WRITTEN BY

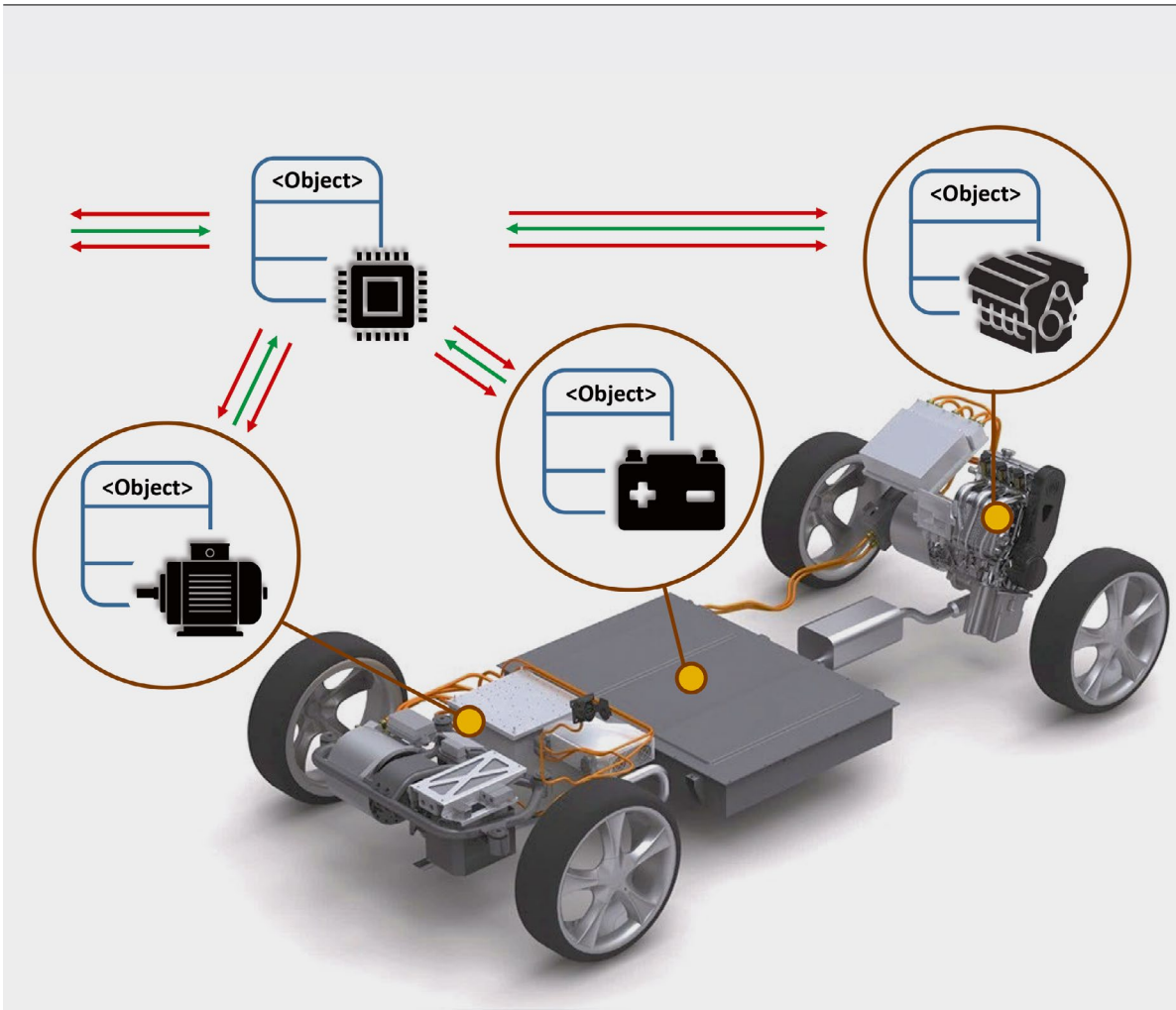
Maximilian Stumpp, M. Sc.
is Research Associate at the Institute for Internal Combustion Engines and Powertrain Systems (vkm) of TU Darmstadt (Germany).

