Service Classification in Service-Oriented ICT Architectures of Future Vehicles

Timo Häckel
Hamburg University of Applied Sciences, Dept. Computer Science,
Berliner Tor 7
20099 Hamburg, Germany
Email: timo.haeckel@haw-hamburg.de

***Abstract*—Information and communication technology (ICT) is already a major driver in the automotive industry. Most of the new features and innovations over the past 10 years were enabled by ICT, still the usage of ICT in today's vehicles lacks behind it's possibilities. By introducing new service-oriented ICT architectures for future vehicles the industry tries to simplify the development and verification process as well as enabling Plug&Play capabilities in cars. As the current discussion shows, these new architectures will be built-on centralised computing units with high processing power using only one communication medium. Software components will be running as services inside and outside of the vehicular network.**

As this is a revolution in automotive software development, this paper gives an overview of the state of the art in research and industry regarding the design and deployment of novel ICT architectures. Based on this survey we extract the key attributes for service classification in the automotive domain and give an example on how to use this classification to control the communication between services, by arranging them accordingly.

I. Introduction

Information and communication technology (ICT) is already a major driver of the automotive industry. Thus Electrics, electronics and software in vehicles are essential for the competitiveness. Already in 2007 M. Broy determined that ICT contibutes up to 50% to the total value of a car and nearly 80% of inovations in the automotive sector were a direct product of the technology transfer from the domain of computer systems [1]. In respect to electric mobility ICT becomes the backbone of all relevant functions [2]. As most of the functions of today's cars are realised in software components, the demand for processing power and data bandwidth has increased drastically. At the same time the integration effort increases, as the impact of existing functions is harder to predict [3]. For this reason architectures and technologies for ICT in vehicles can no longer be viewed as evolutionary innovations, as this would result in a rise of complexity, which becomes to hard to manage. Instead the ICT architectures must be revised so far-sightedly, that they can perform their indisposable role in future cars [2].

The automotive industry recently introduced new standards and plans for future projects, realising a new ICT architecture [4]. The main concepts are a centralised communication medium, most likely real-time Ethernet and a service oriented software development. Each component will be realised as a service and therefore provide an interface for all data exchange and interactions. With the introduction of services to the automotive domain, new challenges and problems will rise. The new open and service based ICT architecture will need to control the communication between services with very different requirements. As in a centralised architecture the driving related services will no longer be separated from passenger related services, two worlds of software development will crash into each other. On the one side there are services with long development cycles and high demands on real-time and security, on the other side services with fast development cycles and lower real-time standards. Therefore, these services need to be arranged in an information architecture which will define communication flows and access restrictions.

This paper aims to define the key classification criteria for automotive services and designs an information architecture for service arrangements based on multiple of these criteria. It is structured as follows. In section II the state of the art in research and industry on creating a new service-oriented ICT architecture will be presented. At first the requirements of automotive ICT architectures will be illustrated. Afterwards we take a look at migration strategies of legacy systems to begin the deployment of new ICT architectures.Furthermore, one solution for a new and dynamic software platform for automotive ICT architectures, the AUTOSAR Adaptive Platform is described. At last the challenge of service arrangement and communication flows in novel service-oriented ICT architectures are shown. Section III will present the results of our research in defining criteria for service classifications. To gain insight in important attributes a service composition experiment was conducted. We describe a simple scenario, its realisation and the results. Based on the results, key attributes of automotive services will be described and we analyse different criteria for service classification. By designing a hierarchical service architecture based on classification criteria, we try to solve the problem of service arrangement in future ICT architectures. Section IV concludes the paper and explains lessons learned as well as describes future work.

II. Related Work

Several ongoing research and industry efforts are aiming at developing a new ICT architecture for vehicles. The following sections will describe the state of the art in research and the main concepts developed by different research teams in the field of creating a new automotive ICT architecture.

A. Requirements of Automotive ICT Architectures

Over the past 30 years there were many innovations in the automotive domain. Most of them were enabled by the introduction of information and communication technology (ICT) in form of electronics and software in vehicles . ICT has already a big impact on the market value of a modern car. [2] and [2] present critical aspects of the current vehicle software architecture and define requirements for future architectures. In detail:

1) Today's Architecture: With the growing need of functionalities and the high influence of such features on the market value, the architecture of hardware and software in today's vehicles was developed in an evolutionary way [3]. Existing systems were modified, extended and interconnected with new components to enable new features. This process leads to several problems, some mentioned below [2], [3]:

- *High hardware overhead:* With over 70 to 100 different interconnected electronic control units (ECU's), modern vehicles are wasting resources of micro controllers and networks.
- *Heterogeneous networks:* One car has various networks for different demands in communication (e.g. time-triggered, priority based, etc.) which makes the development and integration process even more challenging.
- *Increasing demand for interconnectivity and bandwidth:* Most safety functions in cars use data-fusion approaches to get a complete state of the environment. This increases the demand in bandwidth and quality-of-service.
- *Complex system verification:* Heterogeneous networks and black-box ECU's make the system verification process very complex. The integration of cross-domain functionality increases the testing costs.
- *Limited flexibility:* Today's ICT architecture is developed and tested in a static configuration which makes it hard to add components after-sale.

By introducing a new more dynamic ICT architecture in vehicles, these problems need to be solved. It is important to take into account that this will also introduce new challenges. For example the new architecture must be introduced without lowering the security requirements on automotive software.

2) Future Architecture: As figure 1 shows, there is an evident trend for architectures to become more complex than required, considering the evolutionary development of vehicle architectures and the complexity growth over time. Only a revision of the architecture and the use of new technology can bring the complexity down. [2] This could results in much smaller integration costs and an increasing innovation curve. As shown in figure 1, this process has already been observed in the past. For example in 1980 the rise of complexity lead to the introduction of micro controllers and new bus systems like CAN.

To solve the problems described in the Today's Architecture section, the papers [2], [3] suggest the use of the following main concepts for a hardware architecture:

- *Centralised computer architecture:* ICT Architecture based on centralised scalable computing units, which execute all hardware independent functions.
- *Highly integrated mechatronic components:* Sensors and actuators become smart components with standardised data interfaces.
- *Standardised communication backbone:* All bus systems in vehicles will be replaced by one, probably real-time Ethernet.

Although it is not the only communication medium, Ethernet is already in use in today's vehicles and integrated mechatronic components have already been introduced, e.g. in the BOSCH Chassis Systems Control [5].

For the software architecture there are some design principles to consider:

- *Data-centric paradigm:* In a data-centric system, each component describes the communication properties it supports and requires. This leads to a virtually decoupled system.
- *Extra-functional properties:* Operating systems and middleware technologies also need to take extra-functional properties like timing, fault tolerance and security guarantees into account.
- *Plug&Play capabilities:* Standardised plugs and sockets at the hardware level and protocols at software level must be defined to support the detection of new components and the exchange of information needed for the integration.
- *Resource awareness:* A Plug&Play mechanism needs information about available resources to be fully functional. Dedicated components need to manage resources for safety critical and non-safety critical functions to satisfy the components extra-functional properties.
- *Integrated services:* Base services need to be implemented at platform level offering frequently needed functionality, like abstraction of communication infrastructure, sensor and data fusion, and functional safety and security concepts.

Although the task to introduce a new ICT architecture in vehicles is challenging, the advantages gained by such a change will be worth the effort [3]. As new automotive manufacturers like Tesla have shown, the introduction of electric vehicles opens up the automotive market for new competitors. As a change in the drivetrain concept demands radical changes in the ICT architecture, electric mobility is a chance to introduce a new system architecture and new competitors will accept this challenge. In designing a new ICT architecture one has to make sure to offer migration and deployment strategies to the established OEM's, as they will need to adept to these changes to stay in the automotive market.

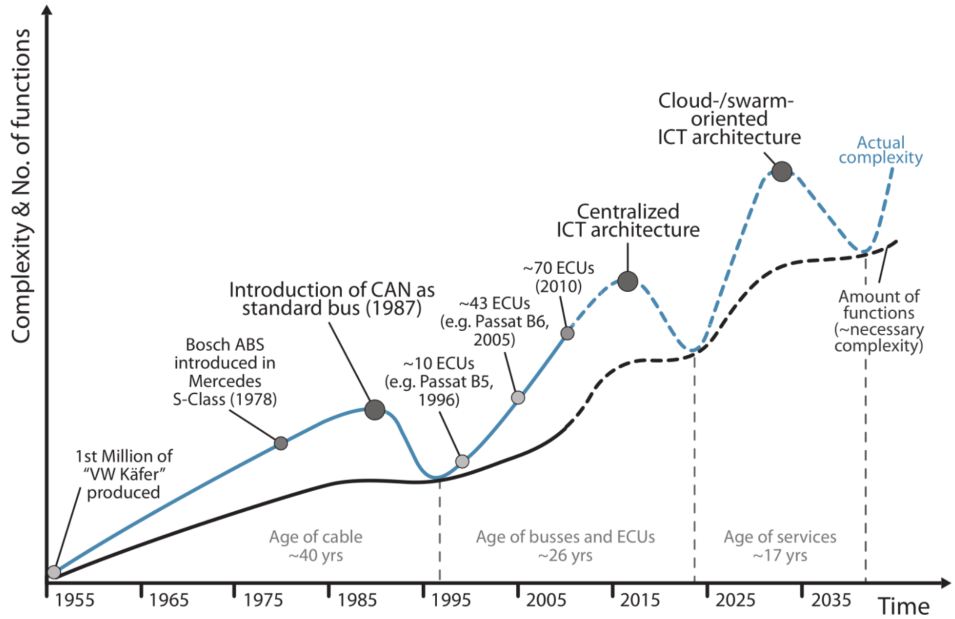

Fig. 1. Evolution of complexity in ICT architectures. [2]

B. Migration and Deployment

In the automotive industry legacy plays an important role [3]. Thus it is necessary for a new ICT architecture to enable the migration of well tested and proven components. This can be achieved by creating standards. As the main organization for standardisation of the automotive industry is AUTOSAR, they adopted the challenge of opening up there well-established system architecture for a new ICT architecture.

A research group of the Dept. of Computer Science of the SCT College of Engineering in Trivandrum India presents the challenges and advantages of a Service Oriented Architecture based connectivity of automotive ECUs and gives a summary of AUTOSAR standards and technologies solving these challenges [6].

Service Oriented Architecture (SOA) is an efficient and flexible way of interconnecting systems. It encapsulates the work performed by a system into a service, which can be accessed by a client using XML messages. SOA provides a loosely coupled way of interconnecting heterogeneous systems in efficient ways.

The use of Ethernet and demand for embedded Ethernet technology solutions in the automotive embedded system industry, paved the way for the possibility of using SOA in automotive embedded systems.

According to the paper [6], a feasibility study is the first and most important step to introduce SOA in automotive components. The authors name AUTOSAR as one important player that took the challenge and tried to integrate SOA in the AUTOSAR platform. By introducing SOME/IP (Scalable service-Oriented MiddlewarE over IP [7]) to the AUTOSAR platform in version 4.2.1 they created a serialization protocol with remote procedure call mechanism. This protocol solves the problems of data transportation (AUTOSAR std. 809, [8]), data serialisation for compatibility to all systems (AUTOSAR std. 637, [9]), service discovery in the network (AUTOSAR std. 616, [10]) and the transformation of ECU modules into services and methods (AUTOSAR std. 660, [11]). This architecture provides the capability of integrating SOA to the automotive embedded system without affecting the existing components.

With this integration of SOA and Ethernet, AUTOSAR has established a migration strategy for proven software components. Nevertheless, this system architecture doesn't solve the problems and challenges described in II-A. Therefore, a dynamic software platform is needed.

C. Dynamic Software Platform

Over the past 10 years AUTOSAR has been established as the organization and main driver for the standardisation for software infrastructure in the automotive industry. As a reaction on upcoming demands and new functionalities, AUTOSAR restructured it's portfolio. The introduction of the AUTOSAR Adaptive Platform is the result of adopting the new challenges of a dynamic software platform.

Members of the BMW Group, as a partner of the AUTOSAR organisation, name the key aspects of the new architecture and describe the structure and components of the AUTOSAR Adaptive Platform [12]. A summary is given below.

The initial version of the AUTOSAR Adaptive Platform was released in march 2017 [13].

1) Key Aspects: A new architecture has to consider the following two key aspects:

- *Integration of heterogeneous software platforms:* The networking architecture of today's cars can be clustered into different domains (e.g. infotainment, chasis or powertrain) which already use different computer and software architecture. While infotainment ECUs usually use Linux, the AUTOSAR Classic Platform is the standard for deeply embedded ECUs. The new use cases demand a third type of ECUs with a higher computing power. All of these three platforms need to be interconnected.
- *Service oriented and signal based communication:* The traditional communication is based on ECUs broadcasting signals to other ECUs. This fits well for control data of limited size, which has to be communicated cyclically. The new use cases, like highly automatic driving, have higher payload demands and Ethernet as a communication system requires more sophisticated protocols. With the concept of service oriented communication, applications provide a service on the communication system. Other applications can subscribe to these services and the data will only be sent to subscribers. The combination of the two types of communication is the second key aspect for a new ICT architecture.

2) Architecture and Components: Figure 2 shows the overall architecture of the AUTOSAR Adaptive Platform. The main goal is to provide a stable programming interface to application developers, called the AUTOSAR Runtime for Adaptive Applications. The interface consists a standard interface for accessing operating system functionalities as well as a communication middleware, which allows communicating with local and remote applications as well as the Adaptive AUTOSAR Services over SOME/IP [7]. Another goal is to smoothly integrate the platform into existing E/E architectures based on Ethernet. The Module Application Execution Manager is responsible for startup and shutdown of the ECU and the applications. It has to take care that the necessary ressources for the applications are available.

In future releases features to implement the new use cases should be provided. This includes support for fail-operational systems, enhanced safety and security features and the integration of Car2X communication.

D. Arrangement of Services

With the introduction of a novel dynamic and open ICT architecture a major challenge is the arrangement of services. As there are many different kinds of services communicating in the same network it is important to create a structure, defining who may communicate with whom and which messages might need prioritisation.

One solution for the arrangement of services was given by researchers at fortiss GmbH in Munich Germany. They presented several concepts for hierarchical information architectures in the automotive domain [14] and identified key research topics, as well as open issues for adopting these concepts in practice, with respect to proper layering and information exchange. There focus was on the technical aspects for design and development of hierarchical information architectures, disregarding most non-technical topics. They explained the important factors of a hierarchical information architecture regarding communication flow, layer traversal, meta-information and architecture design, as well as the upcoming research aspects. This is summarised below.

1) Hierarchical Information Architectures: The described hierarchical information architecture is shown in figure 3. They name the following requirements and conditions:

- *Communication flows:* They distinguish between three different kinds of communication flows:
  - *Vertical or inter-layer communication* refers to an information exchange between different architectural layers and should be permitted. Data transformation, fusion and arbitration should be used to bridge the semantic gap between different layers.
  - *Horizontal or intra-layer communication* refers to an information exchange between different subsystems inside an architectural layer and should be permitted. Otherwise it would be necessary to always move up in the hierarchy and down again, which is not only cumbersome, but could also result in increased latency and reduced performance.
  - *Diagonal or foreign sub-tree communication* refers to an information exchange between both different architecture layers and different subsystems and should be prohibited. Otherwise it would bypass the main idea of hierarchical architectures of encapsulating sub-trees of functionality from higher layers. This mechanism is very important and leads to higher level of abstraction.
- *Layer traversal:* There are two design choices regarding how to traverse (multiple) layers in a hierarchical architecture:
  - *Opaque layers* allowing components to access only they layers directly above or below. This leads to higher abstraction and better information hiding for safety and security aspects, but might also lead to higher latency.
  - *Transparent layers* allowing the access of either only lower layers or all layers of the system. This might be needed to reduce latency, but should be avoided if possible because it undermines the main reasons for hierarchical information architectures.
- *Meta-information and non-functional concerns:* In hierarchical and layered architectures one significant problem is the loss of information due to abstraction between different layers. This problem could be solved by appending meta-information to the data. On the other hand at run time there are also non-functional concerns such as latency or guaranteed response time which could also be added as meta-information.
- *Architecture design:* For the design and construction of the hierarchical information architecture one has to match two key aspects of automotive systems:

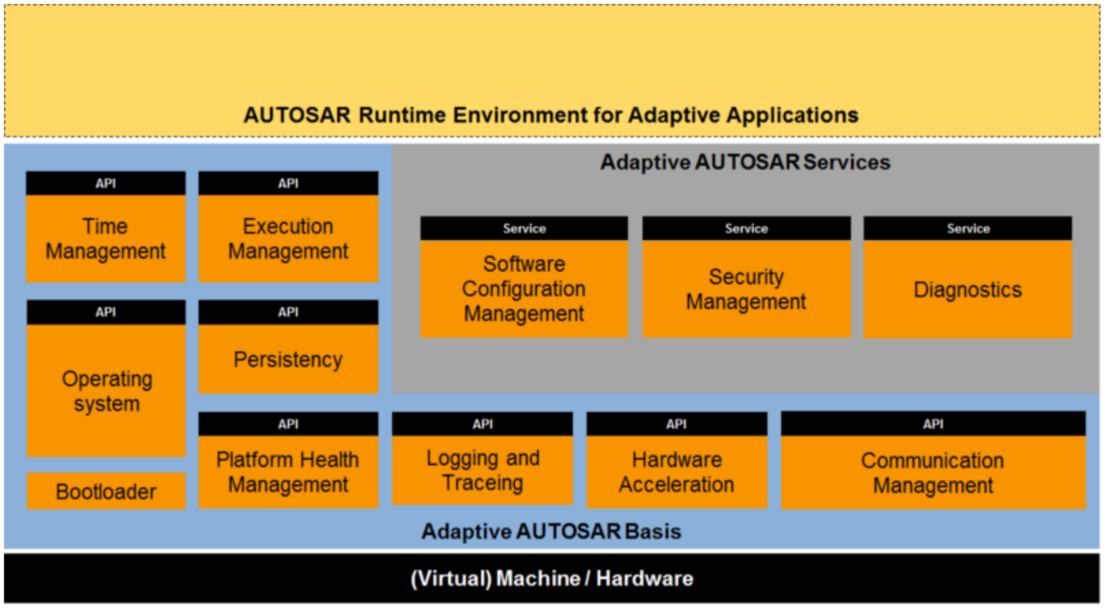

Fig. 2. Architecture of the AUTOSAR Adaptive Platform. [13]

- *Physical aspects* refer to either spatial/ connectivity based hierarchy categories, ordering functions based on location and connectivity or temporal hierarchy categories, ordering functions based on latency and cycle time.
- *Logical aspects* refer to either abstraction based hierarchy categories, ordering functions based on the level of abstraction they represent or relational hierarchy categories, ordering functions based on dependencies between components.

2) Research Challenges: Beside the described architecture the authors also name important research challenges and argue for an interdisciplinary approach, including cross-domain concerns and different viewpoints from research areas. As important areas they define[1] :

- Design conflicts
- Information exchange, data fusion and arbitration
- Hierarchical scheduling
- Safety and security
- Testing and verification

As described above for the architectural design there are many different criteria to arrange services accordingly in a hierarchical architecture. The question how to organize the services in the new structure stays open.

III. Classification of Automotive Services

Automotive software development standards are currently undergoing big transformations. In adopting the challenges on future ICT architecture in vehicles and introducing new standards, the automotive industry paves the way for service-oriented automotive software. The change from defined control loops and decoupled bus systems, to a centralised ICT architecture with only one communication medium and every software component realised as a service, is a revolution in the field of automotive software and will enable comfort in development as well as produce new problems and challenges.

[1]More detailed information can be found in [14].

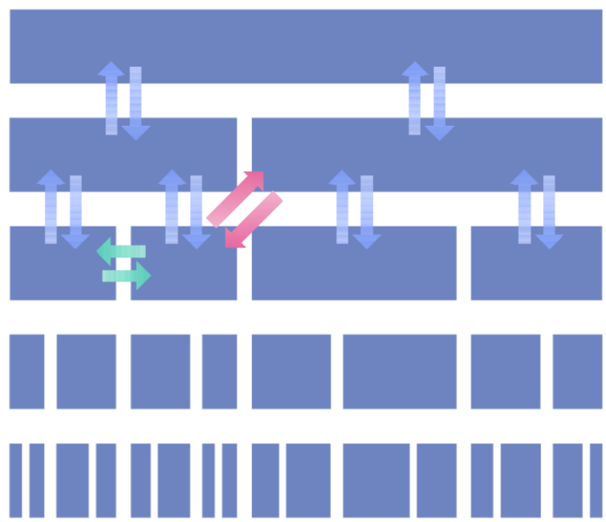

Fig. 3. An example of a hierarchical information architecture with layer separation. The architecture represents a tree of information and, by extension, functionality with fixed nesting. Information flows are only allowed inside a layer (horizontal), as well as to direct ancestors or descendants (vertical). Any access to a foreign sub-tree (diagonal) is forbidden in order to achieve proper encapsulation. [14]

Although many components might stay in encapsulated control loops, most components will be realised as services. Sensors will provide there data as a service and actuators will provide interfaces for controlling. These Automotive services need to comply the ”SPUR” requirements: Secure, Privacy preserving, Usable, and Reliable [15]. Accepting these requirements a service will be defined with an interface for the different access methods and interact with other services threw a middleware provided by the ICT architecture.

In such an open ICT architecture there will no longer be a strict separation of passenger and driving related services[2] any more. Every service could theoretically interfere with every other service. It is the task of a future service architecture to enable access restriction and encapsulation of subsystems as described in section II-D. Therefore meta-information of services is needed to add a level of control.

Section III-A describes an experiment of service compositions and interactions. The goal is to get an idea on important aspects and classification criteria for automotive services. Afterwards, section III-B adds on to this subject by defining criteria and attributes for service classifications. Based on these criteria, a design for a hierarchical information architecture based on multiple service classifications is presented in section III-C.

A. Service Composition Experiment

To gain insight in important service attributes and classification criteria, some example services were created. These services were composed to a scenario with the goal to experience interfaces and interactions of services. The following sections will describe the scenario it self, the realisation and the services that were build as well as the first results.

1) Scenario: The example scenario needed to represent different service attributes with different requirements. One important feature of future cars is the usage of cloud services and the provision of data to the cloud. So the location of the services should vary. Furthermore, the car will provide sensor data to cloud applications and will request services of the cloud as well. The scenario was chosen to get a first impression on service interactions, including vehicle internal and external services of different abstraction levels.

As a result, services for a very simple navigation scenario were implemented. The navigation should use the drivers personal calendar to determine the next destination. To refine route calculation, sensor data of the car like temperature and rain intensity is provided to cloud applications. This way the cloud can determine roads which might be slippery and avoid them in route calculation.

2) Realisation: For the implementation simple Spring[3] RESTful[4] Web Services were used to reduce the implementation time. As a result the following services were implemented:

- *Navigation Service:* With data structures including map images and route calculation. (connected with here API[5])
- *Calendar Service:* Request the upcoming events from a Google Calendar[6].
- *Navigation Controller Service:* Composing the Navigation for the user. The user can choose from locations of calendar items or decide to enter another location to calculate a route for and display it.
- *Rain and Temperature Sensor Services:* Providing dummy data to the cloud.
- *Data Collection Service:* Collecting rain and temperature data in a data base.

Afterwards the services were composed running on the same device as well as running on separate servers in a network.

3) Results: Although this is a very simple example scenario it showed some important aspects of service development. Through the usage of external services with documented interfaces like the Google Calender API or here APIs, the importance of type safety and API consistency were noticed. The better the documentation, the easier the service can be integrated into a new scenario. On the other hand the impact of the location regarding latency due to the distance between the services could be recognized. This must be represented in the arrangement of services in a novel architecture as well. Another aspect could be determined by having multiple services accessing one at the same time. The decision of prioritisation and who to trust becomes important. What if one input says go left and the other one says go right? Therefore services need to be categorized and classified.

B. Service Classification

By classifying services with attributes one can extract the differences in requirements of automotive services. This enables future services as well as the service platform for better decision making, for example regarding access rights and meeting execution deadlines. On the other hand, these classification attributes could be used for the arrangement of services in (hierarchical / layered) information architectures. Below multiple approaches for service classification and different attributes are examined.

1) Service Domain: When trying to define requirements and attributes of services, the service domain is a good starting point, as it has a big influence. Typically automotive software will be categorised in the following domains [15]:

- *Multimedia:* Refers to services regarding entertainment as well as navigation systems.
- *Passenger / Comfort:* Refers to services regarding the passengers comfort, like climate control unit and personalisation of the car.
- *Safety electronics:* Refers to services of safety critical ECUs, like the anti-lock braking system.
- *Engine / drivetrain:* Refers to services regarding the engine and drivetrain, like automatic transmission or battery management in E-cars.
- *Diagnostics:* Refers to services surveying the operation of the car. This software needs to recognise malfunctioning

[2]Passenger related services are all services regarding the passenger including for example entertainment and environment control. Driving related services on the other hand are all services regarding the vehicles drivetrain.

[3]Spring is a framework to create and access WebServices. It is implemented in the JAVA programming language. For more information: https://spring.io/

[4]Representational state transfer (REST) or RESTful Web services are one way of providing interoperability between computer systems on the Internet.

[5]Here provides different APIs for route calculation and maps: https://developer.here.com/develop/rest-apis

[6]Google Calendar provides an API to access calendar data: https://developers.google.com/google-apps/calendar/

components and communicate them to the driver and the repair shop.

Although these domains will incorporate most of the new services, when talking about future ICT architectures there will be services that might not fit into one of these domains . Therefore, a more generous domain should be introduced.

- *Integrated services:* Refers to services which are provided by the platform to help the developers. This includes operating system functions, as well as functions provided by the ICT architecture environment. Examples may be the abstraction of communication, service discovery, automated updating, and safety /security management. As future ICT architectures will be mostly structured for data communication and processing (described in section II-A and II-D) there will be additional services regarding this domain, like data collection, data fusion, data processing and data base services.

Looking at this shows the variety of domains for automotive services and gives a slight insight on differences in requirements of such services.

2) Service Location: Another key aspect to find requirements and attributes of a service is where it is located in the service network. This has a big impact on response time and safety requirements. The following three locations will be differentiated:

- *Internal vehicular network:* Refers to services which are located in the internal vehicular network, running on small ECUs, multi-core ECUs and multimedia systems. In a future ICT architecture these service will most likely be connected via Ethernet and can communicate without a high latency due to round trip times and transfer rates.
- *Cloud servers:* Refers to services which are located outside the vehicle and running on Cloud servers accessed via Internet. These services might have longer response times due to latency.
- *Local ad hoc Networks:* Refers to services which are located outside the vehicle and are connected in a local wireless ad hoc network. This includes IoT devices on streets and traffic lights, services running in other cars, as well as on smart phones. These networks are slower than the internal vehicular network and might have a higher latency. Nevertheless, the connection will still be much faster than to Cloud servers in the Internet.

The location of a service is obviously a key aspect in service interactions as it has a big impact on quality-of-service aspects, like communication latency. On the other hand, the usage of external services opens up the new topic of security. Different questions surface, like who has access to in car services open and who to trust.

3) Service Trust Status: In open ICT architectures with software of different providers running on the same component as well as in the same network, the access rights need to be controlled. A new information architecture needs to take care of the trust status of a service. To solve this issue it is important to introduce a trust hierarchy. There are different mechanisms to consider in trust hierarchies:

- *Trust certificates:* A service can be certified by a certification authority. In trust hierarchies there is always one authority that has the trust of all entities. This central authority can either give directly certify every service or like in a chain of trust, create certificates for other authorities below it and creates a chain.
- *Trust levels:* Another option is to only trust services for a certain level. For example this could be based on the domain, or on the abstraction level. One solution is a trust pyramid with different access levels. This way services may have access to other services based on there security level.

Integrating trust mechanisms into automotive services could help in solving many security problems created by the open ICT architecture.

4) Service Access Method: Looking at examples of existing service architectures in the World Wide Web one can identify different access methods on how to interact with a service. The different access methods might be secure or insecure. There are two categories of access methods:

- *Read only:* As read only service access methods do not change any data at the corresponding service. They are considered to be secure in this way. This access method could be compared with standard get/head requests realised in Web Services.
- *Read / Write:* As read / write service access methods do change the data at the corresponding service. They are considered being insecure in this way. This access method could be compared with standard set/put/post and delete requests in Web Services.

By defining service access methods, the ICT architecture can be labelled much saver. Certain services might get read permissions, others read/write and some may not be allowed to access services at all.

Another method of service interaction that will be used and needs to be supported by a novel ICT architecture is the publish / subscribe mechanism. As described in section II-C1 the service communication needs to enable cyclic data broadcasting, which can be enabled by publish /subscribe patterns.

5) Service Attributes: Thinking about service classifications there is not one single aspect to look at. It's not enough to say a service is driving related or passenger related. It's not enough to name the domain or location of a service. Instead a service based information architecture for future vehicles needs to provide the possibility to attach meta-information to methods and services as well as messages and data-types (as described in II-D). To provide meta-information for services and interfaces one has to define the key attributes of an automotive service. There are different viewpoints for such meta-information, e.g. a client needs different meta information of a service than the execution environment. The following attributes need to be taken into account.

- Identifier, Name, Version
- Domain
- Location
- Class / abstraction level
- Access Methods
- Vendor Id
- Certification
- Availability
- Execution time (guaranteed)
- Execution cycle
- Scope / Access limitations
- Life cycle / update cycle
- Deadline
- Safety / Security requirements

C. Hierarchical Service Architecture Based on Classification

After gaining some insight in attributes and classification criteria for services, the next step is trying to find an architecture for the arrangement of automotive services. Starting with the hierarchical layered information architecture described in II-D, there are still many open questions. We tried answering some of them, with an enhancement of the presented architecture regarding the design conflict of the arrangement paradigm, the information exchange between services and a basic approach for safety and security.

We adopted the main concepts of the hierarchical architecture for communication flows, to achieve a well-structured and organised architecture with encapsulating sub-trees. For better information hiding the opaque layer design was chosen for the layer traversal. On the other hand as shown above, the classification of automotive services is a way to complex subject to have a flat architecture with only one type of classification. Figure 4 shows the designed approach for a hierarchical information architecture based on service classification.

Each automotive service domain has its own sub-tree in the hierarchy. The services are arranged according to there abstraction level, from low level abstraction at the bottom to high level abstraction at the top. Furthermore, there are three basic service locations visualized: In-Car, Cloud and IoT. Due to latency, connections can be marked as slow, normal or fast. The fast connections are express lanes, to skip multiple layers and access a service on a different level, without passing high latency routes. To incorporate the trust status into this architecture we decided to introduce trust guards. These will only allow services with a high enough trust level to access the service interface. For safety aspects it is also possible to only allow communication in one direction.

With the introduction of multiple classification criteria and attributes, the arrangement of services should be much easier to accomplish. Another important feature is the introduction of slow and fast lanes into the communication, as now applications with tight deadlines have the opportunity to get a faster data flow. On the other hand the introduction of trust guards is only a starting point, in giving the architecture a safety and security aspect. More research will be needed in this regard.

IV. Conclusion

This paper gave an overview of the state of the art in research and industry on ICT architectures in future vehicles. It described the trends and open research aspects as well as named some changes that are already applied in the AUTOSAR Adaptive Platform. With the introduction of services, the new challenge of service arrangement is described. Based on these aspects of new ICT architectures and service arrangements, we define the most important service attributes and classification criteria. Based on multiple criteria, we designed a hierarchical information architecture to accomplish encapsulation of sub-systems and define the communication flows.

A. Lessons Learned

After the experience to put the service interactions and interfaces into practice there are some lessons learned to share.

The first aspect is the complexity of API development and the importance of documentation. After using a couple different very complex APIs with many data types, it is possible to say that documentation is key for every user. It took several days to understand the thoughts that went into data structuring with the Here and Google Calendar API.

Another experience was made in the field of data exchange. By integrating the car into the Internet of Things it has become a sensor node for future web services. One important factor that came to mind was who is the owner and who may use the data a car produces. For the car owner another aspect might be privacy, as the data might be connected to the driver and consumers might get a deep insight in the drivers life.

B. Future Work

With the introduction of a hierarchical information architecture for services based on multiple classification criteria we introduced one way of handling the arrangement of services in future vehicles. The next step is to refine this architecture based on experience in practice. Therefore, a prototype will be developed.

References

[1] M. Broy, I. H. Kruger, A. Pretschner, and C. Salzmann, "Engineering automotive software," *Proceedings of the IEEE*, vol. 95, no. 2, pp. 356–373, Feb 2007.

[2] C. Buckl, A. Camek, G. Kainz, C. Simon, L. Mercep, H. Stähle, and A. Knoll, "The software car: Building ict architectures for future electric vehicles," in *2012 IEEE International Electric Vehicle Conference*, March 2012, pp. 1–8.

[3] H. Stähle, L. Mercep, A. Knoll, and G. Spiegelberg, "Towards the deployment of a centralized ict architecture in the automotive domain," in *2013 2nd Mediterranean Conference on Embedded Computing (MECO)*, June 2013, pp. 66–69.

[4] *Explanations of Adaptive Platform Design*, Online, AUTOSAR Std. 706, Rev. AUTOSAR AP Release 17-03, 2016, access: 2017-05-23. [Online]. Available: https://www.autosar.org/fileadmin/files/standards/adaptive/17-03/general/specs/AUTOSAR_EXP_PlatformDesign.pdf

[5] R. B. GmbH. (2012, nov) Chassis systems control ab12 new generation of airbag control units. [Online]. Available: http://life.bosch.com.cn/ebrochures2015/automated/cc/ps/ab12/ab_12_en.pdf

[6] G. L. Gopu, K. V. Kavitha, and J. Joy, "Service oriented architecture based connectivity of automotive ecus," in *2016 International Conference on Circuit, Power and Computing Technologies (ICCPCT)*, March 2016, pp. 1–4.

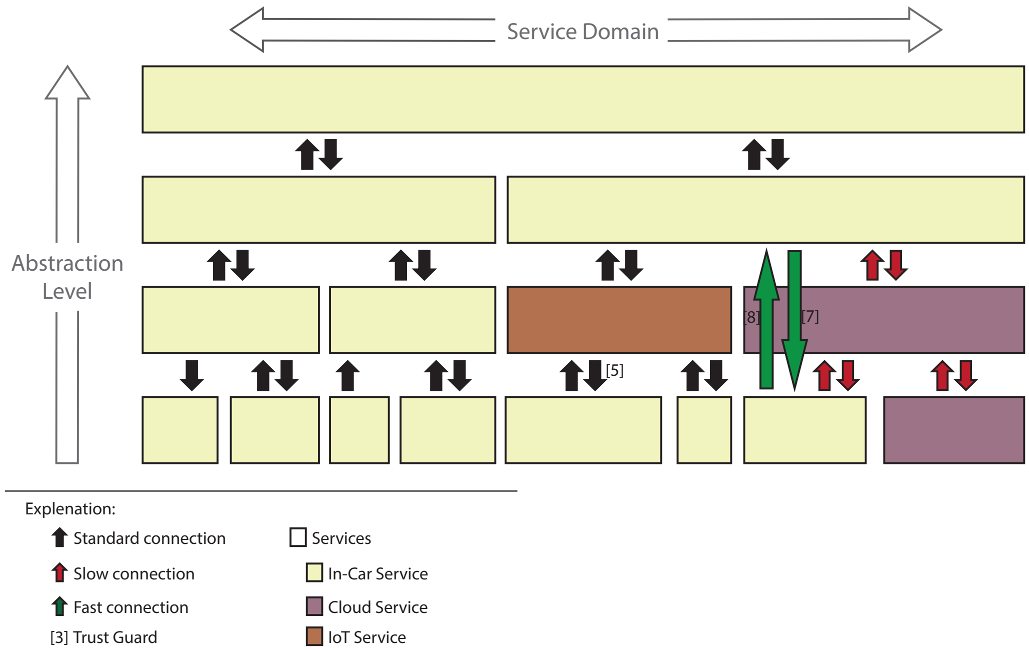

Fig. 4. Hierarchical information architecture, with a service arrangement based on multiple classification criteria.

[7] D. L. Völker, "Some/ip die middleware für ethernetbasierte kommunikation," *HANSER automotive Networks/2013*, pp. 17–19, 2013, german.

[8] *Specification of Module SOME/IP Transport Protocol*, Online, AUTOSAR Std. 809, Rev. AUTOSAR CP Release 4.3.0, 2016, access: 2017-03-07. [Online]. Available: http://www.autosar.org/fileadmin/files/standards/classic/4-3/software-architecture/communication-stack/standard/AUTOSAR_SWS_SOMEIPTransformer.pdf

[9] *Example for a Serialization Protocol (SOME/IP)*, Online, AUTOSAR Std. 637, Rev. AUTOSAR Release 4.2.1, 2014, access: 2017-03-07. [Online]. Available: http://some-ip.com/papers/cache/AUTOSAR_TR_SomeIpExample_4.2.1.pdf

[10] *Specification of Service Discovery*, Online, AUTOSAR Std. 616, Rev. AUTOSAR CP Release 4.3.0, 2016, access: 2017-03-07. [Online]. Available: http://www.autosar.org/fileadmin/files/standards/classic/4-3/software-architecture/communication-stack/standard/AUTOSAR_SWS_ServiceDiscovery.pdf

[11] *Specification of SOME/IP Transformer*, Online, AUTOSAR Std. 660, Rev. AUTOSAR CP Release 4.3.0, 2016, access: 2017-03-07. [Online]. Available: http://www.autosar.org/fileadmin/files/standards/classic/4-3/software-architecture/communication-stack/standard/AUTOSAR_SWS_SOMEIPTransformer.pdf

[12] S. Frst and M. Bechter, "Autosar for connected and autonomous vehicles: The autosar adaptive platform," in *2016 46th Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshop (DSN-W)*, June 2016, pp. 215–217.

[13] AUTOSAR. (2017, mar) Adaptive platform : Autosar. [Online]. Available: https://www.autosar.org/standards/adaptive-platform/

[14] M. E. Jobst and C. Prehofer, "Towards hierarchical information architectures in automotive systems," in *2016 3rd International Workshop on Emerging Ideas and Trends in Engineering of Cyber-Physical Systems (EITEC)*, April 2016, pp. 41–46.

[15] M. Broy, *Model-Driven Development of Reliable Automotive Services - Second Automotive Software Workshop, ASWSD 2006, San Diego, CA, USA, March 15-17, 2006, Revised Selected Papers*, 2008th ed. Berlin Heidelberg: Springer Science Business Media, 2008.