Autosar

Automotive Open System Architecture

How are vehicle functions implemented today?

• Each function has it´s own system although they may communicate through a bus

• Each function has it´s own microcontroller

• The number of ECU´s (Electronic Control Unit) are growning fast

• Hardware and software are tightly connected

• The same vendor supplies both the hardware and the software

• There are no alternative software suppliers

What will Autosar give?

- A standard platform for vehicle software
- An OS with basic functions and interface to software
- The same interface for all basic software
- No applications of its own
- Functionality is supplied as software components
- These components are hardware independent
- The software is exchangable
- The software can be reused
- More than one supplier can compete with their software

What will Autosar give cont?

- Autosar has a layered architecture that frees the applications from the hardware
- This layered architecture is implemented in every ECU
- You do not have to think in terms of ECU´s when you design the system
- The designer picks software components without knowing in what ECU they will be implemented

The story of Autosar

- It all started when the semiconductor vendors needed a driver layer for their microprocessors

- The work started in 2002 in Tyskland within BMW, VW, Daimler Chrysler, Bosch and Continental

- Autosar was founded in 2003 and the work started in 2004

The story of Autosar cont.

- Autosar has four types of members

- Only leading vehicle vedors and subcontracters can be Core partners. These are in charge of organisation and management

Within this group are Peugeot Citroën, Toyota, VW, BMW Daimler, Ford, Opel (GM), Bosch, Continental, Siemens and Volvo
Ford has been represented by Volvo

Saab has had a key role in the engagement from GM

Thew story of Autosar cont.

- The other forms of membership are
- Premium member
- Associate member
- Development member

Autoliv is a premium member

The members work in work groups

The first test

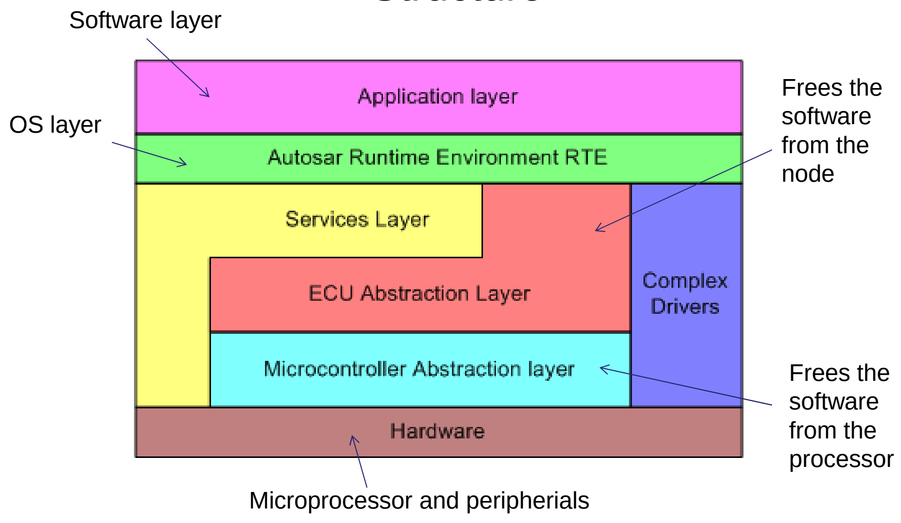
After the first specification 31 software components were ordered from 15 vendors och these components were realised in 56 implementations

The components were installed into two different systems. One 16 bit system and one 32 bit system

The test led to 260 suggestions for changes in the specification

Since then there has been few new suggestions for changes but the standard has developed and growed

Structure



Software layer
OS layer
Application layer
Autosar Runtime Environment RTE
Services Layer
ECU Abstraction Layer
Microcontroller Abstraction layer
Complex Drivers
Hardware
Frees the software from the node
Frees the software from the processor
Microprocessor and peripherials

Operative system

The Autosar OS is a derative of OSEK

Offene Systeme und deren Schnittstellen für die Elektronik in Kraftfahrzeugen

Open Systems and their Interfaces for the Electronics in Motor Vehicles

The OS is responsible for real time functions, priority based scheduling and different protective functions

Some systems will continue to use their own OS but these must have an interface to Autosar that follows the specifications from Autosar

Virtual Function Bus, VFB

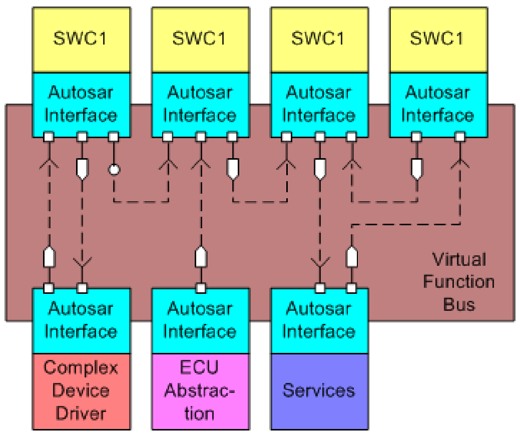
The application components are linked and communicate through an virtual bus, Virtual Function Bus, VFB

VFB is a visualization of all hardware and the system services that the vehicle system supplies

Through VFB a software component doesn´t need to know which components it is communicating with and on which ECU these components are implemented

The VFB is implemented through Autosar Runtime Environment (RTE) and the layer of Basic Software

Virtual Function Bus, VFB cont.



Autosar Runtime Environment, RTE

Through Autosar Real-time Environment, RTE, the software components can communicate without being mapped to specific hardware or ECU

RTE frees the software components from the hardware and from each other

Every ECU in a Autosar system must implement a RTE

RTE uses the hardware abstraction layer Microcontroller Abstraction Layer, MCAL

The application layer, the software layer

The functionality of the applications reside in the application layer

This is the only part of an Autosar system that doesn´t consist of standardized software

A Software Component, SWC, is the smallest part of a software application that has a specific functionality

A software application can be built out of a number of software components

Within Autosar there are standard interfaces so that the components can be used to build the applications

Communication between software components

A SWC can communicate in two different ways

- Client/server

The client initiates the communication and requests a service from the server

The client could be locked while it is waiting for an answer from the server

The Client/server roles are defined by who is initiating the communication and could be switched

A SWC can at the same time act as both client and server in different communications

Communication between software components cont.

- Sender/receiver

The sender expects no answer from the receiver and there will be no answer

The sender is not blocked

The receiver desides on it´s own how and when to act on the information

The interface structure is responsible for the communication and the sender doesn´t know who the receiver is or if there are more than one receiver

The sender doesn´t know in what ECU the receiver is situated

Basic Software Layer

Basic software layer is standardized software without any own functionality that offer both hardware dependent and hardware independent services to higher layers

This takes place through Application Programming Interfaces

This layer makes the higher layers hardware independent

Here we have for example memory interfaces and interfaces to communication busses like LIN, CAN and FlexRay

Microcontrol Abstraction Layer MCAL

Microcontrol Abstraction Layer, MCAL consists of standardized functions that frees the hardware from the software and gives a standardized interface to Basic Software

MCAL handles the microcontrollers peripherial units and supplies processorindependent values to the Basic Software

Through this layer the software is prevented from directly accessing the registers in the moícrocontroller

In this way the microcontroller is abstracted from higher layers and they become processor independent

Development process

The model descriptions within Autosar are standardized to become tool independent

Autosar has given a method for creating the system architecture that starts in the design model

The descriptions have UML syntax (Unified Modeling Language)

The basic system descriptions are independent of how they are to be implemented

Necessary data are among others interface and hardware demands

Standard interfaces are described in XML (eXtendable Mark-up Language)