Smart Cabin Application on High Performance Computing Machine - AC Personalisation

Gowrishankar S[1], Kuldeep Kumar[2], Sreevishakh KP[3], Muralidhara K V[4], Shruthe R[5]

[1]Body Practice, KPIT Technologies Limited, Bengaluru, Karnataka, India
Email: *gowrishankar.S[at]kpit.com*

[2]Body Practice, KPIT Technologies Limited, Bengaluru, Karnataka, India
Email: kuldeep.kumarr[at]kpit.com

[3]Body Practice, KPIT Technologies Limited, Bengaluru, Karnataka, India
Email: *sreevishakh.P[at]kpit.com*

[4]Body Practice, KPIT Technologies Limited, Bengaluru, Karnataka, India
Email: muralidhara.kv[at]kpit.com

[5]Body Practice, KPIT Technologies Limited, Bengaluru, Karnataka, India
Email: *shruthe.r[at]kpit.com*

Abstract: ***Heating Ventilation and Air Conditioning (HVAC) systems have been used for many years to create comfortable temperature environments in enclosed spaces such as residential, industrial, space, and automotive industry. Vehicle HVAC systems work to keep passengers at a comfortable temperature. However, a variety of environmental elements have an impact on thermal comfort, and a person's thermal preferences can vary substantially due to physiological, behavioral, and cultural factors. In this paper we came up with AC personalization application prototype on high performance computing along with machine learning support to predict preferred ac temperature values based on previous learning data sets and three real time input values. These three inputs include ambient temperature (cabin temperature), heart rate and body temperature of the driver. A combination of classic and adaptive Autosar hardware platforms are used for the prototype. A Service-Oriented Real-Time Communication (SOA) scheme over SOME/ip is used for communication between classic and adaptive nodes also we used ANDi tool for showcasing the outputs rather than HVAC module. We used a combination of three machine learning algorithms-Decision Tree Regressor, Polynomial Regressor, Adaboost Regressor for ac temperature prediction model.***

Keywords: Adaptive & Classic Autosar, SOME/IP, Service Oriented Architecture (SOA), Machine learning, RCar-H3e, AURIX™ TC37x.

1. Introduction

Electric vehicles and autonomous driving are two trends that will have an enduring impact on the future of the automotive industry**.** Both the nature of cars and consumer expectations are evolving. Connectivity to the outside world Vehicle-to-vehicle (V2V), Vehicle-to-everything (V2X) has become an essential feature for all vehicles not only for the high-end ones. Customers demand flexible, connected, and scalable functionality, and OEMs are altering their products and services to meet these demands. Automotive industry is moving towards SOA (Service Oriented Architecture) and promoting SDV (Software defined vehicles) into reality [1]. The capacity to use artificial intelligence (AI) and Machine learning has increased potential for innovative development, providing more hands-free, cutting-edge travel experience and comfort to the occupants and also advanced safety features [2]. Machine learning paradigms will serve as the sole foundation for future autonomous vehicles' ability to plan, detect, steer, navigate, and make critical judgements in traffic environments. To accurately capture the functional requirements of intelligently driven autonomous cars, it became necessary to create standardized frameworks and models. An overview of existing architecture is provided in paper [3] along with recommendations for future developments. Physiological sensor fusion (ECG, Heart rate etc.) along with ML is extensively used nowadays to for monitoring of driver state (drowsiness and health conditions) [4][5].

In the field of vehicle HVAC, a lot of research has been done with the goal of improving occupant comfort and energy consumption efficiency. Machine learning techniques are introduced in HVAC systems to ensure thermal comfort to the passengers. In [6] implemented and evaluated performance of different machine techniques - MLR, MLP, MARS, RBF, REPTree, KNN, and RF for predicting equivalent AC temperature considering following cabin environmental parameters: temperature and relative humidity at six points (head, chest, and feet level of the occupants), solar loading at the driver sunroof, cabin air and surface temperatures, driver's central and outboard face vent air temperatures.

