Abstract:
This paper includes brief description of the basic technical concepts of AUTOSAR. AUTOSAR (Automotive Open System Architecture) is an open and systemized automotive software architecture which was jointly developed by automobile OEM (Original equipment manufacturers). To improve cost and reusability, AUTOSAR separates application software from the associated hardware. This stage, AUTOSAR is still in the promotion phase, and many countries are committed to its applications. AUTOSAR concept is the basis for development of the new modules inside of the automotive systems and represents the newest worldwide automotive trend. The purpose is to create and establish an open and standardized software architecture for electronic vehicle control units (ECU). The AUTOSAR standard will serve as a platform upon which future vehicle applications will be implemented.

I. INTRODUCTION:
Automotive Open System Architecture (AUTOSAR) is an international automobile develop affiliation mounted in 2003. AUTOSAR (Automotive Open System Architecture) is a collaboration between car manufacturers. The main goal of collaboration between manufacturers and suppliers is to create and define an efficient E/E architecture for the automotive industry through open industry standards. The main partners are the founding partners of BMW, Bosch, Continental, Daimler AG, Ford, General Motors, PSA and Toyota.

STANDARDIZATION:
By the technical purpose of view, following are the guiding forces for the projected standardization:
- Manage the ever-increasing E/E complexity associated with feature growth.
- Increased flexibility when upgrading and upgrading product modifications.
- Improve quality and reliability of E/E system
- Improve scalability of solution within and across product line
- Allows the detection of errors in early design phases.

OBJECTIVE:
- Hardware abstraction
- BSW standardization
- Standardization of exchange formats
- Reusability of function
- Interface standardization

FUNCTIONAL DOMAIN:

II. LITERATURE SURVEY:

1) AUTOSAR Software Architecture BY Robert Wars of sky:
This paper gives an overview about the different layers of that architecture. In addition, the upper most layer that concerns the application specific part of automotive electronic systems is presented.

2) Basic Concepts on AUTOSAR Development Huang Bo; Dong Hui; Wang Dafang; ZhaoGuifan
IEEE Conference: This paper showed the basic concepts of AUTOSAR and comparisons of some confusing concepts for future in-depth study.

3) A Survey on the Benefits and Drawbacks of AUTOSAR Silverio Martínez-Fernández, Claudia
P. Ayala, Xavier Franch, Elisa Y. Nakagawa IEEE Proceedings of the First International Work shop on Auto motive Software Architecture-WASA2015: As a software reference architecture, AUTOSAR provides a blueprint for developing software architectures for auto motive applications

4) AUTOSAR for Connected and Autonomous Vehicles Simon Fürst, Dr.-Ing Markus Bechter. 46th Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshop (DSN-W),2016:
Over the past 10 years AUTOSAR has demonstrated to be the best established and well-suited organization to coordinate and drive the standardization for software infrastructures meeting the requirements of the automotive industry.

5) Building Automotive Software Component within the AutoSAR Environment – A case study Gia Nghia Vo, Richard Lai, Mohit Garg. IEEE Ninth International Conference on Quality Software 2009: In this paper, it
contains about the modeling of a Power Window Control System using UML and the tool - Enterprise Architect

INTRODUCTION TO THE AUTOSAR CONCEPT:

The basic components are listed below:
➢ AUTOSAR SW-C: It includes an application which runs on the AUTOSAR infrastructure.
➢ SW-C Description: for the interfacing and for the integration of the AUTOSAR software components it provides standard description format, i.e., the SW-C description.
➢ Virtual functional bus (VFB): It is the technology independent level. And it is the sum of all communication mechanisms provided by AUTOSAR.
➢ System constraint and ECU Description: these are used to provide integration of AUTOSAR SW-C into ECUS.
➢ Mapping on ECUS: it contains the information especially about the generation of the Runtime environment and the basic software on each ECU.
➢ Runtime environment: it performs the functionality of VFB on a specific ECU.
➢ Basic software: provides infrastructural functionality on ECU.

III. AUTOSAR METHODOLOGY:

AUTOSAR requires a unified technical approach at certain stages of system development. This method is called the "AUTOSAR method". This chapter introduces all the important stages of system development using AUTOSAR: from system-level configuration to the creation of executable ECUs. The AUTOSAR method is not a complete description of the process or business model, and the "roles" and "responsibilities" are not defined in this method. In addition, there is no requirement for the order in which the operations are performed. This method is a simple work product process: it defines the dependence of the activity on the work product.

