From Signal to Service

Challenges for the Development of AUTOSAR Adaptive Applications

Automotive Ethernet and AUTOSAR Adaptive are key technologies for highly automated driving and comprehensive connectivity services. AUTOSAR Adaptive relies on a service-oriented architecture to ensure communication between the ECUs. This raises many new challenges at the level of the development, test and calibration tools.

Today's applications in the automotive field primarily make use of sensor/actuator combinations. In these solutions, the participating software and hardware components are closely coupled. The communication relationships between the participating ECUs are statically defined during the system design phase and are based on signal-oriented communication. The focus is placed on hard real-time capabilities and the use of cost-effective microcontrollers as well as of networks such as CAN and LIN. The established AUTOSAR Classic platform reliably covers these requirements. The focus is not on extending or adding applications.

Highly-automated driving and back-end connectivity demand dramatically enhanced computational performance and more flexible communication concepts. In addition, it must be possible to extend or add applications across the entire vehicle lifecycle. Ethernet is a suitable transmission medium for this generation of applications. On the one hand, it provides the bandwidth that is required, for example, for the transfer of image data. On the other, IP-based protocols running on Ethernet allow the simple integration of end-user devices and access to back-end infrastructures.

AUTOSAR Adaptive – A Brief Overview

The AUTOSAR Adaptive platform **(Figure 1)** addresses the requirements of these future applications. It forms the central computing cluster for complex applications and establishes the connection to back-end services. At the same time, it interacts with AUTOSAR Classic ECUs in order to control the sensors and actuators. Consequently, AUTOSAR Adaptive is not a substitute for the Classic platform but instead a complementary development platform. The main differences between the two platforms are outlined in **Table 1**.

As the central architectural pattern, AUTOSAR Adaptive adopts a service-oriented approach **(Figure 2)**. Within this, the overall system is modeled as a set of service interfaces.

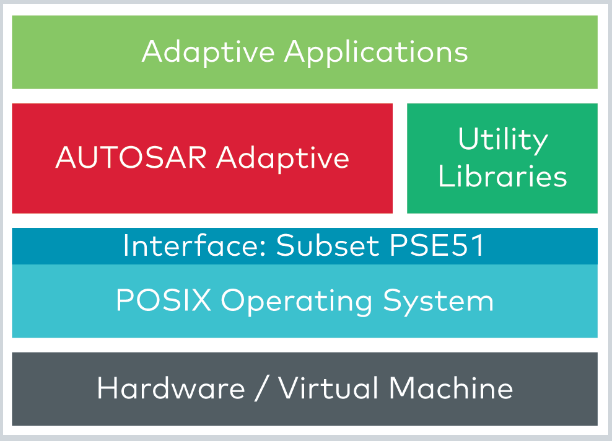


Figure 1: The AUTOSAR Adaptive platform forms the central computing cluster for complex in-vehicle applications.

These encapsulate a certain sub-functionality comparable to a class in object-oriented programming. The service interfaces are implemented by so-called providers and are used by consumers. The communication path between a provider and a consumer is no longer defined at development time but only at runtime. The establishment of the communication paths and the conversion to a physical transmission medium is performed by a middleware. This approach permits the flexible provision and updating of applications across the entire vehicle lifecycle.

Currently, AUTOSAR Adaptive supports the Ethernet-based SOME/IP protocol (Scalable service-oriented middleware over IP) as the middleware protocol. In the future, further protocols will also be integrated. REST-based (Representational State Transfer) services using HTTP are expected to be used here for communication to the back-end services, for example. With AUTOSAR Adaptive, system architectures from conventional information technology are making their way into vehicle applications.

Testing Adaptive Applications

The introduction of these system architectures is confronting development and testing engineers with new challenges. Development and test methods from the conventional ECU development field have to be extended and complemented. Thanks to the service-oriented approach, future applications will be developed largely independent of any assignment to a specific ECU. Suppliers will frequently not provide a fully integrated ECU but instead only a set of applications. To do this, they must be able to test these in isolation. Similarly, in the future, vehicle manufacturers will perform testing more intensively at the level of individual subsystems. This is the only way to control the rapidly growing complexity of the overall system.

The tests of the lower communication layers must be strictly separated from the application tests. The mapping of service interfaces to concrete network protocols should therefore take place downstream and be interchangeable. Such an approach makes testing possible throughout the entire development process. Tests and simulation models can be defined in the early phases and then be extended incrementally. This permits the gradual transition from purely simulated to partially simulated environments.