This paper introduces a smart cabin application over SOA and HPC that allows drivers to personalize their air conditioning temperature settings automatically based on Machine learning without driver intervention based on their physiological data, environmental data, and vehicle data. The application will analyze these data to learn the driver's initial or preferred settings and store them as part of their cabin temperature profile.Figure.1 represents the ac personalization application.

Figure 1: Smart cabin- AC personalisation application

Every time the driver enters the vehicle, the AC temperature must be adjusted and updated based on changes in the weather and sometimes depend on the driver's health conditions also. Figure 2 shows different biological parameters and impacted automotive functional areas.

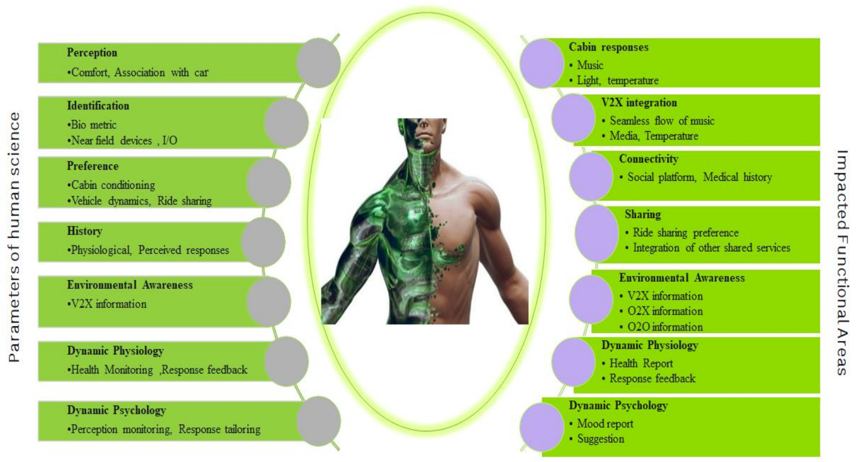


Figure 2: Biological parameters and impacted automotive functional areas

This paper is structured as follows: Section 2 covers adaptive Autosar and service-oriented architectures, section 3 presents the experiments conducted for selecting appropriate machine learning algorithms while Section 4 presents the H/W implementation of application with results followed by conclusion and future scope.

2. Adaptive Autosar and Service Oriented Architecture

Adaptive Autosar is an extension of the basic classic Autosar standard with trend of software-driven electric and electronic (E/E) vehicle architecture into the automotive Industry. The standard Autosar specifications, now called classic Autosar, are bounded for developing applications for specific ECU. Automotive systems are focused on trends like autonomous driving, predictive maintenance, v2x connectivity, OTA updates, and vehicle electrification. New functionalities have introduced high complexity and processing needs, requiring high computing power, faster communications, and continuous updates. The technology that enables these new functionalities includes ethernet and High-Performance Computing (HPC).

Adaptive AUTOSAR focuses on separating hardware from software which also results in multiple compute clusters or ECUs combining distributed software functionality for different functions in one High performance computing machine forming the "vehicle server or central processing unit".

There is a fundamental shift in vehicle software architecture to accommodate high computational requirements. The standard conventional vehicle network is domain based with one central gateway forming the backbone of the complete vehicle network. We have nearly 100 ECUs with dedicated functions, mainly communicating using signal-based mechanisms. These ECUs are mainly classic Autosar based which communicate over a CAN, LIN, FlexRay and few communicating over Ethernet. Service oriented architecture with centralized High-Performance computing reduces the Hardware complexity and supports complex high computing functions like sensor fusion, machine learning algorithms etc.

The architecture of Adaptive AUTOSAR is based on key principles such as C++, Service Oriented Architecture (SOA), dynamicity, and parallelism. SOA in Adaptive AUTOSAR enables applications to offer and consume services across a network dynamically. In [7]describes solution for scalable extension of the existing system architecture with appropriate modules to enable communication with the Adaptive Applications (AAs) part of the system using proxy communication component. Adaptive Applications in Adaptive AUTOSAR are processes that can be started or stopped at any time, providing dynamic behavior compared to Classic AS. AAs can be added or removed individually, unlike partitions in ClassicAS that require re-flashing the entire software. The new AUTOSAR Runtime for Adaptive Applications (ARA) is a new entity that replaces the runtime environment (RTE) and offers dynamic linking of clients and services at runtime [15]. Functional clusters in AdaptiveAS provide functionalities as services to applications and are now processes that can be single threaded or multithreaded. The new resources like HPC and SOA require an operating system (OS) that can meet their dynamic needs. SOC (Service-Oriented Communication) allows for transparent inter and intra machine interactions. Classic AUTOSAR is static, whereas

Adaptive AUTOSAR offers planned dynamics in application deployment, communications, and resources.