Alexander Kuznik, M. Sc.
is Research Associate at the Institute for Internal Combustion Engines and-Powertrain Systems (vkm) of TU Darmstadt (Germany).

Prof. Dr. Christian Beidl
is Head the Institute for Internal Combustion Engines and Powertrain Systems (vkm) of TU Darmstadt (Germany).

Modular Object-oriented Architectures for Scalable Hybrid Powertrains

Object-oriented architectures are already being applied in various applications to design subsystems in a modular way and to scale their functionalities and interactions within the system. In the FVV research project no. 1428 “Modular Hybrid Powertrain”, principles for object-oriented designs and their transfer to hybrid powertrain systems and individual components were developed at TU Darmstadt. This makes it possible to exchange components and configurations to create variants without increasing the development effort.



© TUD

1 FLEXIBLE ARCHITECTURES

Future vehicle platforms are expected to feature a wide variety of different powertrain configurations - the trend is already expressed today in vehicles with Mild Hybrid Electric Vehicles (Mild-HEVs), Full-HEVs and Battery Electric Vehicles (BEVs). Modular platforms are already widely used to reduce production costs and to simplify supply chains. One and the same component is used in different vehicle variants. This is preceded by a high development effort in terms of the applications of each variant, which is based on the complexity and high interdependencies of the subsystems. A cruise control system calculates a drive or braking torque to maintain or achieve a desired speed based on the characteristics and physics of the vehicle variant it is equipped with. If only one component is changed for a different variant, the control system must be adjusted with new parameters.

In information technology, development is based on the principles of Object-oriented Design (OOD), providing flexibility in the replacement or extension of functionalities. The principles are not limited to software but can be extended and applied to hardware elements as well [1]. The interaction between a computer and a printer can be mentioned as an example.

The goal of the FVV research project, which was carried out at the Institute for Internal Combustion Engines and Powertrain Systems (vkm) of TU Darmstadt, is to achieve the same level of flexibility with a variety of components in powertrain development, **FIGURE 1**. This is accomplished through the introduction of object-oriented architectures and the design of component interactions using standardized interfaces. The verification of the developed architecture is performed simulation-based.

2 PRINCIPLES

The development of an OOD-based system architecture proceeds as follows [2]:

– abstraction and definition of objects
– adherence to consistent encapsulation (decoupling of functions)
– design of interfaces according to the development principle of "design by contract"
– creation of a scalable and at the same time robust solution.

The main difference between previous system architectures and OOD is illustrated in **FIGURE 2** by the communication of a controller with an Electric Motor (EM). Traditional systems are primarily designed as classical control loops, where the EM is directly controlled by specific load request. To calculate the load signal, all relevant information about the EM needs to be stored in the controller, requiring a priori knowledge of the boundary conditions.

An object-oriented system design, on the other hand, is based on negotiating the boundary conditions, with all information about the EM encapsulated in the newly defined object "EM". Consequently, the development of the controller can proceed independently of the specifications of the EM being used. An object is thus defined as a hardware or software component that abstracts functionality and characteristics. Each object has attributes (or parameters) and functions that are publicly accessible for other objects. [3]

The design of the interaction between objects is a crucial factor in achieving complete modularity. The fundamental form of communication is realized through the so-called "design-by-contract"