The service interface level is clearly suitable for application-related testing. These interfaces encapsulate the behavior of the applications and form a clearly defined system boundary. They are defined within the framework of a service-oriented system design and are available even in

	AUTOSAR Classic	AUTOSAR Adaptive
Initial Release	2005	2017
Programming Language	Procedural (C)	Object-oriented (C++)
Communication Model	Signal-oriented – with focus on CAN, LIN, FlexRay	Service-oriented – with focus on Ethernet
Operating System	OSEK-based > Cooperative multitasking > Static number of tasks	POSIX-based (PSE51) > Preemptive multitasking > Dynamic number of tasks

Table 1: The most important differences between AUTOSAR Classic and AUTOSAR Adaptive

early development phases. AUTOSAR Adaptive describes the service interfaces and data types within the standardized exchange formats. As a result, the programming interfaces required for testing are generated automatically.

For development and test engineers, a type of service toolbox from which they can choose the service interfaces required for the current test goal would seem useful here. These interfaces are then stimulated and analyzed in the test environment. In such cases, it should be possible to modify the timing conditions during a test, for example in order to check how a consumer reacts to different provider response times.

The dynamic establishment of communication paths at runtime is a further challenge. First of all, it is necessary to take account of the fact that the availability of providers and consumers can change at runtime. Secondly, in some scenarios, the participating static communication endpoints are unknown, for example in the case of data exchange between vehicles. In such cases, development and test engineers need a flexible way of varying the number of participating communication endpoints.

Tools for Service-oriented Testing

One test tool that is available on the market and that supports service-oriented testing is CANoe from Vector. This offers a uniform, cross-network communication concept. This means that CANoe provides a network-independent interface for the application models and test scripts. Actual access and transmission over a physical network are configured independently of this. This approach permits application-related modeling of service-oriented communication. Service interfaces are imported from AUTOSAR Adaptive descriptions and are modified and extended as required **(Figure 3)**. To this end, CANoe is based on the idea of a service toolbox **(Figure 4)**: Users are free to choose the service interfaces required for the current development status and the specific test objective.

The service interfaces, together with the associated methods and events, are available as modeling artifacts directly within CANoe. Users stimulate and visualize these directly using interactive controls. At the same time, they are also available via C#-based programming interfaces within test and simulation scripts.

The CANoe communication concept permits the dynamic creation of communication endpoints during the simulation. At the same time, it is possible to modulate the availability of a service. It is therefore possible to simulate dynamic topologies of the sort that occur in the connectivity field. To permit early validation, CANoe supports the concept of application-level simulations for which users do not have to configure any low-level protocol layers. For the development and testing of real ECUs, CANoe provides connections via Ethernet and SOME/IP.

Challenges in the Field of Measurement and Calibration

Another area that needs to be taken into account when considering Ethernet and AUTOSAR Adaptive is measuring and calibration. This does not relate to use cases based on the standardized XCP calibration protocol but to the use of service-oriented communication as a transmission protocol for measurement and calibration data. The use of point-to-point connections in the service-oriented architecture makes this approach radically different from solutions in which the measured data is read out via a bus topology. How can service-oriented communication be used for measurement and calibration? The SOME/IP protocol used in AUTOSAR Adaptive enables the acquisition and writing of data on the ECU side.

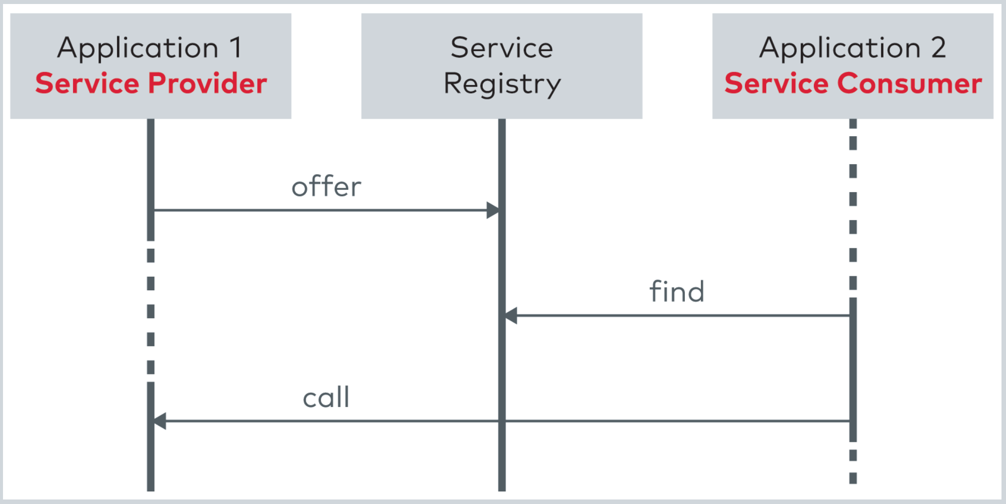


Figure 2: In service-oriented communication, an application provides a service that can be called by other applications.