The AUTOSAR Classic Platform will continue to be the top option for deeply embedded systems that realize typical power train and chassis functionality. These applications have strong requirements for determinism, real-time, and safety [8]. Adaptive AUTOSAR and Classic AUTOSAR are meant to co-exist and cooperate in the automotive industry, rather than replace each other.

SOME/IP is an automotive/embedded communication protocol which supports remote procedure calls, event notifications and the underlying serialization/wire format was developed by BMW group in the year 2011. It has emerged as the middleware of choice for Adaptive AUTOSAR implementation. Serialization, Remote Procedure Call (RPC), Service Discovery, Publish/Subscribe are the key features of SOME/IP protocol [9]. SOME/IP was designed to be a middleware which makes the existing traditional ECUs compatible with the new protocols and technologies. In this way the implementation of SOA can be done with minimal amount of change to the existing system. In [10] demonstrate Digital Cockpit on Linux operating system, within AUTOSAR Adaptive Platform have corresponding outputs in Cluster, Head up Display (HUD), and diagnostic visualization. Communication between these components is based on the GENIVI implementation of SOME/IP middleware which is called vsomeip. Adaptive Autosar contribution in the field of Vehicle2X connectivity by integrating MQTT is explained in [11]. MQTT (MQ Telemetry Transport) is a lightweight open messaging protocol widely used in consumer technology, particularly for Internet of Things. It runs over TCP/IP and is realized as publisher-subscriber.In [13] introduced an innovative method to handle the demands of DDAS (Distributed Driver Assistance Systems) basing on Service-oriented Architecture with distributed functionalities are encapsulated into services. Service-Oriented Driver Assistance (SODA) framework is introduced and evaluated in [13]. It uses the concepts of SOA to manage runtime adaptive automotive systems. A model-based development approach and a middleware architecture make up the two primary components of the SODA framework. This approach allows distributing the components in the vehicle network, well-defined interface to abstract heterogeneity of the various software functionalities, enhancing compatibility between software components developed by different suppliers and run time re-composing of services.

3. Selection of Machine learning algorithm.

We have developed an AC temperature prediction module which learns the AC temperature preferred by the user. The predictor considers the variables Heart Rate, Body temperature and Ambient Temperature to predict the target variable i.e. AC temperature. The methodologies incorporated to develop the module is described in the further sections. For initial database creation, a set of 389 data points of Cabin temperature, Body temperature, Ambient Temperature and AC temperature from Kaggle [14] which is a data science community containing variety of datasets. As there were no direct dataset available matching out requirement we looked for different categories where these variables were individually available and then combined them together as per our requirement.

Once all the data was converted in preferred format these data were transferred to a database (.db) file using python's sqlite3 package. Sqlite3 package allows to transfer contents of a .csv file into database file. This database file will then be used in further stages to train the prediction model. Finding the relationship between variables used for prediction helps in knowing whether they have a strong relationship with each other. For example, if two variables have a strong relationship between each other, then, one out of the 2 variables would be enough to predict the output variable. It also proves useful in finding which algorithm could possibly be used for prediction. For example, if the variables are linearly related then linear regression could be used for prediction. In order to find the association between data present in the data points collected by us we calculated covariance and correlation coefficients between the independent variables and the data points.

Relation	Cabin temp	Ambient temp	Body temp
Covariance	0.236	1.7399	-0.1201
Pearson's correlation	0.158 (Weak)	0.447(Moderate)	-0.096(Weak)
Spearman's correlation	0.2686(Weak)	0.44(Moderate)	-0.11(Weak)

As seen in the above table, we didn't find any significant linear relationship between the variables and hence using an algorithm like linear regression was ruled out at this stage.

To further identify any possible relationship between the independent and target variable we plotted the data points to check if there's a possibility of probably an exponential relationship between the data points. Package used : matplotlib for plotting data.

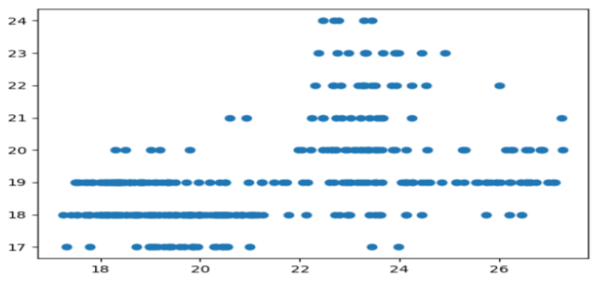


Figure 3: sklearn library's preprocessing for heart rate

As seen in the above plots Figure 3 and 4, there is no linear, nonlinear or exponential relationship that could be derived out of the data points. Hence, we tried to check if normalizing the data could be of help in bringing out any relation between the data variables.

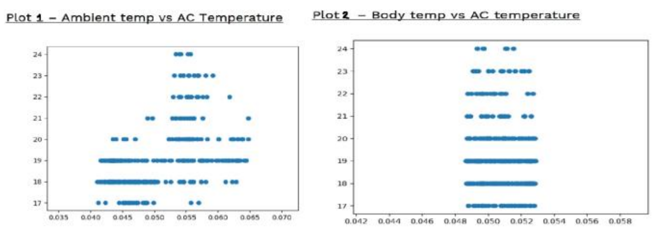


Figure 4: Sklearn library's preprocessing Ambient and Body temperature.

Since no direct relationship could be derived out of the above tests we decided to proceed by testing different algorithms and then choosing the best among them. Using Scikit learn package, we have tested below listed algorithms.

a) Decision Tree Regressor
b) Random Forest Regressor
c) Lasso Regressor
d) Polynomial Regressor
e) Adaboost Regressor

Model	Ratio of train and test data	Temp_in vs ac_temp	Temp_out vs ac_temp	Body temp vs ac_temp
Decision tree regressor	90:10	Mae - 0.84 Mse – 1.56	Mae – 1.3 Mse – 3.56	Mae – 1.48 Mse – 4
Random forest regressor	90:10	Mae - 0.84 Mse – 1.56	Mae - 1 Mse – 1.82	Mae - 0.14 Mse –0.008
Lasso regressor	90:10	Mae - 0.92 Mse – 1.8	Mae – 1.02 Mse – 1.94	Mae - 0.92 Mse – 1.84
Polynomial regressor	90:10	Mae - 0.94 Mse – 1.92	Mae – 0.97 Mse – 1.743	Mae - 0.92 Mse – 1.8
Adaboost regressor	90:10	Mae - 0.84 Mse – 1.56	Mae - 0.84 Mse – 1.56	Mae – 1.2 Mse – 2.58
Decision tree regressor	80:20	Mae - 0.7 Mse – 1.16	Mae – 1.11 Mse – 2.5	Mae – 1.34 Mse – 3.19
Random forest regressor	80:20	Mae - 0.7 Mse – 1.16	Mae - 0.94 Mse – 1.76	Mae – 1.2 Mse – 2.84
Lasso regressor	80:20	Mae - 0.83 Mse – 1.52	Mae - 0.94 Mse – 1.64	Mae - 0.83 Mse – 1.52
Polynomial regressor	80:20	Mae - 0.84 Mse – 1.56	Mae - 0.85 Mse – 1.37	Mae - 0.83 Mse – 1.52
Adaboost regressor	80:20	Mae - 0.69 Mse – 1.12	Mae - 0.76 Mse – 1.15	Mae – 1.33 Mse – 2.56

Figure 5: A table of MAE and MSE obtained

We tested these algorithms by splitting the dataset we collected into different ratios of test and train data i.e. Train: Test = 90:10 & 80:20. We used Scikit learn library's train_test_split module to split the giving dataset into test and train data of required ratio.

Two metrics are used to evaluate the prediction error of an algorithm, Mean Square Error (MSE) and Mean Absolute Error (MAE). Mean absolute error calculates an average of differences between predicted and expected value, gives an idea about how wrong the predictions are. MAE value = 0 indicated perfect prediction. Similar to MAE , Mean squared error takes also squares the differences between the actual and predicted value and lower value of MSE gives better the predictions. We selected algorithms having the least MSE and MAE.

Input	Algorithm	Output
Hear Rate	Polynomial	AC Temperature
Ambient Temperature	Decision Tree	AC Temperature
Body Temperature	Adaboost	AC Temperature

The idea here is to use 3 different algorithms to predict AC temperature using the 3 variables as separate inputs and then combine the predictions i.e., take an average of the 3 predictions. Using above approach we could achieve an accuracy of 55%. To improve accuracy of prediction, instead of taking an average of 3 predictions we compare the 3 predictions, if 2 of the 3 predicted temperatures are same then we consider that as the final prediction. If all three are different, then only we average the predictions and consider it as the final prediction.

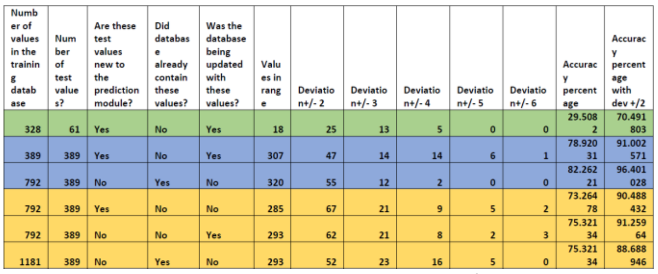
4. Improvement of Accuracy

We came across a technique called as the K-fold cross validation. The dataset is divided into K-folds, in our case, K=10. Then the entire dataset (389) points will be divided into 10 folds, roughly 39 values per fold which will be used as test data and the remaining values would be used to train the model. By this method, the prediction model will be able to train, and test will all possible sections of the data. And hence, we would have a better view of how the prediction module behaves with different kinds of test and training data. We have used Sklearn library's KFold module is used for diving data into K folds And Sklearn library's mean absolute_error and mean_squared_error modules are used to find MAE and MSE. We designed a user feedback loop wherein the user enters temperature value when he is not satisfied with a prediction. All the values in training data which are in the range +/-0.5 of test data will be looked for and their AC temperature will be changed in accordance with user feedback.

We tested multiple algorithms Decision tree, Random forest, KNN regressor, Adaboost and support vector regressor for weight prediction for new incoming test values. With the weight obtained we predicted AC temperature with our usual approach and calculated the deviation. Comparing all the algorithms, we finalized Support Vector Regression to be apt for weight prediction. Once all the blocks were developed, we modularized the code into 5 separate modules.

1) Temperature_prediction_module
2) Prediction_finalization_module
3) User_feedback_module
4) Prediction_correction_module:
5) Training_data_updation

With our prediction module we could reach the maximum accuracy of 96.4% as shown in figure 6.



Number of values in the training database	Number of test values?	Are these test values new to the prediction module?	Did database already contain these values?	Was the database being updated with these values?	Values in range	Deviation +/- 2	Deviation +/- 3	Deviation +/- 4	Deviation +/- 5	Deviation +/- 6	Accuracy percentage	Accuracy percentage with dev +/2
328	61	Yes	No	Yes	18	25	13	5	0	0	29.5082	70.491803
389	389	Yes	No	Yes	307	47	14	14	6	1	78.92031	91.002571
792	389	No	Yes	No	320	55	12	2	0	0	82.26221	96.401028
792	389	Yes	No	No	285	67	21	9	5	2	73.26478	90.488432
792	389	No	No	Yes	293	62	21	8	2	3	75.32134	91.25964
1181	389	No	Yes	No	293	52	23	16	5	0	75.32134	88.688946

Figure 6: Reliabilty check with two sets of 389 datapoints

5. Implementation of AC personalisation application

AC personalization implementation is divided to two platforms, Tc37x Aurix Eval Board ported with classic Autosar for data acquisition from the sensors and R-Car H3e board ported with Adaptive Autosar hosting Machine learning algorithms performs as High computing machine. We have used potentiometers instead of actual sensors, the reason is that if we use actual sensors testing of multiple iterations with different temperature and heart rate values are not possible. SOA communication is established between the classic and Adaptive node over SOME/IP. The final Machine learning predicted temperature and individual machine learning outputs are captured in ANDi tool over SOME/IP. We have monitored all network communications in Wireshark. The ML framework and High-level hardware interaction diagrams are shown in Figure 7 and 8 respectively.

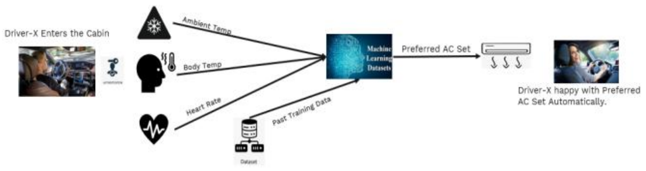

Driver-X Enters the Cabin
Ambient Temp
Body Temp
Heart Rate
Past Training Data
Dataset
Machine Learning Datasets
Preferred AC Set
Driver-X happy with Preferred AC Set Automatically.

Figure 7: Stages of ML frame work

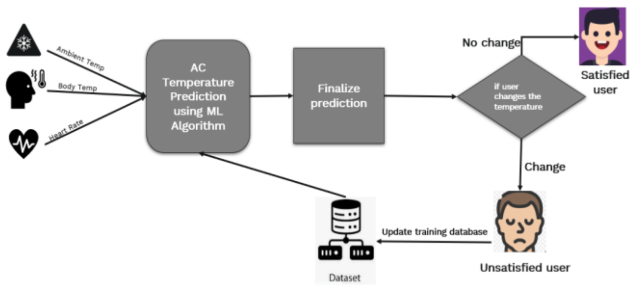


Figure 8: AC personalisation workflow diagram.

Figure 9 demonstrates the high level hardware interactions. All the three communication nodes are connected to an external ethernet switch.

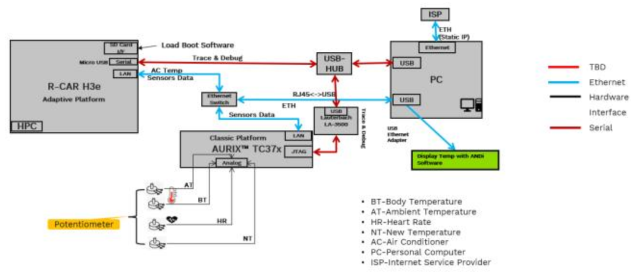


Figure 9: High level Hardware interaction diagram.

SOME/IP communication scheme is elaborated in Fig 10 and 11. Communication starts with Service Discovery (SD) and on successful subscription of event, RCAR will send event data continuously to tester PC which runs Andi scripts. Similarly, Aurix classic platform will send potentiometer values to RCar and shown in figure 10. Here classic node will act as server and RCar will be subscribing to event data.

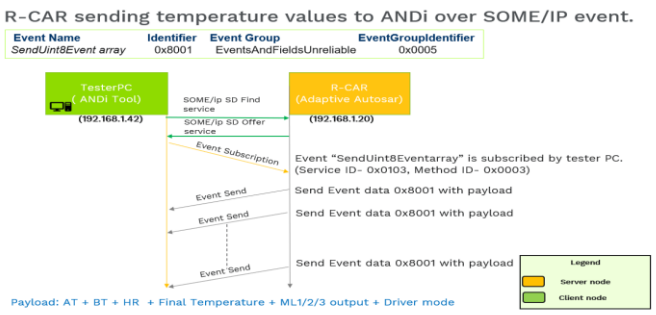

R-CAR sending temperature values to ANDi over SOME/IP event.
Event Name
SendUint8Event array
Identifier
0x8001
Event Group
EventsAndFieldsUnreliable
EventGroupIdentifier
0x0005
TesterPC (ANDi Tool)
(192.168.1.42)
R-CAR (Adaptive Autosar)
(192.168.1.20)
SOME/ip SD Find service
SOME/ip SD Offer service
Event Subscription
Event "SendUint8Eventarray" is subscribed by tester PC. (Service ID- 0x0103, Method ID- 0x0003)
Event Send
Send Event data 0x8001 with payload
Event Send
Send Event data 0x8001 with payload
Event Send
Send Event data 0x8001 with payload
Legend
Server node
Client node
Payload: AT + BT + HR + Final Temperature + ML1/2/3 output + Driver mode

Figure 10: SOME/IP communication scheme -Sending data

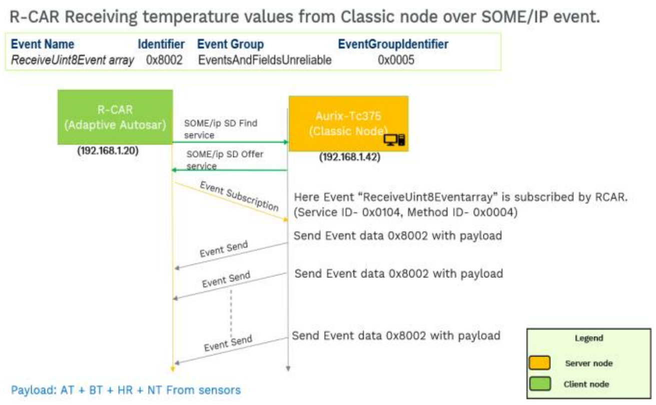
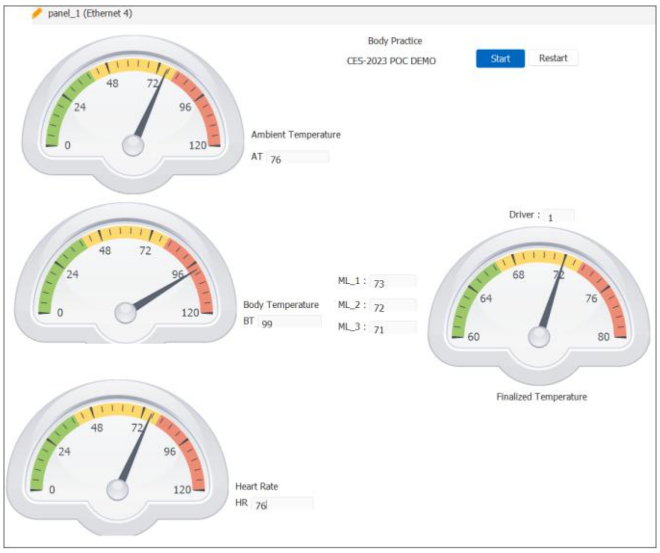

R-CAR Receiving temperature values from Classic node over SOME/IP event.
Event Name
ReceiveUint8Event array
Identifier
0x8002
Event Group
EventsAndFieldsUnreliable
EventGroupIdentifier
0x0005
R-CAR (Adaptive Autosar)
(192.168.1.20)
Aurix-Tc375 (Classic Node)
(192.168.1.42)
SOME/ip SD Find service
SOME/ip SD Offer service
Event Subscription
Here Event "ReceiveUint8Eventarray" is subscribed by RCAR. (Service ID- 0x0104, Method ID- 0x0004)
Event Send
Send Event data 0x8002 with payload
Event Send
Send Event data 0x8002 with payload
Event Send
Send Event data 0x8002 with payload
Legend
Server node
Client node
Payload: AT + BT + HR + NT From sensors

Figure 11: SOME/IP communication scheme -Receiving data.

Figure 12: AC Personalisation demo setup

We have developed ANDI scripts for SOME/IP client which subscribes for event which triggers continuously sending data from RCar which act as server. Panel are developed in Andi tool to showcase input values and final ML predictions as shown in figure 13.



Ambient Temperature
AT 76
Body Temperature
BT 99
Heart Rate
HR 76
ML_1 : 73
ML_2 : 72
ML_3 : 71
Driver : 1

Figure 13: AC Personalisation output on ANDi tool

Acknowledgment
We would like to take this opportunity to express our gratitude to Muralidhara K V - Associate Vice President KPIT technologies for his exceptional support and guidance throughout this research and development as well as providing all hardware and software tools required for the prototype development. Working under his supervision allowed us to expand our skill sets while also fulfilling important research background requirements.

6. Conclusion and Future Scope

We have showcased latest Service Oriented communication Architecture and co-existence of adaptive and classic Autosar nodes through AC personalization application.ML based application predicts preferred AC temperature accurately. We can extend this work to cloud computing to improve accuracy. Service Oriented Architecture (SOA) Interface is recommended in Domain controller architecture which gives flexibility of migrating application between Central Processing units. Security functions are not included in SOME/IP. We can migrate the same application into DDS (Data Distribution Service) which offers high security and encryption inbuilt, or we can add security functionalities externally to SOME/ IP but it can lead to more cost. Implementing this prototype on a real car is the immediate future work in this field with diverse occupant settings to understand the effects of certain extremes conditions and improve ML predictions.

References

[1] Thomas Liebetrau, “ E/E Architecture Transformation How it impacts value chain and networking technologies.”, AmE 2022 - Automotive meets Electronics; 13. GMM-Symposium, 25 January 2023.
[2] K. P. Sreevishakh and S. P. Dhanure, "A Review Paper on Automotive Crash Prediction and Notiication Technologies," 2015 International Conference on Computing Communication Control and Automation, Pune, 2015, pp. 999-1002.
[3] Dr. Rajeev Sobti, Parampreet Kaur,“Model-based Architecture of Software-intensive Intelligent Automotive Systems”, 2018 4th International Conference on Computing Sciences.
[4] Sihem NITA, Salim BITAM, Abdelhamid MELLOUK, “ A Body Area Network for Ubiquitous Driver Stress Monitoring based on ECG Signal “, 2019 International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob).
[5] Marius Minea et al, “ Non-Intrusive Driver Condition Monitoring in Highly Automated Vehicles with Medical Information Support for Emergency Calling”, University Politehnica of Bucharest, Transports Faculty Bucharest, Romania.
[6] Diana Hintea, James Brusey and Elena Gaura, “ A Study on Several Machine Learning Methods for Estimating Cabin Occupant Equivalent Temperature “,In Proceedings of the 12th International Conference on Informatics in Control, Automation and Robotics (ICINCO-2015), pages 629-634.
[7] Mia Stepanovic, Milan Bjelica, Ivan Kastelan, Gordana Velikic, “ Scalable approach to extending automotive software using AUTOSAR Adaptive stack “,University of Novi Sad Novi Sad, Serbia.
[8] Simon Fürst, Dr.-Ing Markus Bechter,” AUTOSAR for Connected and Autonomous Vehicles The AUTOSAR Adaptive Platform”, BMW Group Munich, Germany.
[9] Gopu G.L. Kavitha K.V. James Joy, “ Service Oriented Architecture based connectivity of Automotive ECUs “, 2016 International Conference on Circuit, Power and Computing Technologies [ICCPCT].
[10] Mila Kotur, Marko Dragojević,” Digital Cockpit in AUTOSAR Adaptive Context”, 2018 IEEE 8th International Conference on Consumer Electronics - Berlin (ICCE-Berlin).
[11] Robert Šandor, Mia Stepanović, ” Vehicle2X communication proposal for Adaptive AUTOSAR ” , 2018 IEEE 8th International Conference on Consumer Electronics - Berlin (ICCE-Berlin).
[12] M. Wagner, D. Zöbel and A. Meroth. "Towards an Adaptive Software and System Architecture for Driver Assistance Systems". in Proc. of the 4th IEEE International Conference on Computer Science andInformation Technology, 2011, vol. 4 pp. 174-178.
[13] M. Wagner, D. Z¨obel, and A. Meroth, “SODA: Service-oriented Architecture for runtime adaptive Driver Assistance Systems,” in Proceedings of the 17th IEEE Computer Society symposium (ISORC). Reno, NV: IEEE Computer Society, 2014.
[14] https://www.kaggle.com/
[15] http://www.autosar.org