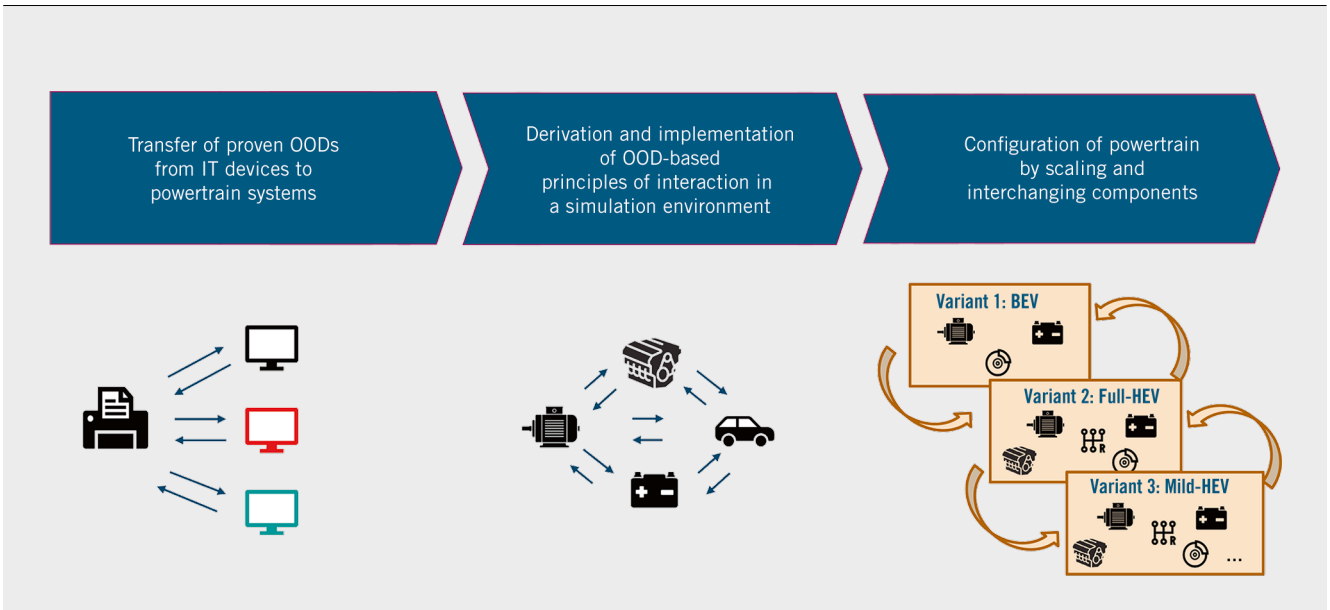


FIGURE 1 Concept of transferring OOD principles to the development of hybrid drive systems (© TUD)

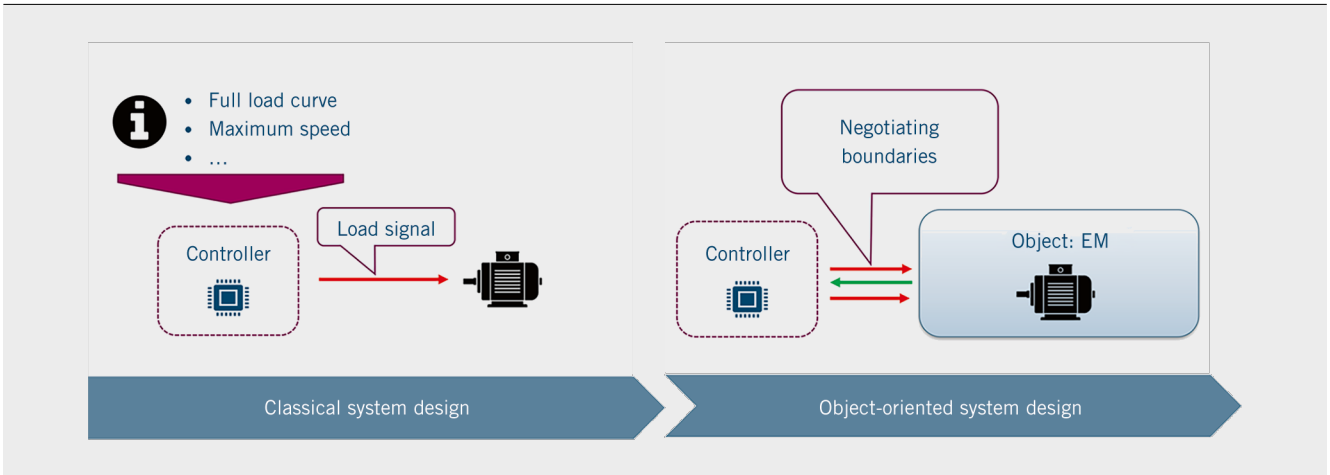


FIGURE 2 Classical versus object-oriented system design based on negotiation (© TUD)