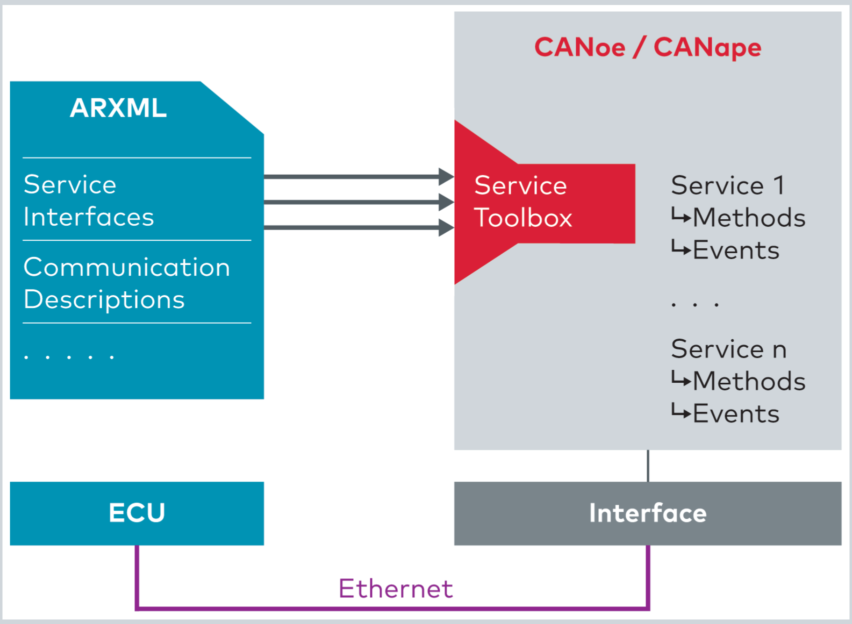
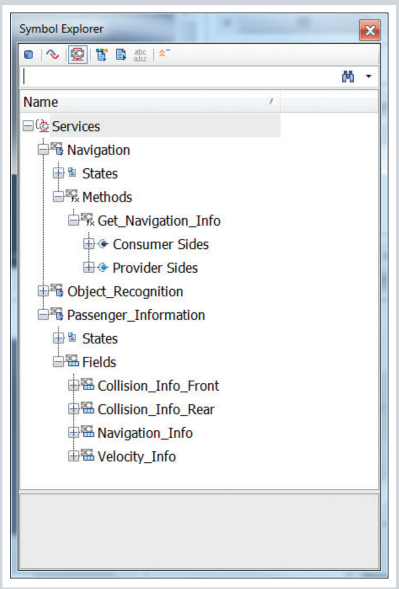


Figure 3: CANoe and CANape filter out the unneeded service interfaces using a type of service toolbox.

The acquisition of measurement data can be subdivided into two use cases: On the one hand, measurement data can be recorded simply by monitoring bus activity. In the case of Automotive Ethernet, this is achieved by interrupting the Ethernet connection and using a so-called network TAP (Terminal Access Point) to create the connection to the measuring tool. In a switched Ethernet network, two participants communicate via point-to-point connections. It is then possible to monitor the data exchanged between these two participants and extract and record it. On the other, data can also be acquired by executing a service in an application. In this case, the measuring tool behaves as if it is another consumer of the service's data. The advantage of this method is that it makes use of functionality that already exists in the ECU. Data serialization is implemented within SOME/IP and the description of the data is present in the service interfaces.


Symbol Explorer

Name

```
Services
  Navigation
    States
    Methods
      Get_Navigation_Info
        Consumer Sides
        Provider Sides
  Object_Recognition
  Passenger_Information
    States
    Fields
      Collision_Info_Front
      Collision_Info_Rear
      Navigation_Info
      Velocity_Info
```

Figure 4: The CANoe user chooses the service interfaces needed for the specific use case from the service toolbox.

