Technische Universität München
TUM School of Engineering and Design

Test Automation in Software-defined Car Manufacturing with Executable BPMN

Simone König

Vollständiger Abdruck der von der TUM School of Engineering and Design der Technischen Universität München zur Erlangung einer

Doktorin der Ingenieurwissenschaften (Dr.-Ing.)

genehmigten Dissertation.

Vorsitz: Prof. Dr.-Ing. Markus Lienkamp

Prüfende der Dissertation:

1. Prof. Dr.-Ing. Birgit Vogel-Heuser
2. Prof. Dr. Alexander Pretschner

Die Dissertation wurde am 12.11.2024 bei der Technischen Universität München eingereicht und durch die TUM School of Engineering and Design am 25.04.2025 angenommen.

Acknowledgments

First, I thank my supervisor, Professor Birgit Vogel-Heuser, who allowed me to pursue a Ph.D. under her supervision. During the time of my Ph.D., I was allowed to learn a lot from her.

I thank Professor Markus Lienkamp and Professor Pretschner.

Furthermore, I am grateful to the Institute of Automation and Information Systems at the Technical University of Munich.

In addition, I would like to thank Dr. Rainer Mäckel, Dr. Oliver Kopp, Dr. Michael Hahn and Maximilian Reihn. The industrial experts involved in the SofDCar project at Mercedes-Benz AG enabled me to evaluate my work under OEM requirements and real-world automotive use cases.

Last, I thank my friends, family, and my parents for their support and encouragement.

Table of Contents

1. Introduction

An unprecedented level of variability and flexibility marks the ongoing trend in the automotive industry (Egyed et al., 2023). Car lines support various propulsion systems including traditional combustion engines, plug-in hybrids and electric motors (Keller et al., 2013), but also varying levels of autonomous driving, and other customized software features. This diversity opens opportunities for new entrants in the Original Equipment Manufacturer (OEM) market to offer specialized products, while posing a challenge for established car manufacturers to maintain their competitiveness under the need of reducing complexity costs (Gülke et al., 2014).

In this context, testing and diagnostics of electric and electronical (E/E) car components is relevant for safety, certification, and emission release. At the same time, the framework conditions are changing: new standards for essential automotive component functions are being developed by associations such as ASAM e.V. (2024a) in order to adapt interfaces for OEMs and suppliers. Moreover, there is also more regulation on the part of legislation, e.g., the EU Data Act (Bundesministerium für Digitales und Verkehr, 2023), which includes improved access to data for car customers.

In this dynamic industrial environment, cross-cutting processes as E/E testing and diagnostics are in transition. Thereby, technological innovation is vital to staying competitive (Vogel-Heuser, 2016). Digital transformation plays a crucial role (Vogel-Heuser & Wimmer, 2023), not only enabling the variant management for new car lines and governmental regulations, but also helping to reduce internal company costs to stay efficient. Especially for the process of complex mass production, there is high potential on optimizing and improving the efficiency of processes (Schäffer et al., 2021).

For car testing and diagnostics as interdisciplinary engineering process from car development over manufacturing to after sales, the common basis builds a flexible testing framework to verify reliable scenarios (Kirchhof et al., 2021). The basis has consequences on a high quality standard for the product car (Schindewolf et al., 2024).

1.1. Motivation and Hypotheses

Selected aspects of industrial automation are studied for testing and diagnostics in cars during manufacturing line processes (Becker et al., 2022). This thesis is grounded in a structured survey from a German automotive OEM from the year 2020,

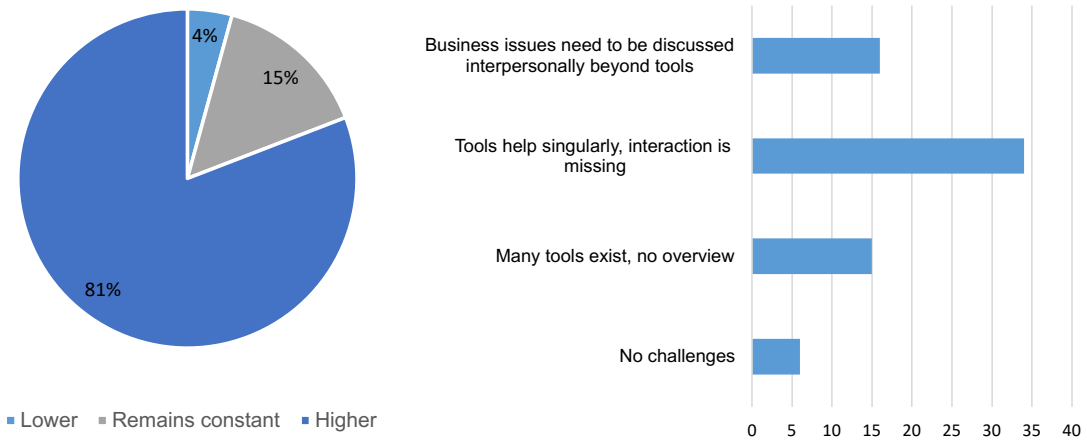


Figure 1: Evaluation of an OEM survey in the year 2020; left: Estimation of medium-term complexity in business processes; right: Assessment of the IT tool landscape [% of participants]

that identified a need for innovations in business process automation to simplify and accelerate work between interdisciplinary teams during car production (see Figure 1 for the results). The survey was conducted among 63 employees in the business of E/E testing and commissioning of passenger cars in automotive manufacturing. Participants were 56 employees (53 on business side, three on IT department), seven managers (six from business, one from IT), whereas 53 were men and ten women. This distribution reflected 25% of the employees in the business, so it was a representative sample size of engineers, computer scientists, workers, business analysts, and corresponding managers.

In the first question, the employees were asked to estimate the complexity of their daily work within the next five years. 81% of the employees answered with an increase, 15% saw no change, and 4% believed the complexity would decrease. The result is shown on the left-hand side of Figure 1. The reasons for the increase in complexity originate in the numerous engine types installed in passenger cars. As a result, the existing tests and diagnostic procedures must be adapted to the new engine types. Software-bound customer functions are also increasing due to automated driving with various sensors. The number of sensors increases, and the latest software and E/E architecture must handle this variance, creating higher test variance. Shorter life cycles of software and diagnostic functions result in more updates and upgrades, impacting change frequency. While this does not directly count as complexity, it requires an adapted car software update and upgrade concept with streamlined internal business processes. Consequently, business processes for car testing and diagnostics as test planning and scheduling should be processed automatically to assist employees.

The second question assessed the current IT tool landscape and the including employee support. Respondents had the option of selecting several answers. The results are shown on the right-hand side of Figure 1. 34 percent of the respondents felt that the current IT tools help, but a cohesive IT ecosystem needs to be installed. An ecosystem can support users with end-to-end processes mapped transparently in the enterprise IT landscape. 15 percent of the respondents interact with many IT tools in their daily business, but an overview needs to be provided. 16 percent only trust on personal exchanges and IT tools to complete their daily work. This especially applies to tool support in failure analysis. 6 percent of the respondents are satisfied with the tools available and demand no further action. The interviewees agreed that the interconnection of the IT systems would support them in their daily work. As a starting point of an ecosystem, business process automation helps to bridge IT system gaps for continuous engineering.

All in all, there is a need for speed in the automotive software development and engineering processes, e.g., as illustrated in Table 1. So methods that help to improve working in an interdisciplinary domain are beneficial (Jolak et al., 2020).

To this end, model-based engineering in terms of automotive testing is already investigated at a German OEM (Kriebel et al., 2018). It means that the I/O behavior of a valid model should be compared to a system under test.

Moreover, business process models are also an important foundation for automotive digital twins (Pfeiffer et al., 2024). These findings are the starting point to evaluate solutions from business informatics to optimize the respective business processes.

Criteria	**Today's typical car**	**Software-defined car**
Development cost	$$$$	$
Time to first prototype	Months	Hours
Time to market	3 years	Weeks to months
Approach	First-time right at Hardware Start of Production	Minimal Viable Product and continuous improvement
Scalability	Every car model needs different code	Same code for all car models

Table 1: Change in automotive software development (kindly adapted from (Slama et al., 2023))

Especially test planning, implementation, scheduling, and execution are of interest in order to assist employees in their future work during car and corresponding manufacturing ramp-up phase as well as series production.

In this thesis, three main challenges of the area in car testing and E/E diagnostics are addressed with focus on car manufacturing.

Reliability is crucial to achieve test quality. This is not only relevant for test failure analysis (Jordan et al., 2022), but also for complexity management when a maximal number of tests with variants should be performed and the resources are limited as in car ramp-up phases. Especially manual testing process can suffer from limited know-how in the selection of an automation method (Haas et al., 2021).

Challenge 1: Inadequate manual approach to manage complexity in test planning during car ramp-up phases.

The complexity of variants and configurations in software engineering for automotive systems is not restricted to a special OEM and requires new test methodologies (Pretschner et al., 2007). Moreover, the integration of in-car and off-car IT technologies becomes more and more relevant. Thus, agile orchestration is important (van Kempen et al., 2023).

Challenge 2: Inadequate offline approach to manage variants for test scheduling during car manufacturing.

Model-driven engineering has positive effects on the quality management (Verbruggen & Snoeck, 2021). Model-based testing and its automation is under research (Pretschner et al., 2017), even though OTX is an ISO standard to represent automotive testing workflows (International Organization for Standardization, 2011). In the case of vehicle diagnostics new tools and technologies from non-automotive areas are seen as challenge and at the same time, the transfer of standardized IT methods to the automotive domain is an opportunity (Bickelhaupt et al., 2023). The combination with the survey's results of Figure 1 lead to Challenge 3.

Challenge 3: Lack of capable approaches to support continuous business process engineering under existing automotive vendor-specific frameworks.

To place these challenges into current research trends, the notation of software-driven cars and software-driven manufacturing are introduced. More details are provided in Chapter 2 of this work.

Software-defined or software-driven cars are vehicles where software plays a central role in their operation, customization, and delivery of services (Slama et al., 2023). Unlike traditional cars, where mechanical systems were paramount, these cars rely heavily on software from engine management to in-car entertainment.

As new automotive standards influence test process modeling and execution, they are also affected by software driven cars as updates w.r.t. to diagnostics need to be integrated into the car along the OEM, i.e., in development, manufacturing, and after-sales processes.

Software-defined manufacturing is a concept in which software and hardware are decoupled for modernized manufacturing processes (L. Xu et al., 2021). This is crucial as there is more variety of products, shorter production times are required to save money, and the hardware-software-connection is accessible through more sensors and data connectors.

So based on the identified challenges, this thesis aims to solve the following research questions.

RQ1: How can test quality for software-defined customer functions be improved by intelligent methods during car ramp-up phases?

RQ2: How can in-car orchestration of tests improve software-defined car manufacturing concerning flexibility?

RQ3: How can executable business process models improve automotive testing processes for software-defined cars in OEM car productions?

Therefore, the aim of the thesis is to investigate the following hypotheses *H1* to *H4*.

One aspect of test quality in car ramp-up phases is especially achieved by selecting the right tests, such that a maximal number of functional relationships though signals in the electrical system occurs. Anomalies can then be detected, and the fault recovery process starts. To this end, clustering can help to organize unlabeled data into dissimilarity groups (D. Xu & Tian, 2015). It is open how the method for a car ramp-up phase looks like.

Hypothesis H1*:* An unsupervised machine learning method as clustering can improve test quality of automotive testing processes during car ramp-up.

Online orchestration of process steps is investigated in automotive software systems, e.g., by (Hu et al., 2023). However, there is no study for the testing and diagnostics processes in software-defined manufacturing.

***Hypothesis H2**:* The concept on in-car orchestration of tests supports flexible car manufacturing w.r.t. software-defined cars and improves the existing scheduling process.

Cross-industry approaches are one characteristic of software-defined manufacturing lines (Oechsle et al., 2023). They can be combined with business process management with execution semantics (Schäffer et al., 2021).

***Hypothesis H3**:* Applying business process management with execution semantic approaches can bridge operational technology (OT) and information technology (IT) for automotive testing processes for software-defined cars in manufacturing, to enable continuous engineering processes without vendor-specific software frameworks.

The approach which follows the hypothesis *H1* to *H3* should be integrated into the OEM architecture for the interdisciplinary teams.

***Hypothesis H4**:* The test system architecture supports the software-driven car architecture and its central components to integrate the engineering processes of software-defined cars by interdisciplinary teams, which are composed of engineers, IT architects, computer scientists, workers and business analysts.

1.2. Structure of the Dissertation

This publication-based dissertation follows the design science research approach and is structured as follows. In Chapter 2, the background and state of the art of this thesis are described. In Chapter 3, the research steps that were performed are presented. The six included papers summarizing the findings of this work are introduced in Chapter 4. In Chapter 5, the overall research contribution is discussed, and in Chapter 6, the directions for future research studies are described. The six included papers are listed in the Appendix.

2. Field of Investigation and Related Work

In this Chapter, the required background of this work is presented to classify related works in the following. This is necessary to identify the research gap tackled in this thesis.

Industrial automation is the fundamental background to be considered. In industrial automation, the management of material, energy, or information states within technical processes are mechanized (Lauber & Göhner, 1999). The corresponding processes are comprised of various methods, which can be categorized into continuous, event-discrete, and object-related types (Lauber & Göhner, 1999).

Event-discrete methods are defined by a sequence of distinct, sequential states, and can be illustrated using Petri nets or flowcharts. Object-related methods focus on the transformation, transportation, or storage of individual items, such as manufactured components, and can be modeled using simulations, state machines, or Petri nets.

Technical processes operate within technical systems and can be automated using process automation systems (Lauber & Göhner, 1999). Such a system consists of three interconnected subsystems: the controlled technical system that executes the process, a computing and communication system, and the operating personnel. Sensors and actuators are used to interact with the technical process. Sensors monitor the system by measuring physical quantities and converting them into electrical or optical signals, while actuators adjust these quantities to manage the system (Lauber & Göhner, 1999). Signals from sensors and actuators are exchanged with the supervisory computing and communication system, which supervises and directs the technical process. Human operators typically interact with this system via a human-machine interface (HMI) to monitor the process and analyse errors.

The further subject of this thesis lies in the special case of automotive manufacturing systems and deals with software engineering aspects. Their interconnection is important for Weyrich (2024) as well. In the next subsection, the basis on manufacturing is given.

2.1. Software-defined Manufacturing

Software-defined manufacturing, also known as software-driven manufacturing, is an approach for production that leverages the power of software to increase flexibility, efficiency, and customization in manufacturing processes (Thames & Schaefer, 2016). Soft-

ware-defined manufacturing includes highly versatile production systems with data continuity along the life cycle and at the same time, from the sensor to the cloud for the OEM and supplier industry (Vogel-Heuser et al., 2023). The concept is studied, e.g., in the project SDM4FZI (German Federal Ministry for Economic Affairs and Climate Action, 2024a).

In traditional manufacturing, machines are typically designed for specific tasks, and reconfiguring a production line for a new product can be time-consuming and costly. In contrast, software-defined manufacturing relies on adaptable machinery that can be quickly reprogrammed to perform different functions. This is possible using software that controls and optimizes the machinery and production processes.

The backbone of software-driven manufacturing is the use of digital twins, which are virtual replicas of physical assets. Another critical element is the Internet of Things (IoT), where sensors and connected devices collect data from the manufacturing floor. Software systems then use this data to monitor performance, predict maintenance needs, and optimize production in real time. Machine learning algorithms can analyze this data to identify patterns and improve processes over time. Software-defined manufacturing also allows for greater customization. Since software controls the production, switching between making different products or variants is more accessible without extensive downtime. This supports the trend towards mass customization, where customers expect products tailored to their specific needs. Software-defined manufacturing reacts to volatile markets as physical production is supervised by software configuration, planning, scheduling, and control. Engineering tasks can be integrated into the car for the lifecycle (see Figure 2). There is already work on the impact of entire life cycles in software-defined manufacturing to increase the efficiency in planning and operating of production systems (Sebastian Behrendt et al., 2023). However, it is not discussed how the manufacturing work changes concretely. Moreover, a service-oriented architecture is not specified for the automotive domain, which is fundamental for software-defined manufacturing.

Figure 2: One idea behind software-defined manufacturing: Concepts from informatics are integrated into manufacturing lines (Vector Informatik GmbH, 2024)

2.2. Software-defined Cars

Future automotive software-defined manufacturing systems ideally produce a software-defined car. It is a vision of software-defined automotive development, where high-performance computers (HPCs) have a central role (Böhlen et al., 2023). In a software-defined car, the car's customer features and functions are primarily controlled by software applications running on in-car HPCs (Liu et al., 2022). This allows for a high degree of flexibility and upgradability. E.g., new functionalities can be added, or existing ones can be improved through over-the-air (OTA) updates, which are studied, e.g., in the SofDCar project (German Federal Ministry for Economic Affairs and Climate Action, 2024b).

The driving experience in software-driven cars is enhanced by advanced driver-assistance systems (ADAS), which build on software algorithms to provide features like adaptive cruise control. Moreover, software-driven cars are connected to the internet, enabling them to receive real-time traffic and updates. This connectivity also allows for remote diagnostics and predictive maintenance.

A prototyping platform for connected-car software is presented in the work of (Haberle et al., 2015). It underlines the possibility of integration cars into existing enterprise information systems and their concepts.

The evolution is also represented in car topologies as shown in Figure 3. Electrical control units (ECUs) are connected with HPCs to realize the software bound functions. For update the access should be via the cloud.

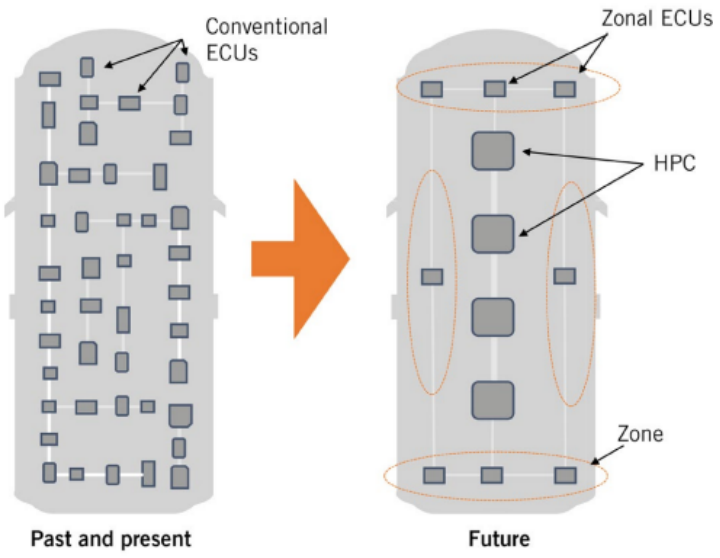


Figure 3: Car topology evolution towards software-defined cars (Windpassinger, 2022)

2.3. Automotive Test Management

When a software-defined car processes through the software-defined manufacturing line, automotive testing comes into play to ensure the quality of the E/E functionality. That is why automotive test management is considered in the subsection.

Software development is guided by the V-model, which includes testing, as already introduced in the motivation of this work. There are also new and agile software development methods. Lohrasbinasab et al. (2020) present an overview on BizDevOps.

Automotive testing involves evaluating complete car, E/E components, and systems across lab settings, simulated environments, and actual driving conditions to verify their safety, dependability, and adherence to regulatory safety standards (Schäuffele & Zurawka, 2024). The device under test can be a fully or partially assembled car in the production or development phase and components on hardware-in-the-loop or software-in-the-loop test benches. An overview of the test phases during the automotive life cycle is shown in Figure 4. The tests relevant for this thesis are located at the OEM side and especially in the upper part. To execute test cases on a device under test at all, engineers must specify the test cases, and the necessary software and hardware must be planned and sorted in a coherent schedule. After faulty execution, there is a need for error analysis. There is ongoing automation potential for the process steps (Haas et al., 2021).

Testing standards are also adapted w.r.t. software-defined cars: the Service Oriented Vehicle Diagnostics (SOVD) project was launched at ASAM (2024b). Standardization aims to create a modern diagnostic interface that enables access to classic control units and emerging software-based systems. The aim is also to allow uniform access for remote, proximity, and in-vehicle diagnostics scenarios (see Figure 5). In the project, a REST API (Representational State Transfer Application Programming Interface) is used which is a web service interface from the theory of information technology based on the REST architectural paradigm (Roy Thomas Fielding, 2000).

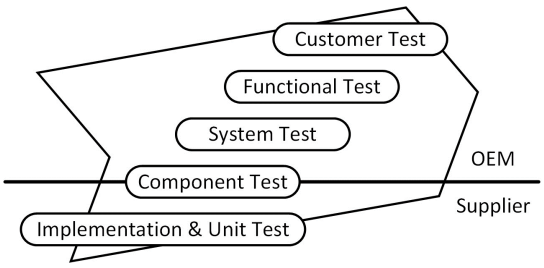


Figure 4: Overview on automotive testing phases, adopted from (S. König, Vogel-Heuser, Mäckel, & Schnittger, 2021)

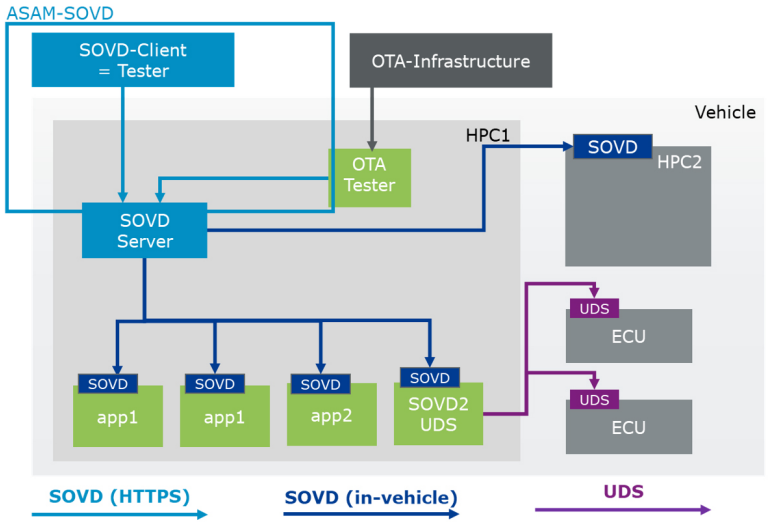


Figure 5: Architecture proposal for standardization via SOVD (ASAM e.V., 2024b)

It enables communication between different applications or systems over the HTTP protocol. Resources are addressed through unique URLs and manipulated through HTTP methods such as GET, POST, PUT, or DELETE. REST APIs are platform-independent and can be used by multiple programming languages and systems. Advantages are platform independence, scalability, flexibility, caching, and simplicity. Disadvantages are security, lack of standardization, overhead, and limited functionality. There is no general standardization for REST APIs, making ensuring compatibility between different systems challenging (Pautasso et al., 2008). However, OpenAPI builds a specification for building APIs (OpenAPI Initiative, 2022). OpenAPI is widely used in the industry and is supported by many tools and frameworks, e.g., Swagger UI. It is a powerful tool for building and documenting APIs and can help to improve collaboration between developers and API consumers (Bogner et al., 2023).

Automotive testing workflows can be represented with the standard OTX. It is an ISO standard for exchanging vehicle diagnostic and test data (ISO, 2011). It was developed to allow for the standardized exchange of complex test sequences and diagnostic algorithms between different systems and components in the automotive industry. It depends on the protocol Unified Diagnostic Services (UDS) and not on HTTP (ISO, 2020). OTX is specified for classic vehicle diagnostics under the UDS diagnostic standard and provides a standardized way to describe diagnostic test sequences. This standardization was crucial in the automotive industry, where manufacturers and suppliers often need to share diagnostic data and procedures.

Nevertheless, software-intensive systems in the automotive domain require optimization regarding costs and IT technology (Broy et al., 2006).

2.4. Modeling and Executing Business Processes using Business Process Model and Notation (BPMN)

To embed automotive testing into a software-defined car manufacturing line, the concept of business process modeling and execution is investigated in the following. There are already attempts to integrate BPMN into the automotive manufacturing domain, e.g., (Dubani et al., 2010; Fernandes et al., 2021).

In this subsection, special attention is paid to business process modeling and automation. Business Process Model and Notation (BPMN) is a standardized approach for depicting business processes in a graphical form that is understandable to business professionals and IT experts (Object Management Group, 2011). BPMN was developed to provide a common language for clearly and consistently describing complex business processes. Fundamental aspects of BPMN include:

1. Standardized Symbols: BPMN uses a set of standardized symbols and notations to represent various aspects of a business process. This includes activities, decisions, events or flow directions.

2. Process Visualization: By graphically representing processes, BPMN helps visualize and understand the workflows within an organization. This aids in identifying inefficiencies, bottlenecks, and opportunities for improvement.

3. Communication and Collaboration: BPMN serves as a common language, facilitating communication among different stakeholders such as business analysts, process managers, and IT developers, especially in the design, implementation, and analysis of business processes.

4. Process Optimization and Automation: BPMN is used for documentation and optimizing business processes. This is part of business process management (van der Aalst, 2013).

5. Flexibility and Scalability: BPMN can depict simple and highly complex business processes. It is flexible enough to be used across various industries and organizational sizes.

Literature exists how to model with BPMN: Bruce Silver presents the BPMN method and a style guideline for modeling (Silver, 2009). Mendling et al. (2010) discuss seven process modeling guidelines. Zur Muehlen and Recker (2019) discuss the limit of modeling applications.

In the context of systems engineering and mechatronics, the standard SysML (OMG, 2023) exists and can be integrated in production systems as well, e.g., as discussed by Vogel-Heuser et al. (2024).

As for test scheduling use cases decision modeling is relevant as well, the notation Decision Model and Notation (DMN) can be applied. The DMN standard is used to implement business rules (Object Management Group, 2020). Corea et al. (2024) have a look at DMN modeling errors in an SAP Signavio case study. More scheduling approaches will be investigated in the next subchapter.

So, BPMN is the de facto standard for workflow modeling, even though it has been subject to critical examination (Dumas & Pfahl, 2016), and other business process notations are also available. Marius et al. provide an insightful analysis of different modeling paradigms and their transformations (Marius et al., 2024), while Kopp et al. (2009) offer a comprehensive overview of alternative notations.

Executable BPMN is an enhancement of the BPMN 2.0 standard and turns a process diagram into a working application. It does this by enriching the standard BPMN elements with execution semantics (Leymann, 2010). These semantics define the technical execution details the standard BPMN does not cover, such as Service Task method calls, User Task forms, Script Task scripts, and Message Event payloads (Lübke & Pautasso, 2019). This allows the interpretation of the models in a way that a system can enact them.

Executable BPMN models are directly deployable to a process automation engine, which interprets the model and manages the state of process instances as they are executed (Stiehl et al., 2019). This engine can handle User Task Management, Service Task execution, sending and receiving messages, and managing timers. This approach provides a seamless transition from process design to execution, ensuring that the automated process faithfully reflects the designed process.

Executable BPMN supports the concept of “round-trip engineering”, meaning that the same model is used for design and execution without translating the model into another form for implementation. This greatly improves alignment between business requirements and IT implementation, reduces the risk of errors in translating business processes into code, and accelerates the delivery of process automation (Krüger et al., 2020).

Table 2: Comparison between OTX and executable BPMN in the automotive domain

Criteria	**OTX**	**Executable BPMN**
Purpose & Focus	It is designed for the automotive industry to standardize the description and exchange of vehicle diagnostic and test sequences. OTX focuses on defining complex diagnostic procedures and tests for vehicles.	BPMN is a graphical notation for specifying business processes in a workflow. Executable BPMN extends this by allowing these processes to be directly executed on a BPMN engine. It is used to automate and manage business processes across industries.
Application Domain	Primarily used in the automotive sector for vehicle diagnostics, testing, and maintenance procedures.	Used across various industries for modeling and automating business processes. It's not limited to a specific domain and is applied for business process management.
Complexity & Technical Focus	Deals with technical details specific to automotive diagnostics, such as interacting with vehicle control units, interpreting sensor data, and executing diagnostic routines.	Focuses on the workflow of business processes. It deals with the sequence of activities, decision points, and data flow, but it is typically less concerned with the deep technical specifics of a particular industry like automotive diagnostics.
Users	Mainly used by automotive engineers, technicians, and diagnostic tool developers.	Used by business analysts, process designers, and IT professionals, who are involved in the design, implementation, and management of business processes.
Execution Environment	Executed in specialized automotive diagnostic tools and environments.	Runs on BPMN engines, business process management systems, and software platforms designed to execute and manage business processes.

Under this circumstance, it is important to adapt the concept of OTX in software-defined car manufacturing and compare it with BPMN. The difference between OTX in the automotive domain and executable BPMN lies in their primary purposes, usage, and the contexts in which they are applied (see Table 2 for more details). Moreover, it is no new aspect, as BPMN is already investigated with the automotive industry, e.g., Dubani et al. (2010) present a design framework for business process modeling in the automotive industry. In summary, while OTX and executable BPMN provide standards for describing and executing sequences, they cater to different needs and are used in distinctly different contexts. OTX is specific to the automotive industry for diagnostics and testing, whereas executable BPMN is a broader tool for business process management across various industries and can bridge gaps to the concepts of software-defined cars in manufacturing.

2.5. Mathematical Methods in Flexible Production Planning & Scheduling

The last background is paid to the concept of flexible process management in terms of test planning and scheduling. This is especially relevant as variant management is an essential challenge in software-defined car manufacturing (see Challenge 2). In this work, to enable flexible business process, suitable mathematical methods are introduced. More details are given in the respective papers.

Production planning and scheduling are critical aspects of operations management aimed at making the manufacturing of products as efficient and effective as possible. An overview on scheduling theory and applications is provided by Blazewicz et al. (2007).

Mixed Integer Programming includes the ability to make integer decisions, which helps schedule problems where decisions are often binary, e.g., if a machine is operational or not. Mixed Integer Programming can solve complex scheduling problems, establish production sequences, and minimize setup times.

Constraint programming is a technique that defines a set of constraints that must be satisfied and searches for solutions that meet these constraints. The technique is beneficial for scheduling problems where the relationships between tasks are complex.

Online and offline scheduling are two approaches to allocating resources and organizing tasks within a production or computing environment.

Offline scheduling refers to planning where all the information about the tasks, including arrival times, processing times, and resource requirements, is known in advance. In this scenario, a scheduler can optimize the task sequence before execution begins, aiming for

an ideal allocation of resources. The advantage of offline scheduling is that it allows for thorough optimization, potentially leading to more efficient use of resources and time. However, its main disadvantage is the need for more flexibility; it can only accommodate unexpected changes or new tasks with a complete rescheduling.

Moreover, online scheduling deals with tasks as they arrive without prior knowledge of future events. This method is dynamic and adaptable, handling unpredictability and real-time changes. The primary advantage of online scheduling is its responsiveness to unforeseen events, making it suitable for environments where conditions change rapidly. The downside is that, due to the lack of complete information, it may only sometimes achieve the most optimal scheduling, potentially leading to less efficient resource utilization than offline scheduling.

All in all, machine learning techniques are becoming increasingly important, with their ability to discern patterns in historical data and forecast future trends (Minguillon & Lanza, 2019).

Having introduced the relevant background of this work on software-defined car manufacturing, modeling, and execution business processes with BPMN and methods for flexible process management in software-defined manufacturing, relevant requirements for this work are derived in the next subsection.

2.6. Test Automation for Software-defined Cars under new Standards

This section defines the requirements of test automation during software-defined car manufacturing. They are derived from the requirements of Section 1.1. and grouped in four categories. In Table 3, the rating scheme of the requirements is presented, which is fundamental for the further research activities in this work.

2.6.1. Requirements for Reducing Complexity in Test Planning

The complexity of car test planning in the development process is high (S. König, Vogel-Heuser, Mäckel, & Schnittger, 2021). In the car startup phase, cars should contain various features the customer can experience ($R_{Complexity}$). At the same time, a service should be provided, that increases the quality of the entire test process ($R_{Quality}$). Testing of a software release is dependent on a limited number of hardware and software prototypes which are assembled within different phases based on production scheduling dates. Executing a test case on a prototype depends on combined car equipment codes. It is ensured that functional relationships in the electrical system are tested and can be diagnosed. Test

execution should be performed on prototypes with the greatest difference of all equipment codes. Prototypes that are assembled first should be tested first. Anomalies can be detected, and a fault recovery process can be started. A high test coverage in each test is preferable.

2.6.2. Requirements for Managing Variants in Test Scheduling

The requirement $R_{Variants}$ is split up into two components. Test management in terms of scheduling need to be optimized ($R_{Optimize}$). The relative requirements are already discussed in the work of (S. König, Reihn, et al., 2023). As software-defined cars provide a high-performance computing unit, online and in-car orchestration can flexibly adapt testing routines ($R_{Orchestration}$). A test $i \in I$, from the set of all tests I, has a run time $\{t_i\}_{i\in I}$, $t_i \in \mathbb{R}$ >0. A test can have a direct predecessor $\{pre_i\}_{i\in I}$, $pre_i \in I$. A test i ∈ *I* may require a set of predecessors $\{P_i\}_{i\in I}$, $P_i \subset$ I , which must terminate successfully before i starts. A test i ∈ I may require a set of mutual exclusive tests $\{M_i\}_{i\in I}$,, $M_i \subset I$, which may not run parallel to i. A test i ∈ I may consume $\{r_{i,j}\}_{i\in I, j\in J}$, $r_{i,j} \in$ [0, 100] of automotive bus resource j ∈ J. A test i ∈ *I* may require or change a status of an object *l* ∈ L , i.e., $\{c_{i,l}\}_{i\in I, l\in L}$,, $c_{i,l} \in \{require_{on}, require_{off}, turn_{on}, turn_{off}, any\}$. An example is the presence of a worker at a test location.

The operation of the scheduling program within the car network relies on the car's HPC architecture. All relevant scheduling software must be tailored to this HPC architecture. Diagnostic tests may be time-sensitive and require low latency. Execution of diagnostic tests might be time critical and require low latency. Thus, the problem deals with soft and hard real-time assumptions. The computational scheduling process must take << 1s so that rescheduling takes no longer than the planned execution time of a test of, e.g., 1s (S. König, Vogel-Heuser, Wilch, et al., 2023). CPU and RAM load on the HPC, which is stressed by the scheduling program, should be low. Diagnostic test schedules should be created initially and only adjusted by rescheduling if there is a defined disturbance. Diagnostic tasks might also include human interaction in sub-tasks. Variability in the flexible execution of the tests has to be controlled by checking the schedule status and adjusting it if necessary, as soon as disturbances are present (S. König, Vogel-Heuser, Karg, et al., 2023). Tests that are already in progress should not be interrupted, so the schedule program is non-preemptive.

2.6.3. Requirements for Continuous Test Process Automation throughout a Car Life Cycle

In a software-defined car in a digital factory, the continuity of processes is critical ($R_{Continuous}$). Therefore, there are various requirements for test process management (S. König, Vogel-Heuser, Fieg, et al., 2021). The test and diagnostics workflow must be modelled with the help of its textual specification and its programming source code as well as with the help of specialist knowledge ($R_{Diagnostics}$).

The requirement R_{Low} demands low training periods for new employees. This is especially important as car diagnostics is complex, and experts are hard to acquire. To gain acceptance, the modeling of production testing workflows should be faster, clearer, simpler, and easier to maintain. Involving engineers from the outset is crucial to prevent the rejection of the modeling approach.

A programmer manages the variant handling of individual production workflows for manufacturing, utilizing the source code and adhering to various test specifications. Building variants of workflows depends on experts' knowledge and textual programming for each car line ($R_{Scheduling}$). This work demands an approach that allows sequential or parallel scheduling of test procedures.

A central requirement in software-defined manufacturing demands no programming knowledge for schedule descriptions (S. König, Vogel-Heuser, Fieg, et al., 2021). The process of modelling production testing workflows could take more time than traditional source code implementation. The modelling must occur with the help of predefined, detailed, modelled tasks at hierarchical level. Technology should be used where code programming is no longer required and visual representations are preferred ($R_{No-code}$).

A consistent solution demands the execution of sequence specifications such that specification and implementation are not separated ($R_{Seamless}$).

In manufacturing shopfloor work, a simple analysis of failed procedures to find causes is essential ($R_{Analysis}$). Workers demand an easily understandable analysis method, e.g., with graphical visualizations.

The requirement $R_{Manufact}$ represents the automotive manufacturing domain with limitations by assembly line production, i.e., cycle times, high volumes, and few faults.

2.6.4. Requirements for the System Architecture in Software-defined Cars

The approach of this work should be implemented in an automotive environment (R_{System}). The requirement R_{IT} supports the IT integration in large enterprise scopes with OEMs and suppliers, i.e., backend integration support (S. König et al., 2024).

The requirement R_{Re-Use} in software-defined manufacturing demands the support of a cross-industry approach (Oechsle et al., 2023). This means the industry can benefit from solutions proven to help, i.e., even in industries outside the automotive domain.

A software-defined car supports generic approaches for trend impact containment ($R_{SofDCar}$). Information systems solutions, e.g., web technology paradigm, especially come into play in vehicle diagnostics.

A unique role plays the integration into the vehicle network. The requirements demands a solution integrated into the vehicle network, i.e., it comes with applicable standards such as AUTOSAR (Staron, 2017) and low CPU and RAM consumption (R_{In-Car}). In car production settings, production-related APIs are accessible only within the production environment and must meet strict latency requirements in the millisecond range, such as for window scaling or communication with test benches. Some latency requirements are mandatory for safety or certification purposes. Therefore, the test execution system must be integrated into each individual car to perform the specific diagnostic processes required for that car within the required latency limits.

Thus, instead of a single workflow engine executing many instances of the same process model, multiple workflow engines in the set of all cars each execute a single instance of the same model.

The approach should include an API Management for automotive standards with RESTful HTTP API, e.g., SOVD (requirement R_{API}). From the standard SOVD, it can be deduced that web services based on RESTful HTTP APIs have an increasingly important role in car networks.

2.6.5. Focus of the Thesis

The focus of this work is on the fundamental feasibility of the derived artifacts in the automotive industry rather than on their efficient programming. This does not imply that the software artifacts are designed to be inefficient; rather, further optimization, such as more

efficient scheduling algorithms, could potentially yield even greater time savings. Comprehensive user acceptance studies are also beyond the scope of this work, based on the premise that an effective and functional solution will naturally achieve user adoption.

Table 3: Rating scheme of requirements to evaluate research activities in this work

$R_{Complexity}$ - Reducing Complexity in Test Planning during Car Ramp-Up Phase	
+	Approach considering: reduced complexity in test planning and improved test quality during car ramp-up phase
°	Approach with limited consideration of: reduced complexity in test planning and improved test quality during car ramp-up phase
-	No approach
$R_{Variants}$ - Managing Variants in Test Scheduling	
+	Approach considering: online scheduling of test and diagnosis procedures on an in-car HPC and improved schedule run-time
°	Approach with limited consideration of: online scheduling of test and diagnosis procedures on an in-car HPC and improved schedule run-time
-	No approach
$R_{Continuous}$ - Continuous Test Process Management throughout Software-defined Car Life Cycle	
+	Support of graphical modeling and execution of test and diagnosis use cases for software-defined cars
°	Limited support of graphical modeling and execution of test and diagnosis use cases for software-defined cars, e.g., lack in abstraction, interpretation and executability
-	No support
R_{System} - System Architecture in Software-defined Cars	
+	Including quantitative and qualitative studies for integration into software-defined car backend
°	Limited quantitative and qualitative studies for integration into software-defined car backend
-	No studies

2.7. Related Work on Test Automation in Software-defined Car Manufacturing and the Integration of Executable BPMN

The related work for this thesis is investigated w.r.t. the requirements of the previous subsection in three clusters: software-defined cars, software-defined manufacturing with a focus on the flexibility of testing processes, and executable BPMN for test specification and execution. As an overview, the relevant related work is clustered in Figure 6. The classification of this thesis into the related work is summarized in Table 4. Building on these results, the research gap is identified. This gap is later used as driver of the research contributions of this thesis.