principle. This involves a negotiation between two objects with requests and responses. **FIGURE 3** illustrates the negotiation process in three steps [3]:
– placement of a service request
– negotiation of boundary conditions
– signment and execution of the contract.
For a torque request, this involves the exchange of the maximum and minimum available torques. These boundary conditions apply to the respective requested time and are therefore situation-dependent and dynamic. The information and services that can be requested in each case must be defined through a protocol as a standardized interface. Such a protocol also includes, among other things, the exact signal forms, units, and sampling rates. This enables the independent development of the respective subsystems or objects.

3 OBJECT-ORIENTED DESIGN OF THE POWERTRAIN

The abstraction of the system includes the analysis and study of all components and functionalities. The objects of the OOD-based architecture are formed in such a way that clear main functions can be assigned in each case. In a hybrid powertrain, the EM has the main function of providing torque, analogous to an Internal Combustion Engine (ICE). The objects can now be specifically defined with attributes such as full-load characteristic curves. Upon closer examination of the system, certain functional dependencies between individual objects can be identified. The EM requires the battery as an electrical power source. Since the provision of torque by the EM in the considered system is always accompanied by a power supply from the battery, both objects can be assigned to a common object called “electric powertrain”. This facilitates interaction with other objects, as communication can be handled through a single object. Similarly, a common object called “ICE powertrain” can be introduced, which combines the internal combustion engine (ICE) with the gearbox and the clutch.

Another necessary abstraction that needs to be made is the functional abstraction into hierarchical levels. The longitudinal motion of the vehicle is described in different levels of abstraction. The basic task of the powertrain includes the kinematic planning of the vehicle in time and space. This fundamental function can be carried out by objects, for instance the driver, cruise control, and other Advanced Driver Assistance Systems (ADAS). To ensure

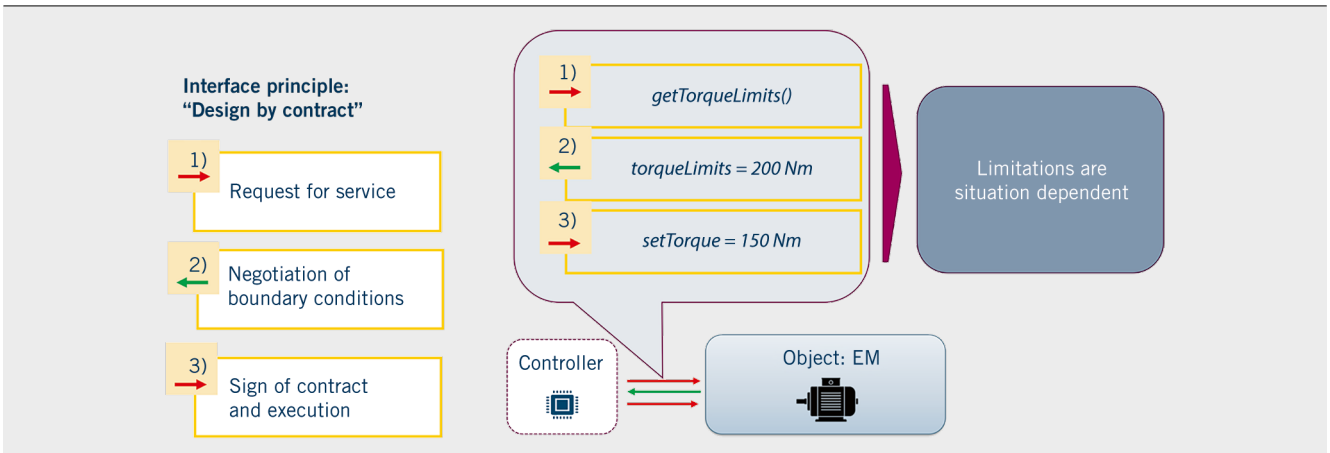


FIGURE 3 Interaction of objects by means of negotiation based on design by contract (© TUD)

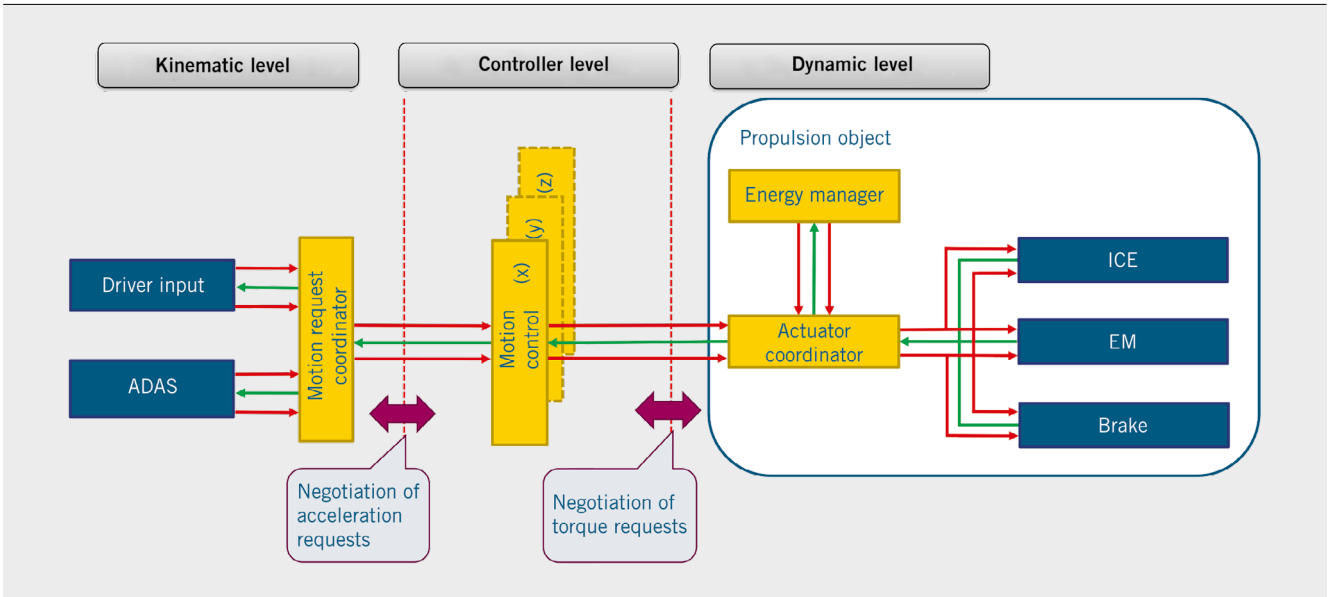


FIGURE 4 Object-oriented architecture of the hybrid powertrain (© TUD)

a consistent decoupling from the functions of the actuator objects, the objects at the kinematic level are only allowed to request an acceleration instead of a torque. Actuator objects provide torque and are thus responsible for physical execution of the kinematic planning. This results in a dynamic level and a controller level for translating acceleration requests into torque requests. For this purpose, a new object "motion control" needs to be introduced, which calculates a torque request based on the vehicle mass and the desired acceleration.

FIGURE 4 shows the complete OOD-based architecture. Two additional objects that must be introduced can be seen. Since there are multiple objects in each level with the same main functionality, so-called coordinators are required. They are also necessary to completely decouple the levels from each other. Objects in the kinematic and controller levels must not depend on the existence of individual actuator objects. The object "motion control" should only request the overall available maximum and minimum torque. The object "actuator coordinator" is responsible for the implementation on the dynamic level. It is aware of the existence of individual actuator objects and is responsible for the prioritization and the distribution of the torque requests. For implementing an energy-optimal torque distribution, the object "actuator coordinator" can consult the recommendations provided by the newly defined object "energy manager." It is important to note that the final decision regarding the torque distribution lies with the object "actuator coordinator." A coordinator is also necessary at the kinematic level. The object "acceleration request coordinator" is tasked with accepting, prioritizing, and negotiating the acceleration requests with the object "motion control."

4 IMPLEMENTATION AND VERIFICATION

The OOD-based architecture is implemented and verified in a simulation environment. The objects are modeled in Matlab/Simulink, **FIGURE 5**. Classical component models are abstracted into objects and defined accordingly with attributes and methods. The driver, the rest of the vehicle, the virtual environment, and the road are modelled in IPG CarMaker. The verification of the developed approach is done by scaling or replacing the corresponding actuator objects, whereby three vehicle variants are configured.

The functionality of full modularity is demonstrated by introducing new technologies as new objects and the system response to the boundary behavior of individual objects, such as the lower and upper state of charge of the battery. Furthermore, the integration of a new object is demonstrated using the example of the electrically heated catalyst and how the interaction can be realized in the overall system in an object-oriented manner.

Here, a selected application example is described: the interaction of the objects after replacing the EM of a 48-V hybrid vehicle with an overload-capable machine during an uphill drive at a constant speed. **FIGURE 6** shows that the torque requirement exceeds the nominal power of the EM, which communicates the maximum and minimum torques in overload operation to the object "actuator coordinator". After 6 s, the maximum duration of the overload is reached, so the object "electric powertrain" reports lower torque limit values. In response to the driver's request and the changed conditions, the object "actuator coordinator" decides, based on the recommendation of the object "energy manager", to start the internal combustion engine. The OOD architecture enables the vehicle control system to adapt flexibly to different conditions of EM operation without explicitly being applicated for it.

5 CONCLUSION

As a result of the FVV research project, a fully modular architecture has been developed based on object-oriented design principles. It has been demonstrated that existing component models and controllers can be defined as objects with attributes and methods, and design by contract principles are applicable for communication. A structured division into kinematic and dynamic levels is necessary for decoupling, as well as the introduction of

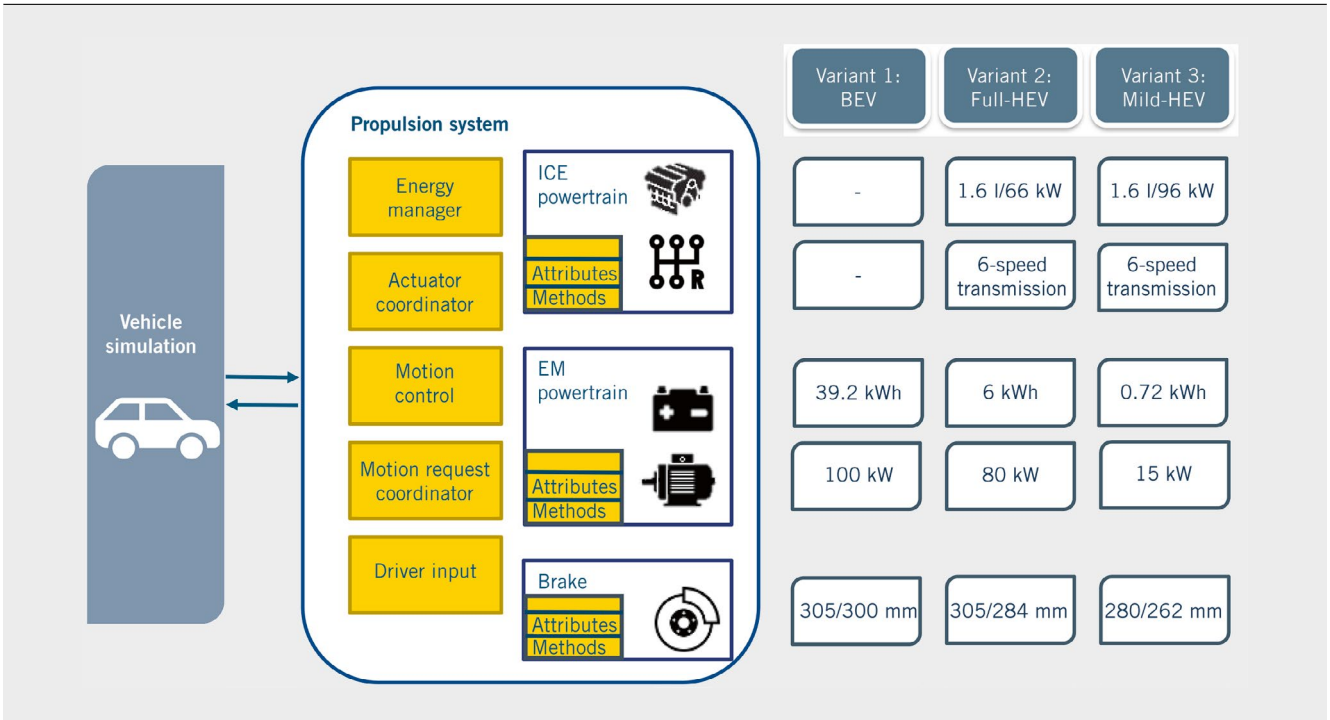


FIGURE 5 Simulation setup of the OOD for three vehicle variants (© TUD)

new coordinator objects. The coordinators play a crucial role as they form the central interfaces within their respective levels. Additional objects, such as the "energy manager ", can provide torque recommendations as advisory objects, but they should never directly control the actuators. The architecture has been transferred to a simulation environment and validated using selected application examples. It has been shown that vehicle configurations can be created and functionally implemented by scaling individual objects. Furthermore, new technologies can be integrated into the objects without requiring modifications to the entire system.

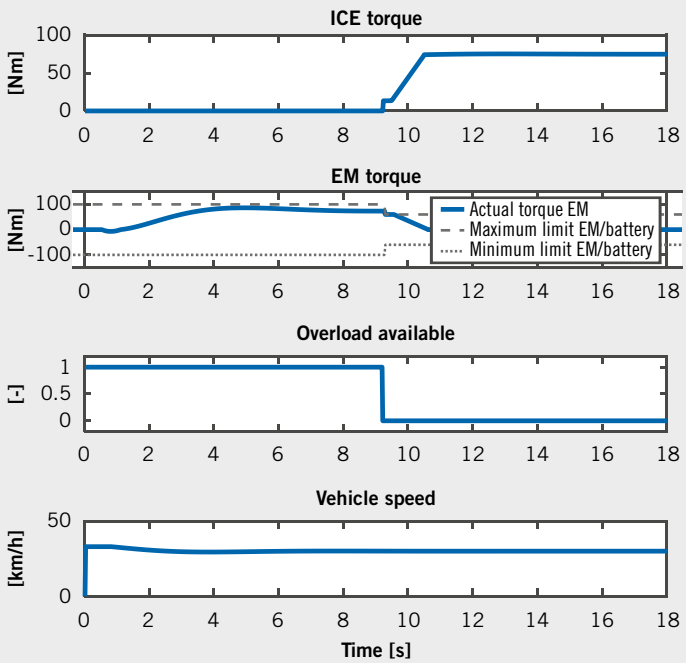


FIGURE 6 Torque split with EM operating with overload capacity (© TUD)

REFERENCES

[1] Martin, R. C.: Pearson new international edition. Agile software development, principles, patterns, and practices (1st ed.). Harlow: Pearson Education Limited, 2014

[2] Held, V.; Heitmann, A.: Modularization of vehicle control systems based on the application of object-oriented design principles. 8th International Munich Chassis Symposium 2017, chassis.tech plus, Munich, 2017

[3] Booch, G. et al.: Object-oriented analysis and design with applications (3rd ed.). Upper Saddle River, N.J.: Addison-Wesley, 2007

THANKS

The research project (FVV project no. 1428) was performed by the Institute for Internal Combustion Engines and Powertrain Systems (vkm) of TU Darmstadt under the direction Univ.-Prof. Dr. techn. Christian Beidl. It was funded by the FVV e. V. The project was conducted by an expert group led by Dr. Veit Held (formerly Stellantis/Opel Automobile) and Dr.-Ing. Thomas Opitz Held (formerly Stellantis/Opel Automobile). The authors gratefully acknowledge the support received from the from the FVV and from all those involved in the project.