interaction between SWCs and the BSW is also done through the RTE using standardized interfaces with the service flag set. However, the BSW communicates with each other using common C calls.

We can use the precedent laid out by the Classic Platform in order to understand how the AdaptiveAS will manage the interactions. In the Adaptive Platform both adaptive applications and functional clusters are implemented as processes, albeit as was previously stated, the adaptive applications are bound to only use PSE51, which as part of being a more constrained interface, does not offer Inter

Process Communication (IPC). If PSE51 does not include IPC, then how can applications interact with one another?

For this new problem, the Communication Management (CM) has been defined. It provides an SOA-like communication – in fact, service-oriented communication (SOC) – for applications and can be exploited for inter and intra machine interactions in a transparent way. This is, in a manner, similar to how one SWC can connect to another SWC in the same ECU or with a SWC in a different ECU.

Functional clusters, on the other hand, are not bound by PSE51 and can use IPC to interact with each other. What is probably more important to note is, how they interact with AAs. In this case there are two options: one library-based (the interface library linked to AA calls IPC), and the other service-based (the process uses CM).

Adaptive AUTOSAR and Classic AUTOSAR are different in the manner that each target a distinct set of ECUs, with separate needs and resources allocated to them.

Differences Between Classic AUTOSAR and Adaptive AUTOSAR

- Classic AUTOSAR is meant to be run in a single- or multi-core architecture. Adaptive AUTOSAR is intended to run on top of and take advantage of HPC architectures.

- Classic AUTOSAR defines an Operative System. Adaptive AUTOSAR only defines an execution context and an Operating System Interface (POSIX, PSE51).

- Classic AUTOSAR is defined in a signal level. Adaptive AUTOSAR is defined in a service manner.

- Classic AUTOSAR is static in nature. Adaptive AUTOSAR offers “planned dynamics” both in application deployment, as well as communications and resources.

- Classic AUTOSAR is meant for the deeply embedded ECUs. Adaptive AUTOSAR is intended for a new generation of ECUs.

Lastly, **Adaptive AUTOSAR** is not meant to replace **Classic AUTOSAR**; both are meant to co-exist and cooperate in satisfying the needs and challenges presented to the automotive industry.

Conclusion

In conclusion, AUTOSAR (AUTomotive Open System ARchitecture) and Adaptive AUTOSAR are two software architectures designed for the automotive industry. While both architectures aim to standardize and streamline the development of automotive software, they differ in their target applications and characteristics.

AUTOSAR is a standardized platform for developing software for embedded systems in vehicles, providing a common framework for different software components to communicate and work together. It has been widely adopted in the industry and has proven to be effective in reducing development time and costs.

Adaptive AUTOSAR, on the other hand, is a newer architecture that focuses on more advanced and complex functions, such as autonomous driving and advanced driver assistance systems. It provides a flexible and scalable platform that can adapt to different hardware and software configurations and can support various levels of functional safety.

References

https://www.lhpes.com/blog/what-is-adaptive-autosar

https://www.renesas.com/us/en/application/automotive/common-automotive-technologies/autosar/autosar-layered-architecture

https://www.autosar.org/

https://en.wikipedia.org/wiki/AUTOSAR