IV. OVERVIEW OF METHODOLOGY:

Figure 3b gives a rough overview of the AUTOSAR method. It describes the design steps from system-level configuration to the creation of executable ECU files. First, you need to define system configuration records. This is a system architecture or design task. You should select hardware and software components and define general system boundaries. AUTOSAR intends to simplify the formal description of these initial system design decisions by using information exchange formats and templates. In order to define the system configuration data record, it is necessary to fill in or edit the corresponding template.

This applies to the information in the following software packages:
➢ Software components-Each software component requires a description of the software API, data types, ports, interfaces, etc.
➢ ECU resources: specifications required for each ECU Compared to Processor, memory, peripherals, sensors and actuators.
➢ System limitations-contains electronic limitations related to the display of bus signals, topology and related software components.

It depends on the use case, whether you need to create the template from scratch, or whether it can be reused by doing
some editing. In fact, in this case, the AUTOSAR method is highly reusable. Supported by editing tools.

The "configuration system" operation mainly allocates software components to the control unit based on resource and time requirements. The result of this operation is a description of the system configuration. This description contains all system information (for example, bus display, topology) and the display of which software component is located in which control unit. Other steps must be performed for each ECU in the system (as shown in Figure 3b). The specific system file extracts the information from the system configuration description required by the specific ECU, and then pastes it into the ECU system configuration code segment. The Configure ECU operation will add all the information required for implementation, such as: Task plan, necessary basic software modules, basic software configuration, and information such as assigning executable files to tasks, to a specific control unit. The executable software for that particular controller can be created based on this information. In the final step of co-mailing the executable file, the executable file is created based on the ECU configuration, which is described in the ECU configuration description. This step usually involves generating code (for example for RTE and basic software), compiling the code (compiling the generated code or compiling software components that can be used as source code) and linking all elements into an executable file. In parallel with these briefly described method steps, several steps are required, for example, the integration of software components into a complete system, to generate component API and implement component functions. However, this software implementation is not shown in Figure 3 for clarity. However, the realization of the software components is more or less independent of the ECU configuration. This is the basic function of the AUTOSAR method.

LAYERED ARCHITECTURE:

Based on the ECU software architecture, layered architecture has been developed with AUTOSAR to allow a clear and structured interface definition and well-defined abstraction of hardware. The multi-layer software architecture has 5 layers and can implement complex device drivers that cannot be synchronized on the same layer.

REFINED LAYERED OF SOFTWARE ARCHITECTURE:

In order to make the software architecture more modular view, the layer architecture introduced of resources in the ECU, thus limiting possible allocations. Finally, an important aspect of this activity is the development of a matrix for system communication. The system interaction matrix completely describes the structure of the network. These elements are described in the topology and the content and timing of these frames. In order to assign all software components to the control unit, knowledge of the system architecture is very much needed. The AUTOSAR system adjustment tool supports adjustment. This will help you make the above technical decisions (for example, through a clear graphical representation), save the results, and make changes later if necessary. About 80 basic software modules have been defined. The description of all these modules will go beyond the scope of this overview. Therefore, a more abstract point of view was proposed as shown in the figure6.

V. FUNCTIONAL INTERFACES:

The software that realizes the functions of the car is mainly encapsulated in software components. The interface standardization of AUTOSAR software components is a key element to support the scalability and portability of the functions of electronic control units across different platforms. All specified functions must correspond to AUTOSAR software components. It must have clear interface semantics and be published in the AUTOSAR distribution function catalogue. The goal is to integrate each standard-compliant implementation of software components into the system with less work. In this sense, compliance with the standard will mean that the component provides certain functions, which are precisely defined by the fully defined AUTOSAR interface.

Nevertheless, the standardization may be developed incrementally where attainable entities for standardization might be:

- Level of abstraction
  - Functional aspects
  - Behavior and implementation aspects
- Level of decomposition
  - Low degree of decomposition of the useful domain
  - High degree of decomposition of the useful domain
APPLICATION OF AUTOSAR:

- Powerful interaction with devices (sensors and actuators), with limited computing power and memory resources (compared to company solutions).
- Connect to automotive networks, such as CAN, LIN, Flex Ray or Ethernet microcontrollers (usually 16 Bit or 32bit).
- Real-time execution of programs and systems. From internal or external flash memory.

ADVANTAGES OF AUTOSAR:

- Standardization of specification change formats
- Layered structure of Basic Software(BSW)
- Software sharing among companies

VI. CONCLUSION:

With AUTOSAR, you can develop application software at an abstract level, with special consideration for the interaction between software components. It provides a modular architecture with a layered structure and standardized interfaces, which connect the layers and components of the architecture.

VII. REFERENCES:
