Context Prediction Architectures in Next Generation of Intelligent Cars

Conference Paper · November 2018

DOI: 10.1109/ITSC.2018.8569617

CITATIONS
0

READS
143

6 authors, including:

Sina Shafaei
Technische Universität München
18 PUBLICATIONS **33** CITATIONS
SEE PROFILE

Morteza Hashemi Farzaneh
Technische Universität München
11 PUBLICATIONS **52** CITATIONS
SEE PROFILE

Stefan Kugele
Technische Hochschule Ingolstadt
60 PUBLICATIONS **206** CITATIONS
SEE PROFILE

Context Prediction Architectures in Next Generation of Intelligent Cars

Sina Shafaei[1], Fabian Müller[2], Tim Salzmann[3], Morteza Hashemi Farzaneh[4], Stefan Kugele[5] and Alois Knoll[6]

***Abstract*— The growing number of intelligent components inside a car leads to a considerable increase in amount of the produced data. Context aware paradigm plays a major role in managing this data and offering a numerous number of prospects and advantages for existing and new intelligent applications inside the car. Following that, enabling context prediction promises reliable solutions in terms of enhancing the comfort of the occupants and vehicle dynamics. Moreover, this would be a great step toward facilitating highly automated and autonomous driving. However, due to the complex nature of the data resources in an intelligent car and also the lack of comprehensive studies on different aspects of this concept in automotive, defining a functional architecture for context prediction requires broad knowledge and better understanding of multiple domains which are involved and have impacts. In this paper, we investigate the most effective elements and factors in each one of the related domains which help to enable context prediction architectures inside the intelligent cars and analyze the feasible dimensions in detail, cover their advantages, and address the challenges ahead. We elucidate the possibility and validity of our considerations with the help of two use cases of adaptive HVAC and ACC systems.**

I. Introduction

The biggest benefit of context-aware applications becomes visible in mobile sensor-rich devices. Smartphones and wearables became extremely adaptive to the environment they are used in and are therefore improving the interaction with the user. However, the set of sensors and the desired functionality of such devices is limited and well defined. A car by definition, is a mobile device, too. Compared to smartphones it is equipped with an even richer set of sensors and aims to fulfill many kinds of functionality including highly complex functions. Dey et al. defines the *"context"* as *"Any information that can be used to characterize the situation of an entity"* [1]. Due to the high amount of sensory data, it is not straight forward to define which sensor input should form the context. There are already some approaches on bringing the context-awareness in automotive software architecture. The context data can be stored on a central server [2] or on a layered architecture which relies on services offered by a context provider [3].

[1]Sina Shafaei is with Department of Informatics, Technical University of Munich, Germany sina.shafaei@tum.de

[2]Fabian Müller is with Department of Informatics, Technical University of Munich, Germany Fa.Mueller@tum.de

[3]Tim Salzmann is with Department of Informatics, Technical University of Munich, Germany Tim.Salzmann@tum.de

[3]Morteza Hashemi Farzaneh is with Department of Informatics, Technical University of Munich, Germany morteza.hashemi@tum.de

[3]Stefan Kugele is with Department of Informatics, Technical University of Munich, Germany stefan.kugele@tum.de

[3]Alois Knoll is with Department of Informatics, Technical University of Munich, Germany knoll@in.tum.de

Modern context-aware applications, in addition to using context to improve the user experience, also predict the future context. This empowers a huge number of new applications and use cases. Furthermore, as the future context for a functionality is predicted, the application can anticipate by adjusting how it interacts with the user or automatically remodels its functionality. In an automotive context-aware and context prediction architecture this enables a wide range of new driver comfort features and extended driving functionality. For example the seat and in-cabin temperature could be adjusted before the driver even enters the car. Moreover, a context-aware architecture can considerably increase energy efficiency of the systems [4]. By enabling the context prediction in car architectures, this could even be improved further which is a great advantage in modern electric vehicles with limited energy capacity. However, research in automotive domain regarding the integration of context prediction architectures has only recently begun. To be able to make progress in this domain it is important to understand and address the challenges and effective parameters first, therefore we consider two leading use cases in an intelligent car in the following for this purpose.

A. Heating, Ventilation and Air Conditioning (HVAC) System

HVAC is a set of functions to fulfill the aim of improving the comfort of the driver and passengers inside the cabin. While every modern car provides some kind of HVAC system, their context-awareness is very limited at the moment and do not employ context prediction. Much of the functionality of an HVAC system is very straight forward and results in simple actions taken by the system, therefore, HVAC is one of the simple, yet promising start points to bring context-awareness and demonstrate the possibility of enabling context prediction inside the car. Our leading example for an HVAC system is the comfort functionality where the inner space temperature of the car is automatically adjusted to ensure the welfare of the occupants. The effectiveness of this service can be detected by the emotion recognition systems which are exclusively designed to monitor the emotional status of the driver and passengers.

B. Adaptive Cruise Control (ACC) System

ACC describes a function for semi and highly automated driving. It enables the car to automatically stay on a lane with a preset velocity or follow a car on the lane if it goes slower than the preset velocity. In contrast to HVAC it only provides a single function to the user. The underlying functionalities feeds ACC and the output, desired trajectory, is genuinely more complex than the output of HVAC functions. The

leading functionality of ACC we consider here is the *lane detection* which is an important sub-function of ACC to keep the car on the lane.

We start with an intuitive definition for context prediction in automotive in Section II. In Section III this paradigm is examined from the software engineering perspective. We further evaluate prediction dimensions, methods and their dependencies in Section IV. The impact of cloud-based services beside the Car-to-X concept on context prediction structures is analyzed further in V. Section VI, respectively covers the related considerations in current architectures of automotive software for runtime environments. In Section VII we cover the concerns and address the challenges on ensuring the safe manner of intelligent components that are employing prediction methods. In the last Section VIII we go through the communication infrastructure and considerations which are necessary to be taken into account and discuss upon the state-of-the-art developments in this domain on the way of satisfying the newly raised requirements.

II. Context Prediction

The ambiguity of the term "context prediction" itself, in this domain is the first thing which needs to be addressed. For context-awareness we stick to the definition of Dey et al. as *"A system is context-aware if it uses context to provide relevant information and/or services to the user, where relevancy depends on the user's task"* [1]. For general applications we define context prediction as the process of inferring future context information from past and current context. As modern cars have a steadily increasing amount of sensors and other information sources (see Section V) it is not always easy to define which part of the overall available information describes the context of the desired functionality. In ACC, it is hard for a human to define the context features beforehand. Relevant information might not be included as it does not seem relevant for the developer. Therefore, another application of prediction methods is predicting the relevant features (the context) from the overall available information for a function. For the field of automotive software architecture, we redefine the term "context prediction" by splitting it to:

- Context feature prediction
- Context time-series prediction

and use to differentiate between them and to explain the challenges using each methodology individually or connected in an automotive software architecture. As the concept of context-aware computing and context prediction has a large influence on all components within automotive software structure, the main task is to define an architecture where the individual components can interact together effectively. Starting from traditional automotive software architectures, existing components could be merged or removed and new components might need to be considered. Also introducing context-awareness and prediction together in a certain way into a component of the architecture may affect how these concepts have impact on other components. The main goals here are first, understanding the possibilities and challenges for the introduction of context prediction into individual components and then investigating the effects of a component which employs context prediction on all other components. The context processing can either happen on application layer with integration of the new components which provide intelligent services or be applied on runtime environment and possibly forwards the result toward the applications.

III. Software Engineering

Defining Context, *Storing and Sharing Context*, *Robustness*, and *User Experience* are among the most important challenges of context-aware computation which are depicted by Dey et al. [1] and Winograd [5], and can be categorized under the context prediction challenges as well. Most of them have already been solved for mobile context-aware applications. However, bringing context-awareness and prediction to an automotive environment reopens all again.

Depending on which dimensions of context prediction are used and which way of introducing context processing into the architecture is selected (Section II), the structure of the software for functions and systems inside the car must adapt substantially compared to traditional automotive software development. This is necessary in order to maintain the performance while satisfying the requirements for the software systems in cars, like the specific context for operation of the systems [6]. The simplest approach is only using context time-series prediction with a fixed prediction horizon. Here the software functionality have a fixed input feature set with fixed prediction steps. Therefore, traditional software patterns perform well. However, as soon as the input is determined dynamically (e.g. online context feature prediction) the software can become very complex and quickly reach the limits of traditional rule-based software patterns.

Most of the context prediction applications today, have software which is strictly rule based, programmed specifically for a single application. This works well for non-complex functionality and fixed input dimensions. However, if the functionality itself is of such complexity that it cannot be programmed rule-based, machine learning could be used to infer an output (or action) of the function given its current (including past and future) context. We call this *context output* or *action inference* interchangeably. In special cases this can be seen as another context prediction dimension. For instance, when the context is used to predict what the desired action of the driver/passenger will be. Therefore, it is required to investigate generally applicable softwares or model-based patterns which can be used dynamically for varying input dimensions and any complex functionality. Traditional software engineering in automotive will considerably change with integration of context prediction due to the increasing number of the new technologies which are meant to bring intelligence into the cars (respectively the advancements in hardware) while these changes must still maintain and fulfill the concerns w.r.t. requirements engineering, compatibility and software integration [6]. Functionality will be produced

by models instead of rule based software. Our concern therefore, is to bring existing software architectures and machine learning together and investigate models (like reinforcement learning) which can be universally used merely by adjusting the model parameters. This enlightens the importance of the concerns we describe here along with the ones represented in Section IV.

Another key challenge for enabling the context-awareness and prediction, is the software and system integration. In the automotive industry this is done using the Autosar (Automotive Open System Architecture) as a runtime environment. Although the Autosar provides a powerful architecture to develop automotive software components independent from specific base-hardware and operating system [7] but the lack of enough flexibility in its coding guidelines leaves open challenges regarding the probabilistic models. Therefore, it is important to focus on how next generation of context-aware softwares can be designed so it can be integrated into embedded hardware and employ the new technologies like service-oriented computing in automotive software architecture [8] to preserve not only the independency from hardware or operating system but also supporting the capabilities for run-time adaptations. Alternatively different possibilities of runtime environments need to be evaluated for their capabilities and enforcement in software which we cover in Section VI.

IV. Prediction and Inference Methods

There are widely used methods for prediction and inference in different domains of research related to context prediction. A thorough comparison of them is represented in [9]. The *sequence prediction* approach is one of the quite researched ones in theoretical computer science. D. Cook et al. [10] provides a comprehensive overview of sequence prediction techniques with focus on smart homes. Another approach is based on *Markov chains* which are formal models and some projects applied them to address context prediction problems like [11] in which the authors addressed an active device resolution problem or the work presented at [12] where the authors used discrete time Markov chains to predict the driving route. *Dynamic Bayesian Networks* (DBN) is a generalization of Markov models while avoiding some of the Markov model's shortcomings which got used in numerous projects like the user modeling and user goals inference at [13] and respectively predicting the person's indoor movement at [14] by representing the context as DBN where the current room depends on several rooms visited previously. Here the duration of staying depends on the current room, time of the day and also, day of the week.

The neural networks in machine learning are gaining more and more success in solving a variety of problems like pattern association, recognition, and function approximation, hence seem promising enough to be used in the context prediction core of the architectures. One of the earliest works is [15] where the authors described a smart house that predicts expected occupancy patterns in the house, estimates hot water usage and the likelihood for a zone to be entered. Similar use cases based on neural networks for context prediction can be found in [16]–[18]. Nearly all of the aforementioned prediction and inference methods, are employed for specific use cases therefore, it is obvious that there is lack of a general prediction method which performs well equally in all of the use cases.

In order to select a method for the context prediction core in future automotive architectures there are a set of concerns which need to be taken into account. The *knowledge inference* factor must be identified beforehand since some of the state-of-the-art prediction approaches, like neural networks, do not consider the prior knowledge inference. Moreover, the prediction core must provide a *reliability estimation* while maintaining the *readability* for the developer in order to have the option of validating the correctness and verifying the safety (see Section VII) of the predicted context or taken decisions. This indicates also the importance of the *observability* factor w.r.t. being a white box or black box. In most of the cases the *information loss* in the preprocessing step is unavoidable, yet this loss shall not affect the important relevant features in a way that the predicted context falls into the undesired region. Besides all of this, the remaining concern is the mutual dependency between the predicted context and system actions which is still an unsolved issue. However, the works on Markov Decision Processes (MDP) address this issue. MDPs are a plausible and practically effective way to predict the context in situations when Markov models are applicable and control actions can significantly affect prediction results. We also address a set of considerations regarding the prediction core which we do believe are of importance in next generation of intelligent cars due to the complex nature of the generated data. These concerns are categorized into *Feature Prediction*, *Time-Series Prediction*, *Output/Action Inference*, and *Online Learning* which are explained in details in the following.

A. Feature Prediction

The feature prediction for a function, focuses on finding correlations between the feature subsets (the existing information) and the output. If such correlation exists the information is likely to be part of the function's context. So far this has not been a focus of context-aware architectures. However, other fields of study have already started research in this area. In marketing for example, finding correlations in big data sets (e.g. buying habits) is of great interest. Here, some tools like association rule learning or market basket analysis aim to find correlations in data. Regarding this issue an evaluation of the existing models from other fields and checking the feasibility of integrating them into this domain is beneficial. It's worth mentioning that correlation does not automatically deduces a causal connection.

B. Time-Series Prediction

For time-series prediction, research has already been started in non-automotive context prediction architectures. Sigg [19] and Rosa [20] both give overviews of different

machine learning models and the corresponding feasibility and integration for context time-series prediction. For a general model it is important to be able to set or automatically vary the prediction horizon. For ACC use case, only the past and next seconds are really important. Similarly in HVAC use case it is of interest to predict the context for the next minutes or even hours. A practical example is when the planned navigation route is over a mountain pass. As the temperature is expected to drop, the heating should be increased. For this aim an inspection into the existing models that are used in traditional prediction architectures and adapting or extending them is necessary.

C. Output/Action Inference

Finding a relatively generic model which describes the output/action inference within a context prediction-based architecture in car is a challenging task. Therein multiple factors play a role: first of all depending on the functionality, the complexity of the output might be high. In our ACC use case the output is a structure describing the lanes on the road. The output is much more complex when there is more than one single output value like it was in HVAC (e.g. desired temperature). Also the input dimension might vary (dynamic context features) and the data representation in those dimensions can vary too (normalization, discrete/continuous, etc.). Therefore, it is hard to find a supervised model which is generic enough to serve all purposes. Using reinforcement learning, however, enables a shift of complexity from the model itself towards finding an action policy and a suitable reward function. Here the model and learning algorithm are the same for all use cases. Just the policy and reward function needs to be defined for each use case separately and desirably define them dynamically and online.

D. Online Learning

For a plug and play architecture it is mandatory to learn the necessary context features online. If the context features are learned online, the other context related parts have to be highly adaptive and learn online as well since their input dimensions are exposed to change. In this case, most of the desired context-aware functionality has to be available in the car at the very beginning of the use or at least be learned very fast. Especially for the context output prediction it is important to quickly learn the desired output. For the use case of HVAC the algorithm has to acquire the knowledge about the desired temperature of an individual after a short period of time and quickly adapt to changes. Otherwise it is not an enrichment and will not be accepted by the driver. This is not feasible with pure online learning. Hence a thorough research into possibilities to combine offline and online information for the training of models is needed like the work of Ye et al. [21] in representing an online planning approach with regularization that got included in an autonomous driving system for real-time control of the vehicle which itself was originated by the proposals of Gelly et al. [22] on "algorithms that combine the general knowledge accumulated by an offline reinforcement learning algorithm, with the local knowledge found online".

E. End-to-End Learning

Having said that all context related components support different aims, they also are highly dependent on each other. If the context features are determined online, the subsequent components must be adaptive. This raises the question of whether independent models for each component are reasonable. Working towards end-to-end learning could be an alternative. Even a single model for all functionality which receives all information as input and all expected output/actions as output can be possible. For HVAC functionality with its multiple actions of limited complexity this is a reasonable approach to develop. In contrast, for the use cases like ACC with a highly complex output and underlying structures this does not seem to be a promising approach. On the one hand, finding good models for end-to-end learning is hard and often a matter of experience. On the other hand it is still reasonable to use a reinforcement learning approach to shift the focus to finding a good policy instead. An end-to-end learning approach has the advantage that interfaces between components are dynamic by definition.

V. Cloud services and Car-to-X

Concepts of cloud-based services and Car-to-X (c2x) information sharing, are now growing fast in automotive domain. These bring many opportunities to context-aware and context prediction-based applications. New context can become available to cars without the cars having to sense it themselves. This might be the exchange of trajectories with other vehicles (Car-to-Car (c2c) communication) for the functions involved in automated driving or information about the current status of the environment around the vehicle, like traffic lights. Using that information, a context prediction structure can better understand and predict more accurately due to the quickly changing environment. For the ACC lane detection use case, c2c communication could bring great improvement. If a car in front has a better set of sensors then we could use the sensor data of that car to predict the context and be generally context-aware of the lanes in ego vehicle. Especially for prediction, using data from cars which are ahead of us, can bring big performance improvements. However, this requires the context features to be predicted online. Main challenges here, come from the communication itself. As Wagner et al. pointed out, ensuring the compatibility interfaces for c2x is a major challenge [23]. Despite these challenges the inclusion of cloud services and c2x communication besides investigation on existing communication approaches, which expand the sensory information of the car, can be used as context and to predict the future context. However with all the benefits that cloud services or c2c and c2x brings to the driving, there is a raising concern on ensuring the security of the network. Avoiding malicious data injection into the intelligent components by an attacker especially on the services which rely on online learning and detecting the safety violations by the provided

information (e.g. trajectories) either by the environment or other vehicle are important.

VI. Runtime Environment

In a traditional automotive software architecture the runtime environment is a middleware which connects software components, functionality, and hardware. A middleware controlling all functional components opens the possibility to move context-awareness and context prediction into the runtime environment and then provide the context to functional components. This is already covered in mobile devices in a way that the responsible middleware detects if the phone is tilted, predicts if it will be in the near future and provides this information to the relevant application like camera. Even enforcement of context-awareness is possible. In the mobile example, the middleware can adjust the resolution (e.g. from 720x1080 to 1080x720) to enforce the application to change its displays according to the context. Paspallis et al. [24] uses a middleware for collecting context accordingly.

Multiple projects have been developed to create a dynamic runtime environment capable of the features needed to enable context prediction. Sommer et al. [25] created a platform called RACE which is capable of *Plug & Play* and works greatly together with today's concept of highly sensorised cars and other communication concepts. Depending on which sensors and which other software is installed in the car, software component can change its behavior. For the HVAC use case this means that if an additional sensor is installed (e.g. facial fatigue detection or emotion recognition of passengers) this new information is used by the functional software to adjust the heating. We should also consider this fact that enabling plug & play might require a standardization of interfaces for the components of the architecture. We can investigate this issue from two perspective of *Sensor* and *Function Monitoring*

A. Sensor Monitoring

One possible functionality besides the plain communication between components could be detection of sensor failures which, nowadays, is done in a fail-silent manner. In this case, the function gets stopped and the driver gets informed over a display and then respectively an error memory entry is set up. If the context and more data sources are added dynamically via Car-to-X or cloud services this becomes more complex. While the challenges are substantial this brings huge advantages, too. For example a detected failure of the front radar of the ego vehicle could result in the runtime environment replacing the relevant context of that sensor with the sensory data provided by the preceding vehicle. As sensors are not reliable w.r.t. being entirely fail-safe, the faulty or missing data automatically gets replaced by a different data source like the work represented at König et al. [26] which made first steps including such assumptions into context prediction. Alternatively if the runtime environment detects a sensor which is disturbed but still delivers context, it is challenging to filter the errors because all the sensors in that case need to be checked regularly. In *Highly Automated Driving (HAD)* this is categorized under the quality issues. For example, the performance loss of the cameras has to be detected regularly in case of situations like muddy camera. The quality of data and therefore the quality of the context, besides the correctness, is highly decisive towards the success of a context prediction automotive architecture. Therefore, behavior of the middleware which can evaluate and improve the current status of sensors and their delivered data can be a potential interest for the future researches. This also have some impacts on improving the general safety aspects which got discussed in Section VII.

B. Function Monitoring

Similar to sensor monitoring, monitoring the correct functionality of the components by runtime environment is a desired feature. Especially with complex functionality components, using probabilistic models, a check if output is still generated by the component as well as a plausibility check on the output, improves the stability of the system. A study which was discussed by Buckl et al. [27] proposes changes in context prediction architectures to open the possibility of testing single functions for failures and errors. Thereby the architecture must be the least complex possible even with an increasing number of components.

VII. Safety

In a domain-agnostic model, Varshney et al. [28] defines safety as the minimization of risk and uncertainty. In other words, it is the absence of failures and dangerous situations. From this perspective, safety is an essential consideration when it comes to an automotive software architecture.

Currently the functional safety concerns of automotive software are addressed by the ISO 26262 [29], which is an international standard established with the aim of ensuring that all components have been designed and developed with rigor to ensure safety by minimizing random and systematic failures. However, when it comes to knowledge inference and machine learning-based solutions in context prediction, the traditional verification and validation methods do not perform very well. Software methodologies and tools to establish safety assurance in safety-critical systems (e.g. [30]) have been around for a long time, and in fact represent quite a mature field. The trouble, however, is that these have been developed for traditional software systems, i.e. the ones which have been explicitly programmed to work in a certain way, with clearly defined requirements.

If we take the case of neural networks as one of the promising approaches for context prediction (see Section IV), the main challenges from the perspective of safety can be enumerated as *black box structure*, *implicit specification*, and lack of acceptable *coverage-based testing* [31].

Splitting the data into *train*, *validation*, and *test* data is one of the methods that we are very familiar with even to date, and one that is extensively used to ensure the developed adaptive system works well for a given set of inputs. This method helps to verify the functioning of neural network, but is not usually extensive enough to be considered as

a guaranteed approach in ensuring the safety in context-based high-critical systems [32]. One of the main reasons for lack of trust in the train-validation-test split method, is that one is only left with a small set (normally around 20%) of the samples to test against, wherein the chances are that cases of high interest or an unpredicted context might not even be tested. A suitable way to overcome this problem, is to use test data generation tools that can generate synthetic data points which can be used for testing the trained neural networks. This approach was found to aid the verification procedure for neural networks according to their correct behavior by unveiling missing knowledge a.k.a context, in fixed neural networks and increasing confidence in the working of adaptive neural networks [32].

Rule extraction algorithms can, thus, be used to model the knowledge that a neural network has acquired during the training phase. These rules can be generated in a conjunctive or subset selection form. The rules extracted thereby can either be manually verified owing to the human readable format, or can be automated with a model checker. This method can be helpful to establish trust in the system, as it augments the explainability of the system. It also aids traceability of the requirements, as one can verify if the rules depict functional requirements specified for the system. They can also help examining the various functional modes of the system, and to ensure that safe operation mode is induced by certain inputs. Though this method brings along a lot of advantages, it can be a good solution for offline learning systems, where in the verification and validation, system designer can extract rules from the network after training is complete. As it was described in Section IV, the online feature learning is a must for plug & play architectures and in that case, it becomes an added overhead for the system since the rules need to be extracted after every iteration to ensure that it is learning as expected. This can be expensive in terms of computation and time. However, there are solutions like online monitoring which is a technique that uses multiple monitors working together as an oracle to provide information about the functioning of the neural network in order to aid stability and convergence analysis [33]. The goal here is to ensure that the adaptation dynamics does not cause the network to diverge, triggering behavior unpredictably. Data sniffing [34] is an example based on the foregoing technique, which studies the data entering and exiting a neural network. If a certain input could pose negative results, then the monitors generate an alert and could even possibly flag down the data, thereby not allowing it to enter the system. This method is extremely useful in cases where outliers could degrade the functioning of the system.

The RACE project [25] uses a centralized platform computer to achieve a safety-critical execution environment with interfaces for verifying and testing the components. An alternative is to implement a separate controlling instance. The ACC is context-aware and context prediction-enabled and uses a probabilistic model to determine its actions which needs to be approved by a formally verifiable component. Only if the action is approved by the controlling instance then it is executed. Thereby the advantages of context prediction are combined with the formally verified safety of the controlling instance.

VIII. Communication

The newly generated data and the formed context, following the Increase in number of applications in an intelligent car, arises challenging requirements in the lowest level of the context prediction architectures for in-vehicle communication systems. For instance, it has to provide mixed-criticality to support different data traffic priorities, including traffic with hard real-time requirements. Such a communication system is supposed to provide higher bandwidth (e.g. required for best-effort video data transmission $> 1GB/s$) and timing guarantees to fulfill safety-related requirements (e.g. deterministic minimum latency and jitter, reliability, and fail-operational). Considering the mentioned requirements, Institute of Electrical and Electronics Engineers (IEEE) decided to extend Ethernet standards to develop real-time *Audio Video Bridging (AVB)* [35] technologies. These standards are significant for automotive and automation application domains. AVB working group defines a set of standards to provide timing and synchronization guaranties. Its objective is to guarantee a worst-case latency of $2ms$ for critical and real-time traffic class A and worst-case latency $50ms$ for traffic Class B over maximum of seven hops. To control the latency values, Credit-Based Shaper (CBS), Stream Reservation Protocol (SRP) and timing and synchronization have been specified in *IEEE 802.1Qav*, *IEEE 802.1Qat*, and *IEEE 802.1AS*. Using the CBS mechanism, network bursts (transmission of multiple frames directly behind each other) are managed to fulfill timing requirements.

AVB standards have been further extended to reduce the worst-case latency and jitter down to microsecond range and to support redundancy mechanisms. These additional features are significant for safety-relevant applications such as fail-operational. The Time-Sensitive Networking (TSN) [36] task group deals with developing the required standards towards a unified hard real-time Ethernet technology.

Considering the new standards, *IEEE 802.1 Qbv Time-Aware Shaper (TAS)* defines a time-triggered mechanism to use gate drivers to prioritize data forwarding of the switch ports. Hence, experienced latency of high-priority and critical Ethernet frames in a switch is minimized. This requires reservation of enough bandwidth for such critical context using *IEEE 802.1Qca Path Control and Reservation* and *IEEE 802.1Qcc Stream Reservation Protocol (SRP)* enhancements and performance improvements. It is also important to support a frame preemption mechanism (*IEEE 802.1Qbu*) to interrupt low-priority frame transmission to the advantage of high-priority frames. Precise gate timing requires clock synchronization which is supported by *IEEE 802.1AS-Revision* standard (Timing and Synchronization for Time-Sensitive Applications in Bridged Local Area Networks). TSN proposes *IEEE 802.1CB Frame Replication and Elimination* standard to support seamless redundancy for application which are supposed to be fail-operational.

Despite all advantages of TSN which makes it a suitable candidate for the future context prediction architectures, there are some challenges which have to be considered towards a successful integration. Theses are among others, security at MAC layer and high network configuration and verification overhead. From the perspective of prediction core of the architectures which are based on machine learning approaches (like the ones explained in Section IV), the correctness of the learned context is highly critical. However, there are no available solutions for security and anomaly detection in TSN networks at the moment. This is a significant motivation to follow the standardization activities in *IEEE 802.1Qci* and to develop security solutions and introduce them to this working group. Now, there are no competing solutions, which deal with the TSN hardware components. However, this will change in the next couple of years, when these components are stabilized in automation and automotive domains. One of the main challenges of Ethernet (MAC layer) is that security aspects have never been a significant consideration in its design. Its architecture focuses more on low-cost and easily deployable Local Area Networks (LAN). This challenge gets even bigger by considering TSN (which also acts at MAC layer) in safety-critical domains and applications like acc.

Supporting plug & play features in TSN networks is a further challenge which requires tools supporting modeling [37], [38] and automated schedule synthesis [39]. The existing tools consider only the timing aspects but not the reliability features or the impact of wireless communication on the network performance and safety. Especially, current activities in the working group *IEEE 802.1CM* w.r.t. 5G development, are highly relevant for network management when this comes to the cloud-based solutions and integration of Car-to-X concepts as it was mentioned in Section V.

IX. Conclusion

In this work we expressed the impact of context prediction in cars which implies many opportunities for improvement in the functions not only in the current car architectures but more importantly in the next generation of intelligent cars on the way of reaching to full autonomy. As we pointed out throughout this work, understanding different aspects of the involved domains, beside addressing their challenges is a crucial matter in defining an efficiently functioning architecture for context prediction in order to improve the different aspects of automotive mobility.

Although, there are multiple dimensions in which context-awareness and context prediction architectures could be defined and become operational, we only focused on comfort and Intelligence with a simple comfort function (HVAC) and a relatively complex intelligent one (ACC), and with the help of those two, we did demonstrate the advantages and applicability of deploying such structures. We adapted the definition of context prediction to fit better into the automotive domain and did investigate the main concerns on the way of developing context prediction architectures from the perspective of *software engineering* by addressing the challenges that traditional methods face and presented possible solutions for them. We covered the promising approaches of *prediction and inference methods*, considering the advancements in domain of machine learning beside the challenges that are of importance w.r.t. *safety* of such systems. *Cloud-based services & car-to-x* integration by recent developments in the domain of 5G and cloud technologies into the car structure was examined beside their effects and requirements on context-awareness and context prediction paradigm. Current status and the future vision of the *runtime environments* in software architecture of the cars was investigated from two perspective of *sensor* and *function* monitoring and at the end, we proposed the most suitable approaches and listed the challenges in car *communication* infrastructure in order to satisfy the newly raised requirements. Having said all these points, we believe that enabling context prediction architectures for the next generation of intelligent cars, is a necessary and indeed a beneficial prerequisite to improve performance and increase the acceptance of highly automated cars.

X. Acknowledgement

This work was supported by BMW Group. We thank our project partner for putting our research into an innovative practical context.

References

[1] G. D. Abowd, A. K. Dey, P. J. Brown, N. Davies, M. Smith, and P. Steggles, "Towards a better understanding of context and context-awareness," in *International Symposium on Handheld and Ubiquitous Computing*. Springer, 1999, pp. 304–307.

[2] G. Rigoll, "The connecteddrive context server–flexible software architecture for a context aware vehicle," *Advanced Microsystems for Automotive Applications 2007*, p. 201, 2007.

[3] M. H. Tran, A. Colman, and J. Han, "Service-based development of context-aware automotive telematics systems," in *Engineering of Complex Computer Systems (ICECCS), 2010 15th IEEE International Conference on*. IEEE, 2010, pp. 53–62.

[4] S. Loughran, "Towards an adaptive and context aware laptop," *HP LABORATORIES TECHNICAL REPORT HPL*, no. 158, 2001.

[5] T. Winograd, "Architectures for context," *Human-Computer Interaction*, vol. 16, no. 2, pp. 401–419, 2001.

[6] M. Broy, "Challenges in automotive software engineering," in *Proceedings of the 28th international conference on Software engineering*. ACM, 2006, pp. 33–42.

[7] G. Park, D. Ku, S. Lee, W.-J. Won, and W. Jung, "Test methods of the autosar application software components," in *ICCAS-SICE, 2009*. IEEE, 2009, pp. 2601–2606.

[8] M. Wagner, D. Zobel, and A. Meroth, "Towards runtime adaptation in autosar: adding service-orientation to automotive software architecture," in *Emerging Technology and Factory Automation (ETFA), 2014 IEEE*. IEEE, 2014, pp. 1–7.

[9] A. Boytsov, "Context reasoning, context prediction and proactive adaptation in pervasive computing systems," Ph.D. dissertation, Luleå tekniska universitet, 2011.

[10] D. J. Cook, "Prediction algorithms for smart environments," *Smart environments: Technologies, protocols, and applications*, pp. 175–192, 2005.

[11] K. Kaowthumrong, J. Lebsack, and R. Han, "Automated selection of the active device in interactive multi-device smart spaces," in *Workshop at UbiComp*, vol. 2. Citeseer, 2002.

[12] J. Krumm, "A markov model for driver turn prediction," 2016.

[13] D. W. Albrecht, I. Zukerman, and A. E. Nicholson, "Bayesian models for keyhole plan recognition in an adventure game," *User modeling and user-adapted interaction*, vol. 8, no. 1-2, pp. 5–47, 1998.

[14] J. Petzold, A. Pietzowski, F. Bagci, W. Trumler, and T. Ungerer, "Prediction of indoor movements using bayesian networks," in *International Symposium on Location-and Context-Awareness*. Springer, 2005, pp. 211–222.

[15] M. C. Mozer, "The neural network house: An environment hat adapts to its inhabitants," in *Proc. AAAI Spring Symp. Intelligent Environments*, vol. 58, 1998.

[16] A. Gellert and L. Vintan, "Person movement prediction using hidden markov models," *Studies in Informatics and control*, vol. 15, no. 1, p. 17, 2006.

[17] E. Al-Masri and Q. H. Mahmoud, "A context-aware mobile service discovery and selection mechanism using artificial neural networks," in *Proceedings of the 8th international conference on Electronic commerce: The new e-commerce: innovations for conquering current barriers, obstacles and limitations to conducting successful business on the internet*. ACM, 2006, pp. 594–598.

[18] W. Lin, W. Wu, and Q. Zhang, "Handover strategy of smart mobile terminals among heterogeneous wireless networks," in *Proceedings of the 2008 International Conference on Advanced Infocomm Technology*. ACM, 2008, p. 30.

[19] S. Sigg, *Development of a novel context prediction algorithm and analysis of context prediction schemes*. kassel university press GmbH, 2008.

[20] J. H. da Rosa, J. L. Barbosa, and G. D. Ribeiro, "Oracon: An adaptive model for context prediction," *Expert Systems with Applications*, vol. 45, pp. 56–70, 2016.

[21] N. Ye, A. Somani, D. Hsu, and W. S. Lee, "Despot: Online pomdp planning with regularization," *Journal of Artificial Intelligence Research*, vol. 58, pp. 231–266, 2017.

[22] S. Gelly and D. Silver, "Combining online and offline knowledge in uct," in *Proceedings of the 24th international conference on Machine learning*. ACM, 2007, pp. 273–280.

[23] M. Wagner, A. Meroth, and D. Zöbel, "Developing self-adaptive automotive systems," *Design Automation for Embedded Systems*, vol. 18, no. 3-4, pp. 199–221, 2014.

[24] N. Paspallis and S. E. Alshaal, "Improving qoe via context prediction: A case study of using wifi radiomaps to predict network disconnection," in *Proceedings of the 8th ACM/SPEC on International Conference on Performance Engineering Companion*. ACM, 2017, pp. 31–34.

[25] S. Sommer, A. Camek, K. Becker, C. Buckl, A. Zirkler, L. Fiege, M. Armbruster, G. Spiegelberg, and A. Knoll, "Race: A centralized platform computer based architecture for automotive applications," in *Electric Vehicle Conference (IEVC), 2013 IEEE International*. IEEE, 2013, pp. 1–6.

[26] I. König, B. N. Klein, and K. David, "On the stability of context prediction," in *Proceedings of the 2013 ACM conference on Pervasive and ubiquitous computing adjunct publication*. ACM, 2013, pp. 471–480.

[27] 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 *Electric Vehicle Conference (IEVC), 2012 IEEE International*. IEEE, 2012, pp. 1–8.

[28] R. Salay, R. Queiroz, and K. Czarnecki, "An analysis of iso 26262: Using machine learning safely in automotive software," *arXiv preprint arXiv:1709.02435*, 2017.

[29] ISO, "Road vehicles–Functional safety (ISO 26262)," 2011.

[30] A. Wassyng and M. Lawford, "Software tools for safety-critical software development," *International Journal on Software Tools for Technology Transfer*, vol. 8, no. 4-5, pp. 337–354, 2006.

[31] C.-H. Cheng, F. Diehl, Y. Hamza, G. Hinz, G. Nührenberg, M. Rickert, H. Ruess, and M. Troung-Le, "Neural networks for safety-critical applications-challenges, experiments and perspectives," *arXiv preprint arXiv:1709.00911*, 2017.

[32] B. J. Taylor, M. A. Darrah, and C. D. Moats, "Verification and validation of neural networks: a sampling of research in progress," in *Intelligent Computing: Theory and Applications*, vol. 5103. International Society for Optics and Photonics, 2003, pp. 8–17.

[33] B. J. Taylor, *Methods and procedures for the verification and validation of artificial neural networks*. Springer Science & Business Media, 2006.

[34] Y. Liu, T. Menzies, and B. Cukic, "Data sniffing-monitoring of machine learning for online adaptive systems," in *Tools with Artificial Intelligence, 2002.(ICTAI 2002). Proceedings. 14th IEEE International Conference on*. IEEE, 2002, pp. 16–21.

[35] "Audio video bridging," http://www.ieee802.org/, accessed 10.04.2017.

[36] "Time-sensitive networking," http://www.ieee802.org/, accessed 10.04.2017.

[37] M. H. Farzaneh, S. Shafaei, and A. Knoll, "Formally verifiable modeling of in-vehicle time-sensitive networks (tsn) based on logic programming," in *Vehicular Networking Conference (VNC), 2016 IEEE*. IEEE, 2016, pp. 1–4.

[38] M. H. Farzaneh and A. Knoll, "An ontology-based plug-and-play approach for in-vehicle time-sensitive networking (tsn)," in *Information Technology, Electronics and Mobile Communication Conference (IEMCON), 2016 IEEE 7th Annual*. IEEE, 2016, pp. 1–8.

[39] M. H. Farzaneh, S. Kugele, and A. Knoll, "A graphical modeling tool supporting automated schedule synthesis for time-sensitive networking," in *Emerging Technologies and Factory Automation (ETFA), 2017 22nd IEEE International Conference on*. IEEE, 2017, pp. 1–8.