This active pathway also makes it possible to write data to the application. Here again, the data that is to be transferred is described via the service interface. Thanks to a relatively simple extension, it is possible to add or extend a service during development and then use it for calibration.

However, this active communication method leads to an additional load on the system and the existing communication paths – just as when performing measurements using XCP on CAN. In the case of data-intensive measurement and calibration operations, it is therefore preferable to choose a solution that makes use of the ECU's debugging or trace interfaces, for example.

Measuring and Calibration Tools

Vector's CANape calibration tool is a good way of acquiring measurement data from service-oriented ECUs. It is based on the service interfaces and is able to import AUTOSAR-Adaptive descriptions **(Figure 3)**. In this case, CANape also imports the communication descriptions present in the files. This makes it easier for users to configure their measurement setups. CANape uses the service interface definition to interpret the measurement data and is able to save it in the usual measurement data format, such as MDF for example. As mentioned above, the data is acquired either by monitoring into communications between two participants or by actively requesting the data from the service providers **(Figure 5)**.

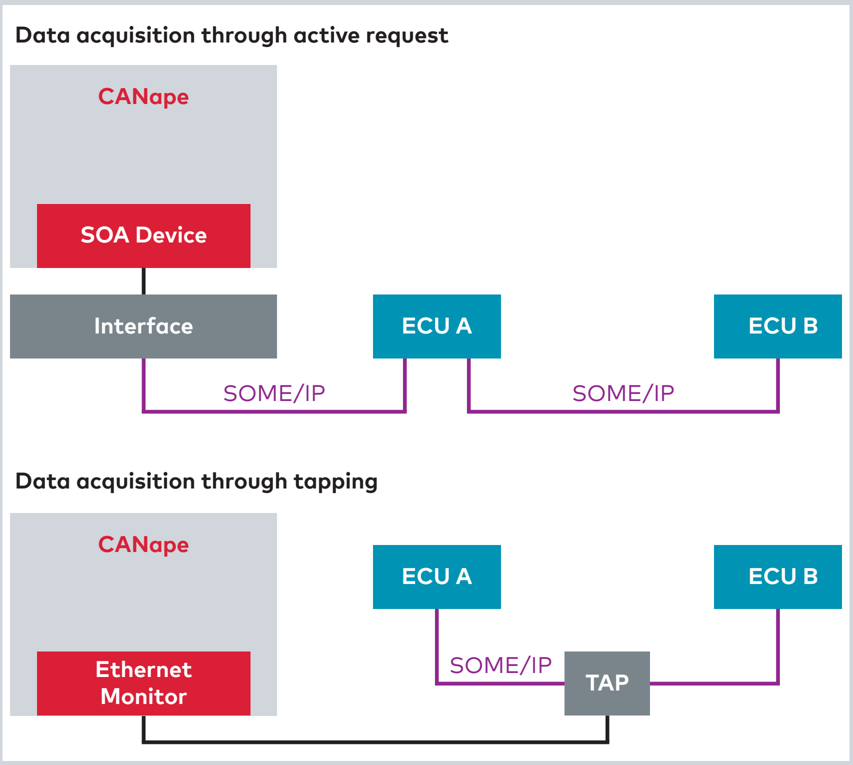
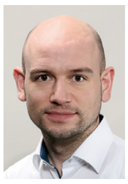


Figure 5: CANape records the measurement data either by actively requesting it from the service providers or by monitoring the communication between two participants.

Conclusion

AUTOSAR Adaptive transforms vehicles from being closed systems into systems that are networked with the environment. As a result, system architectures from the conventional IT world are increasingly being introduced into vehicles. Existing test concepts and tools must take account of this. First tools for the testing and calibration of service-oriented applications are already available. They enable development engineers to reduce the effort involved in constructing their test environments. In the future, the automotive industry will see the emergence of ever more tools that support the principle of service orientation.

Dr. Simon Frohn
studied Computer Science at the University of Stuttgart and received his PhD from the Chair of Computer Networks at the University of Leipzig. In 2012 he joined Vector and, as Senior Software Development Engineer in the SOA & Adaptive team, is responsible for the development of CANoe.



Fabian Rees
Fabian Rees studied Computer Engineering at the University of Constance and joined Vector in 2011. As a software manager, he is responsible for AUTOSAR Adaptive in the field of measurement and calibration.

Translation of a German publication in Elektronik automotive, Special Edition Automotive Ethernet, March 2018