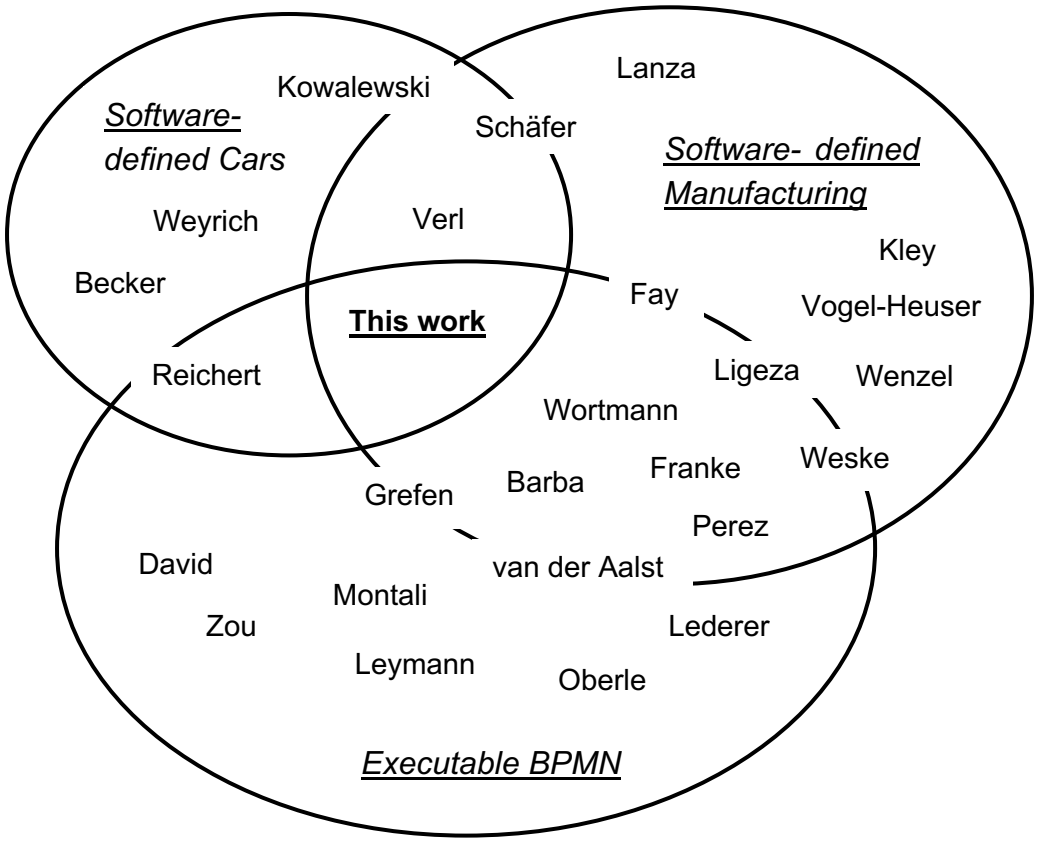


Figure 6: Overview of relevant State-of-the-Art contributions w.r.t. working groups, their field of contribution, and the identified research gap, which is highlighted in grey.

The requirement $R_{Complexity}$ in software-defined car manufacturing

The requirement $R_{Complexity}$ belongs to the area of software-defined car manufacturing. An overview on clustering algorithms for data-based applications is presented by Jain and Dubes (1988). A hierarchical weighted clustering algorithm for the optimization of mobile hybrid networks is presented by Brust et al. (Brust et al., 2007). A clustering approach to

improve test case prioritization is discussed by Carlson et al. (2011). There is no approach that fulfills the requirement $R_{Quality}$.

The requirement $R_{Variants}$ in software-defined car manufacturing

Following the definition of requirement $R_{Variants}$ in Section 2.6.2., there is an intersection of software-defined cars and manufacturing within testing. Automotive testing processes are improved by production scheduling methods in the study of Stütz et al. (2023). The industrial application is automotive as in this work, however, details on the concrete mathematical models and conditions are missing.

The application of integer difference logic to handle automotive hardware and software co-configurations is studied as well (Jost & Sinz, 2024). The authors investigate how to address software and hardware compatibility issues in the automotive industry by using satisfiability modulo theory. This allows for considering version ranges in a compatibility description. There is no content w.r.t. the requirement $R_{Orchestration}$.

Bartels and Zimmermann (2009) study the scheduling of tests in automotive research and development projects. This paper does not consider the requirements for manufacturing lines as requirement $R_{Orchestration}$.

Bronner et al. (2018) present a production planning approach to mount all-electric vehicles within existing assembly lines. This is no work w.r.t. $R_{Orchestration}$.

Dörmer et al. (2015) show the application of master production scheduling and sequencing at mixed-model assembly lines in the automotive domain. The requirement $R_{Orchestration}$ is not considered. This also corresponds to the work of Huang and Zhou (2018). In their paper, they provide a SAT algorithm for complex job-shop scheduling.

A model-based approach for simultaneous scheduling of machines and vehicles in flexible manufacturing systems is presented by Lin et al. (2018). This is no work w.r.t. $R_{Orchestration}$. The application of scheduling in vehicle repair processes is studied by Pilar et al. (2023). The requirement $R_{Variants}$ is not fulfilled. Shi et al. (2017) conduct vehicle test scheduling analytically. The requirement $R_{Orchestration}$ is not fulfilled.

Witkowski et al. (2010) solve an flexible open-job shop scheduling problem with the mathematical methods of Greedy Randomized Adaptive Search Procedures (GRASP) and simulated annealing. This work does not satisfy $R_{Orchestration}$.

Constraint-based planning and scheduling is applied for the optimized management of business processes in the work of Barba (2013). The thesis is based on theoretical work to planning and scheduling for business processes. It includes an execution environment named ConDec-R to execute declarative process models with resources. It optimizes business processes in run-time and proposes an example of a repair problem. This work can be transferred to $R_{Variants}$, but not to $R_{Orchestration}$.

The requirements $R_{Continuous}$ in executable process models with BPMN

The requirement $R_{Continuous}$ reflects automotive-specific conditions for testing processes and continuous engineering.

Schäffer et al. (2021) present a concept for process-driven applications to build a bridge between the two domains of Business Process Modeling and Software Engineering. This concept focuses on the modeling and execution of business processes with the help of BPMN. The work does not discuss the complexity of test scheduling via explicitly presented mathematical models as required by $R_{Variants}$.

Giacomo et al. (2015) present ways to include declarative process models into BPMN. This is important if rule-based decisions should be integrated into the workflow of BPMN process models. However, in the setting of this work, declarative process models are described via rule-based optimization models following $R_{Variants}$. This is relevant for the work of Peinl and Perak (2019) as well, who apply BPMN and DMN for customization of manufacturing execution systems. Valencia-Para et al. (2019) have a look at the notation DMN for the process of data quality measurement and assessment. However, for complex processes, this work requires an automation method.

The work of Pesic is relevant regarding declarative modeling and constraint-based workflows management systems (Pesic, 2008). Pesic & van der Aalst (2006) present a declarative approach for flexible business process management in organizations and ConDec as respective language. This supports the contributor's work to find for a non-imperative test automation solution. However, the authors do not fully support $R_{Continuous}$ in their work.

Mülle et al. (2019) present a data-flow verification scheme for business processes.

Nguyen et al. (2021) present a concept for automated generation of tests for self-driving cars. Nevertheless, it corresponds to the upper part of software testing in Figure 4.

Lübke (2024) presents a BPMN profile for test case execution visualization. This may be interesting for further studies in $R_{Continuous}$ with focus on worker user interfaces. But this is no related work for this thesis.

David et al. (2023) discuss the workflow engines in enterprise applications in the context of business process management. They have no focus on the automotive domain.

Funk (2023) creates a low-code business process execution platform with the language Python, the notation BPMN and DMN. The author presents BPMN as low-code language and SpiffWorkflow as an implementation. The provided external services are done via BPMN Script Tasks, not via the Service Task.

The concept of subject-oriented business process management is covered in the book of Elstermann et al. (2023). On the domain of subject-oriented business process management, Piller (2023) evaluate BPMN 2.0. The idea includes that actors as subjects form a processor orchestration. It is a more general concept that could also include automotive scenarios.

Valeras et al. (2022) model and execute Internet of Things-enhanced business processes through BPMN and microservices.

Poss and Schönig (2023) present an approach towards location-aware business process execution.

Geiger et al. (2018) present a BPMN 2.0 guideline for implementation.

Malekan et al. (2018) study the robust execution of BPMN process diagrams.

Kirikkayis et al. (2022a) discuss a framework called IoTDM4BPMN to handle Internet of Things-driven business rules with the application of BPMN and DMN. They consider modeling and executing BPMN processes of low-level and high-level data. They also consider extending the standard BPMN elements for the IoT processes (Kirikkayis et al., 2022b). Finally, they present BPMNE4IoT as a framework for modeling, executing and monitoring IoT-driven processes (Kirikkayis et al., 2023). This summary has a view on logging as well and formulates a domain independent solution. Nevertheless, they have no concrete application to test planning and scheduling. They have no details on the concrete execution semantics. However, they have a user study with participants from university and industry to include different user groups for general acceptance.

Ochoa et al. present an architecture for managing architecture-as-a-service based business processes including a template for REST services (Ochoa, Larrinaga, & Pérez, 2023). A context-aware workflow management architecture is provided by Ochoa, Legaristi et al. (2023) including a real-world evaluation study concerning robot operating systems.

In a wider sense of software-defined manufacturing, Nordemann and Toenjes (2022) discuss the usage of resilient BPMN over wireless.

The choreography of BPMN fragments for microservice composition is present by Ortiz et al (2022). The paper presents an approach to propagate changes in a BPMN workflow including multiple companies among the companies. This setting is required in a distributed executing setting, which is very different to the setting of a car manufacturing line.

Cloud-based process execution engines with the architectures and interfaces are studies by Mangler and Rinderle-Ma (2022).

Aziza et al. (2022) model and implement governmental information systems with BPMN.

A systematic literature review is conducted by Abouzid et al. (2022) to model Internet of Things-behaviour in supply chain business processes under the usage of BPMN.

Compagnucci et al. (2023) conduct a systematic literature review with the focus on Internet of Things-aware business process modeling views, requirements and notations.

The application of ontologies and BPMN to model and execute production processes with capabilities and skills is presented by Kocher et al. (2022). Their focus is on modeling skills and they also contribute to orchestration by executing process models. There is no focus on concrete production requirements in terms of vehicle diagnostics or flexible scheduling.

The concept of integrating robotic process automation into business process management is discussed by M. König et al. (2020). It includes a case study with an exemplary BPMN process model.

An integrated platform for business process management systems and robotic process automation is discussed in the same working group and published by Flechsig et al. (2022). They provide a high-level architecture of a business process management system and robot process automation platform. There is only descriptive information concerning the orchestrator and process engine.

A different concept, but the same notation as in (Vogel-Heuser et al., 2022), BPMN++, is discussed by Cheng et al. (2022) for the comprehensive modeling for industrial internet application.

Grambow et al. (2022) present a framework for context-aware process execution in Industry 4.0 processes. Bouroumi et al. (2022) present a contextual approach for business process execution. The papers study executable process models, however, the connection to automotive manufacturing is missing w.r.t. $\boldsymbol{R_{Continuous}}$ and $\boldsymbol{R_{System}}$.

The German paper of Kalach shows how to model REST-service compositions (Kalach, 2014). This is an implementation part of the covered related work in this thesis.

Pautasso (2011) shows BPMN for REST. This is a general aspect.

Erasmus et al. (2020) use business process models to specify manufacturing operations in terms of fragments. The same working group presents the HORSE framework which contains a reference architecture for cyber-physical systems in hybrid smart manufacturing (Traganos et al., 2021). The architecture is complex including robotic control, but it includes the BPMN standard in the Camunda open-source workflow management system to represent manufacturing process management systems. There is no focus on automotive testing control or integrated automated optimization solutions to handle flexible testing schedules.

A process-driven approach within the engineering domain is shown by Schäffer et al. (2021) with the combination of BPMN with process engines. This is related work, but the concrete implementation regarding the automotive requirement $\boldsymbol{R_{Continuous}}$ is missing.

Wiśniewski et al. (2018) provide a constraint-based composition of business process models including a formalization. There is no automotive case study or system architecture w.r.t. $\boldsymbol{R_{Continuous}}$ and $\boldsymbol{R_{System}}$.

Model-driven engineering of manufacturing automation software projects is discussed by Vogel-Heuser et al. (2014).

With focus on the automotive industry, Sivri and Krallmann (2015) present the concept on process-oriented knowledge management within the product change systems.

Ferreira et al. (2016) highlight a hybrid process management for collaboration in the automotive industry.

Brunninghaus & Rocker (2022) study the concept on low-code development in worker assistance systems that improve flexibility and adaptability. This generally contributes to the assumption of $\boldsymbol{R_{Continuous}}$.

Strategies for integration models into existing processes is discussed by Hasic et al. (2018) in the context of redesigning processes for decision-awareness.

The management of functional safety processes for automotive E/E architectures in integrated model-based development environments is studied by Adler et al. (2014). This thesis has no focus on security aspects as there are automotive functionalities for them.

The requirement $\boldsymbol{R_{System}}$ in software-defined car manufacturing

The following related works focus on the context of software-defined manufacturing. Vogel-Heuser et al. (2023) discuss methods, current approaches and applications in software-defined manufacturing.

The requirement R_{System} counts for the integration of the software-defined car in the manufacturing line during testing processes. There are mainly two groups: one that comes from software-defined cars and one from software-defined manufacturing.

Pesl et al. (2024) present large language models for service compositions in the context of software-defined cars. This approach covers the domain of software-defined cars.

Fuchß et al. (2023) present an expert survey on the use of informal models in the automotive industry for software-defined cars.

Mütsch et al. (2023) discuss current trends for software-defined cars. The paper explores the shift from traditional simulation methods to data-driven approaches in autonomous vehicle development, highlighting trends like generative models and neural networks for enhanced realism. It emphasizes ongoing challenges in realism, controllability, and validation, suggesting the need for standardized data formats and future research to address these issues.

Bickelhaupt et al. (2023) discuss challenges and opportunities of future vehicle diagnostics. Although the focus of this paper is particular on technical details regarding vehicle diagnostics and vehicular high-performance computers, they identify the need for more proven standardized IT methods adapted to the automotive area.

Stötzner et al. (2024) investigate variant management for deployment technologies for software-defined cars including a meta model. They propose an enhanced Variable Deployment Metamodel (VDMM) to better manage deployment variability across technologies such as Ansible and Terraform. Stötzner et al. address variability management in software deployment rather than in testing or car manufacturing processes.

Weller et al. (2023) present a DevOps approach for software-defined cars. They use the *Topology and Orchestration Specification of Cloud Applications* (OASIS, 2020) as deployment automation technology as it is an open standard, vendor-neutral and technology independent. The focus is on over-the-air updates.

Kriebel et al. (2018) investigate an improvement of model-based testing in automotive software engineering. While they are sure that model-based testing is important for automotive testing, they do not consider related manufacturing processes and new car technologies.

Wen et al. (2016) present a RESTful framework for Internet of things in modern manufacturing. This framework can be transferred to cars in manufacturing lines and is related work in terms of $\boldsymbol{R_{System}}$.

Brandstetter et al. (2023) present an approach to design a software-defined manufacturing systems for large-scale production of electrolyzes for hydrogen, which is based on a systematic literature review for the term of software-defined manufacturing system architectures. They consider the criteria: domain suitable, central access for distributed locations, greenfield suitable, service-oriented architecture and provision of product and production data. Nevertheless, the domain is not automotive.

As already mentioned in the Introduction, software-defined cloud manufacturing for industry 4.0 is discussed by Thames and Schaefer (2016). The authors transfer cloud technologies in software-defined manufacturing with the focus on firmware updates. They do not consider variants or configuration.

Behrendt et al. (2023) have a look at the entire life cycle of software-defined manufacturing with decoupled abstraction levels in machine, production system and production network. In another work, S. Behrendt et al. (2023) improve production systems in terms of flexibility and changeability through software-defined manufacturing by using the decoupled abstraction levels. The authors show how reconfigurations are no longer limited to technical constraints. The focus is not on the processes of planning and scheduling.

Neubauer et al. (2023) study a cloud-based evaluation platform for software-defined manufacturing. They describe a platform from product description to service deployment description and validate it in an OEM-like set up with focus on updating computing hardware and control systems.

Oechsle et al. (2023) present a real-time architecture for the use case of software-defined manufacturing. Walker et al. (2023) have a look at real-time execution models for container-based control applications in software-defined manufacturing. Both papers are related work for $\boldsymbol{R_{System}}$.

Kuhn et al. (2023) study process optimization in virtual commissioning during software-defined manufacturing, that contributes to $\boldsymbol{R_{System}}$ outside the automotive domain.

Dietrich et al. (2024) apply iterative planning for production system-wide optimization control loops in software-defined manufacturing. This makes no contribution to $R_{Variants}$.

Gelgfren et al. (2023) present a mixed-integer model for scheduling in body-in-white production systems for cars. So, their focus is on the automotive domain, but the underlying model is different because of a different production system.

Vogel-Heuser et al. (2022) present BPMN++ as a concept to support teams in cyber physical production system design and operation. This concept focuses on the interdisciplinary work with collaboration between different processes stages. However, there is no focus on executing BPMN process models or the automotive production systems.

In the following Table 4 the classification of the related work w.r.t. the requirements of Section 2.6 is presented. The rating scheme follows the definition of Table 2. None of the related works considers all four requirement groups.

Table 4: Classification of related work sorted into working groups following the rating scheme introduced in Table 2

Working Group & Relevant Publications \ Requirement		$\boldsymbol{R_{Complexity}}$	$\boldsymbol{R_{Variants}}$	$\boldsymbol{R_{Continuous}}$	$\boldsymbol{R_{System}}$
van der Aalst	(Pesic & van der Aalst, 2006), (Pesic, 2008)	-	+	+	-
Barba	(Barba, 2013)	-	+	+	°
Becker	(Weller et al., 2023), (Stötzner et al., 2024)	-	°	-	°

Fay	(Kocher et al., 2022)	-	-	+	°
Franke	(Schäffer et al., 2021)	-	-	+	°
Grefen	(Erasmus et al., 2020), (Traganos et al., 2021)	-	°	+	+
Györödi	(David et al., 2023)	-	-	°	+
Kowalewski	(Kampmann et al., 2020), (Kampmann et al., 2022)	-	°	-	°
Lanza	(S. Behrendt et al., 2023), (Sebastian Behrendt et al., 2023)	-	-	-	+
Lederer	(Elstermann et al., 2023)	-	-	+	+
Leymann	(Haberle et al., 2015)	-	-	-	°
Ligęza	(Wiśniewski et al., 2018)	-	°	+	-
Kestler	(Kuhn et al., 2023)	-	-	-	°
Kley	(Stütz et al., 2023)	-	+	-	-
Montali	(Giacomo et al., 2015)	-	°	°	-
Perez	(Ochoa, Larrinaga, & Pérez, 2023), (Ochoa, Legaristi, et al., 2023)	-	-	+	+
Oberle	(Brandstetter et al., 2023)	-	-	-	°
Reichert	(Kirikkayis et al., 2022a), (Kirikkayis et al., 2022b), (Kirikkayis et al., 2023)	-	-	+	+
Schäfer	(Egyed et al., 2023), (Schindewolf et al., 2024)	-	°	°	°
Verl	(Neubauer et al., 2023), (Oechsle et al., 2023), (Walker et al., 2023), (Dietrich et al., 2024)	-	-	°	+
Vogel-Heuser	(Vogel-Heuser et al., 2022), (Vogel-Heuser & Wimmer, 2023), (Vogel-Heuser et al., 2024)	°	-	°	-
Wenzel	(Gelgfren et al., 2023)	-	°	-	-
Weske	(M. König et al., 2020), (Flechsig et al., 2022)	-	°	+	+

Weyrich	(Bickelhaupt et al., 2023)	-	-	°	-
Wortmann	(Kriebel et al., 2018), (Jolak et al., 2020), (Fuchß et al., 2023), (Pfeiffer et al., 2024)	-	°	°	°
Zou	(Wen et al., 2016)	-	-	-	°

The research gap of this thesis work is formulated as follows:

Research gap

A model-driven approach for flexible car-individual test orchestration in software-defined manufacturing does not exist in the domain of software-defined cars. None of the surveyed approaches provides the integration of flexible testing into car production scheduling schemes for software-defined cars under the integration of a non-vendor-specific tool chain and IT-proven technologies.

3. Research Activities

In this thesis, three research activities are derived under the framework of design science research and are presented in the next sub-sections. The research activities include the quantitative analysis with mathematical solutions including software, the qualitative feasibility study with tool support and an overall evaluation with a focus group.

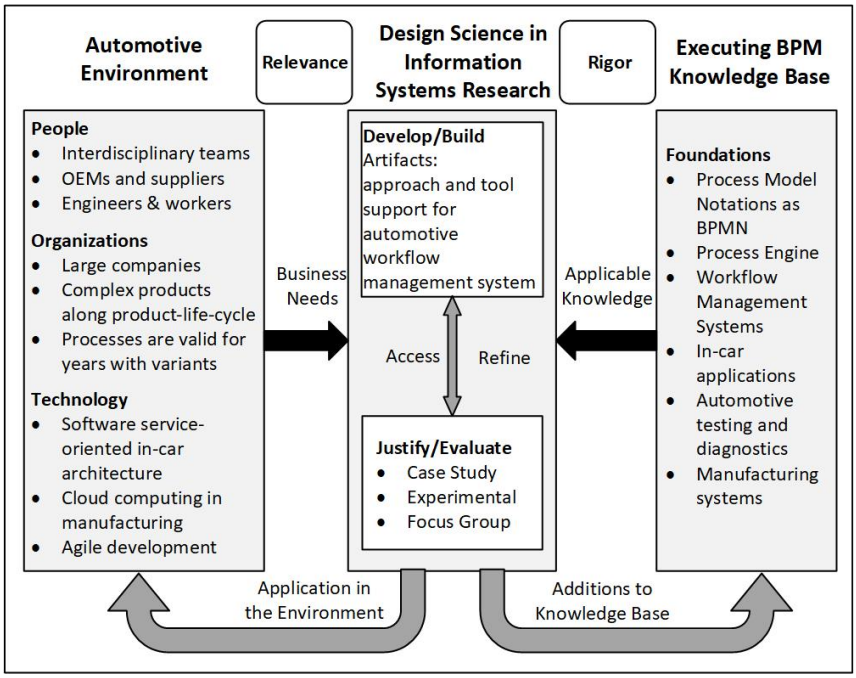


Figure 7: Research framework for this work (adapted from Hevner et al. (2004))

The research in this work is conducted following the method of design science research (vom Brocke et al, 2020; Dresch et al, 2014): This research method is used when the desired research goal is an artifact (Dresch et al, 2014). Design science research focuses on problem solving and this approach is suitable to the thesis work's business problem which is addressed by artifact creation in form of models and software prototypes.

Hevner et al. (2004) propose how design science in information systems research can be processed, including guidelines to be followed. The underlying research framework, including adaptive processing in this work, is therefore illustrated in Figure 7. Technical prototypes mapping the solution approach are initially created and adapted with three iteration cycles over six months. Finally, the approach is evaluated with an automotive focus group. The references between the requirements of Section 2.6 and the mentioned research activities are listed in Table 5.

Table 5: Evaluation scheme of requirements w.r.t. research activities (X: activity is performed; grey: not performed)

Requirements and Research Activity	Requirements	Qualitative Analysis	Quantitative Analysis	Tool Support
$\boldsymbol{R_{Complexity}}$	$R_{Quality}$	X		X
$\boldsymbol{R_{Variants}}$	$R_{Optimize}$		X	X
	$R_{Orchestration}$		X	
$\boldsymbol{R_{Continuous}}$	$R_{Diagnostics}$			X
	R_{Low}	X		X
	$R_{Scheduling}$		X	X
	$R_{No-code}$			X
	$R_{Seamless}$		X	
	$R_{Analysis}$			X
	$R_{Manufact}$	X		X
$\boldsymbol{R_{System}}$	R_{Re-Use}	X		X
	$R_{SofDCar}$	X		X
	R_{In-Car}		X	
	R_{API}	X		

All activities are performed at a German OEM on real industrial case-studies.

One part of the relevant notation was already introduced in Chapter 2 and so the artifacts refer to a test $i \in I$, with I the set of all tests, that will be modelled, planned, scheduled, and executed in the following. For this reason, more suitable mathematical notation elements will be introduced in the following.

3.1. Test Planning

For test planning in the car ramp-up phase, a hierarchical clustering method is presented which fulfills the business requirement $\boldsymbol{R_{Complexity}}$. Solutions with machine learning are plausible for unsupervised clustering (S. König, Vogel-Heuser, Mäckel, & Schnittger, 2021). The machine learning method is embedded into a business procedure. Furthermore, there is a quantitative quality assessment presented in comparison with a baseline model.

Quantitative Feasibility Study with OEM Internal Tool Support

The hierarchical clustering method is represented as procedure that is implemented as a pipeline in an OEM internal tool.

3.2. Test Scheduling

To build a scheduling method for the tests in I, a mathematical optimization model is presented. It is a mixed-integer linear system. Details on the basis method and its evaluation are presented in the work of (S. König, Reihn, et al., 2023).

Quantitative Analysis with OEM Internal Tool Support

The model is implemented in an OEM internal tool and validated via an industrial case study. For online scheduling a further industrial case study is performed (S. König, Vogel-Heuser, Karg, et al., 2023). With this study the effect of rescheduling can be validated for software-defined car manufacturing.

3.3. Executable BPMN for Improved Test Management

To represent the tests as business processes, the help of directed graphs is needed. The notation of the work of Leymann & Roller (2000) is useful.

Qualitative Feasibility Study with Open Available Tool Support

The concept of this thesis in presented in a transition table in Figure 8. The transition for specification and implementation is discussed in terms of the guideline for modeling production workflows in testing and diagnostics (S. König, Vogel-Heuser, Fieg, et al., 2021). Manual tasks should be automated as far as possible to master complexity.

Goal	Specification & Implementation	Planning	Scheduling	Execution of process models
User	Engineer	Engineer	Computer scientist	Software
Before	Text-based & source code	Manually	Manually	Runtime & worker-guided
Now	BPMN	Machine Learning based	Automated Orchestration	BPMN Process engine

Figure 8: Transition table for optimized test management, adopted from (S. König et al., 2024)

Quantitative Analysis with OEM Internal Tool Support

For the BPMN Process Execution Engine an integration test on a real automotive component is performed to prove the setup (S. König et al., 2024). Moreover, it is benchmarked against a state-of-the art process engine from the field of BPMN execution models (S. König, Vogel-Heuser, Wilch, et al., 2023).

Following the work of Yan et al. (2010), evaluation aspects for suitable BPMN tooling are main users, license, localization and function aspects as modeling support and model creation. For this thesis, especially the product aspects for the users and the modeling support in terms of BPMN elements are essential. The data flow should be made visible and the deployment of a BPMN model should be integrated for the automotive domain. These aspects are not given in the current tool support, so BPMN tool support was implemented independently by the OEM.

Overall Assessment via Focus Group Evaluation & OEM Internal Tool Support

The overall concept is assessed by a focus group evaluation with OEM experts. During the focus group evaluation an integration test took place with the BPMN modeling tool and process engine. Moreover, the participants were able to interact with the OEM internal BPMN tool support (S. König et al., 2024).

The respective architecture that combines the artifacts is shown in Figure 9.

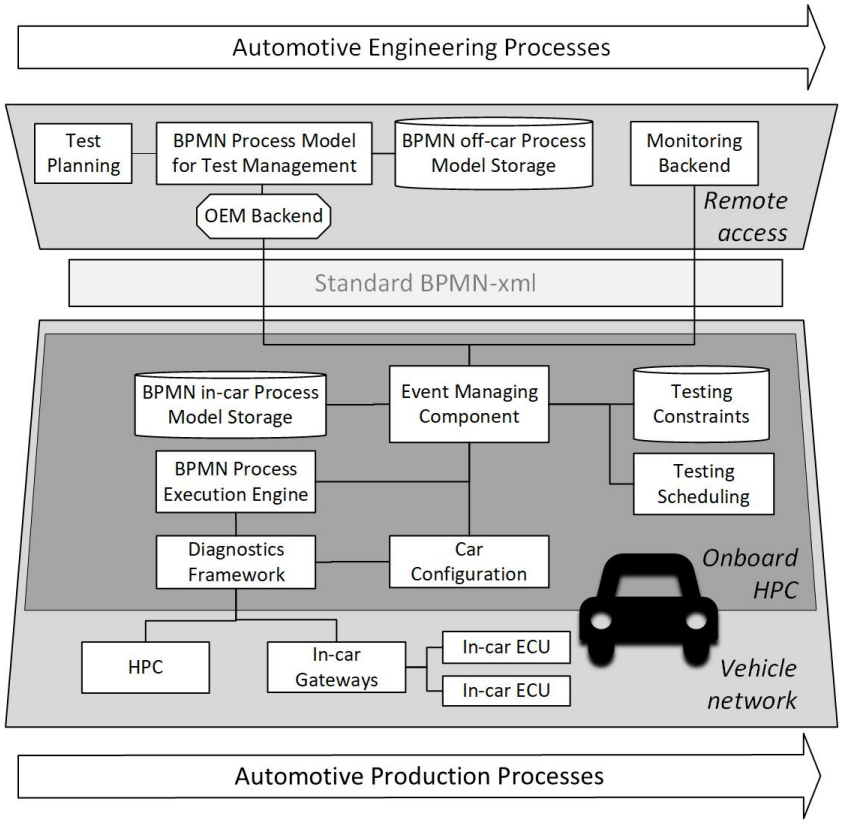


Figure 9: Architecture overview for optimized test management, adopted from the concept presented in (S. König et al., 2024)

4. Summary of the Publications

This chapter provides an overview of the six papers that form the verification basis for the approach of Chapter 3. The papers are listed in Scopus or Web of Science databases. The candidate's individual contribution of each paper is listed in Table 6. The full papers are attached in Appendix A.

Table 6: Overview of the candidate's individual contribution

	Conceptualization	**Data collection**	**Elaboration**	**Writing**
Paper1- [published]	60	55	70	65
Paper2- [published]	70	50	60	55
Paper3- [published]	35	40	60	60
Paper4- [published]	80	55	55	65
Paper5- [published]	50	40	40	75
Paper6- [published]	80	60	75	70

Notes:

- Conceptualization: Development and conceptual design of the thesis
- Data collection: Gathering, collection, acquisition, provision of data or software
- Elaboration: Analysis, evaluation or interpretation of data and conclusions
- Writing: Drafting of the paper or article manuscript

4.1. Improve Test Quality by Applying a Clustering-based Test Planning Procedure for Customer Experience Vehicle Functions (*Paper1*)

Simone König, Birgit Vogel-Heuser, Rainer Mäckel and Dominik Schnittger

Summary of the paper

Cars become more autonomous and connected with their environment. This functionality is implemented in so-called software-driven customer experience vehicle functions. There is a ramp-up phase during car development and before the start of car series manufacturing. Manufacturing lines are also in the ramp-up phase, but this paper focuses on the

ramp-up of the product car. The number of cars to be tested for software-driven customer experience vehicle functions is limited. Nevertheless, engineers want to achieve a high test coverage and quality for these functions. The state-of-the-art method for test planning is experience-based and manually conducted. This has been the starting point for applying unsupervised machine learning algorithms to achieve an automated test plan for customer functions.

The paper presents a machine learning based procedure using hierarchical clustering to automate test planning in the car ramp-up phase following the requirements of Section 2.6.1. It factors in prototype availability and equipment importance, aiming to maximize test efficiency. Implemented at an OEM, the method proved to enhance testing quality, showcasing the practicality of machine learning in improving customer experience with vehicle functions. With this work the manual work of test planning is interchanged by a mathematical, automated solution, that can easily assist the engineers to consider all conditions.

Individual contribution of the candidate

The contribution to this paper includes the conceptual design of the clustering procedure, gathering and collecting the data and literature from the manufacturing domain, and evaluating the procedure. Concerning these aspects, the candidate wrote the manuscript.

This subsection was published as *Paper1*: (S. König, Vogel-Heuser, Mäckel, & Schnittger, 2021)

> S. König, B. Vogel-Heuser, R. Mäckel and D. Schnittger, "Improve Test Quality by Applying a Clustering-based Test Planning Procedure for Customer Experience Vehicle Functions," 2021. 22nd IEEE International Conference on Industrial Technology (ICIT), Valencia, Spain, 2021, pp. 736-743, doi: 10.1109/ICIT46573.2021.9453674.

4.2. Modelling Production Workflows in Automotive Manufacturing (*Paper2*)

Simone König, Birgit Vogel-Heuser, Etienne Fieg, Michael Hahn and Oliver Kopp

Summary of the paper

Executing business process models in automotive manufacturing reduces IT interfaces and, thus, failure sources. To this goal, the standards Business Process Model and Notation (BPMN) and Decision Model and Notation (DMN) have been evaluated to model existing automotive production workflows in the context of testing and diagnostics.

The paper focuses on enhancing business process modeling in the automotive industry, particularly for diagnostic workflows in car assembly lines. In this context, it evaluates the effectiveness of BPMN and DMN. The authors propose a structured guideline for modeling these workflows to bridge the gap between business processes and production. The paper contributes to the field by offering a detailed overview and analysis of business process modeling in automotive testing and diagnostics, addressing the need for proper guidelines. The study includes a SWOT analysis and validation through expert interviews and testing, emphasizing the practical application and benefits of the proposed guideline.

The paper discusses the implementation of the Business Process Model and Notation (BPMN) and Decision Model and Notation (DMN) in automotive manufacturing, specifically in car assembly line diagnostics following the requirements of Section 2.6.3 and building the basis for further study. It proposes guidelines for modeling production workflows in this sector, addressing existing gaps in business processes and production alignment.

With this work the usage of BPMN and DMN in the automotive testing domain is highlighted. Employees mention, that for the concrete case of scheduling, DMN is not sufficient as all possible paths are hard to model. Rather, an automated solution is required.

Individual contribution of the candidate

The contribution to this paper includes the overall conceptual design of the thesis work, the collection of relevant data, and the interpretation of the data. The candidate led the drafting of the manuscript.

This subsection was published as *Paper2*: (S. König, Vogel-Heuser, Fieg, et al., 2021)

> S. König, B. Vogel-Heuser, E. Fieg, M. Hahn and O. Kopp, "Modelling Production Workflows in Automotive Manufacturing," 2021. IEEE 23rd Conference on Business Informatics (CBI), Bolzano, Italy, 2021, pp. 39-46, doi: 10.1109/CBI52690.2021.10053.

4.3. Flexible Scheduling of Diagnostic Tests in Automotive Manufacturing (*Paper3*)

Simone König, Maximilian Reihn, Felipe Gelinski Abujamra, Alexander Novy and Birgit Vogel-Heuser

Summary of the article

Enabling various customization options for software-bound car functions also demands flexible manufacturing processes. Automotive testing and commissioning processes during car manufacturing are scheduled w.r.t. minimal run time. To automate the scheduling process, a mathematical optimization solution is needed that considers constraints on time, predecessor-successor relationships between tests, mutual exclusion criteria, resources, and status conditions on the car and assembly line.

The article discusses a new approach to scheduling diagnostic tests in the automotive industry. It focuses on the challenges posed by the increasing complexity and customization of cars, which require more sophisticated testing schedules. The authors present a mathematical model and an automated scheduling procedure to optimize test schedules, considering constraints such as time, resources, and mutual exclusions following the requirements of Section 2.6.2. They also demonstrate the implementation and effectiveness of this approach through a case study with an Original Equipment Manufacturer (OEM). The used system solver is a SAT solver. The goal is to minimize the total test time in a factory, thereby reducing costs and increasing efficiency in automotive manufacturing.

With this work, test scheduling can be automated and is no longer performed manually.

Individual contribution of the candidate

The contribution to this paper includes the literature review and data collection. The candidate conducted the interpretation of the data, and she initialized the manuscript writing.

This subsection was published as *Paper3*: (S. König, Reihn, et al., 2023)

> S. König, M. Reihn, F.G. Abujamra et al. Flexible scheduling of diagnostic tests in automotive manufacturing. Flexible Services and Manufacturing Journal **35**, 320–342 (2023). https://doi.org/10.1007/s10696-021-09438-3

4.4. Online Test Scheduling in Car Production Lines (*Paper4*)

Simone König, Birgit Vogel-Heuser, Florian Karg, Kathrin Land, Emanuele Carbone, Adam Hradecky, Michael Hahn and Oliver Kopp

Summary of the paper

The design of car-specific schedules for testing and diagnostic processes during car manufacturing is optimal w.r.t. flexibility when scheduling is performed online, i.e., adaptive rescheduling on an in-car component.

To find a time-efficient strategy for the automated scheduling of the test procedures, the following experiment was performed to compare the strategy of in-vehicle online scheduling on total test time compared to off-vehicle offline scheduling following the requirements of Section 2.6.2.

In this experiment, tests with accurate run times of a German car production line were chosen.

As a baseline model, offline scheduling with the CP-SAT solver of Google OR-Tools is used. This refers to the implementation of the work in Paper3. The offline schedule is calculated once and needs to be adjusted online.

The adapted solver is a software prototype in Python programming language and is also used as the core function for online scheduling. The schedule calculation time for $N<30$ in the experiment study is about 100 ms. This seemed sufficient for optimization during the run time on the HPC.

The paper concludes that online test scheduling optimizes manufacturing processes, add value to car production, and improve the resilience and flexibility of scheduling systems. It addresses the aspect of flexibility in software-defined manufacturing.

Individual contribution of the candidate

The contribution to this paper was the overall design of the thesis work and data collection. The candidate conducted the experimental setup, interpretation of the data and the drafting of the manuscript.

This subsection was published as *Paper4*: (S. König, Vogel-Heuser, Karg, et al., 2023)

S. König et al., "Online Test Scheduling in Car Production Lines," 2023. IEEE Intelligent Vehicles Symposium (IV), Anchorage, AK, USA, 2023, pp. 1-8, doi: 10.1109/IV55152.2023.10186671.

4.5. BPMN4CARS: A Car-Tailored Workflow Engine (*Paper5*)

Simone König, Birgit Vogel-Heuser, Jan Wilch, Tobias Unger, Michael Hahn, Stjepan Soldo and Oliver Kopp

Summary of the paper

The execution of automotive testing and diagnostic processes within the car architecture requires a car-specific process engine. A car tailored BPMN workflow engine is required as the BPMN standard provides a technique to orchestrate services using workflows.

The paper discusses integrating business process management into car networks, focusing on executable BPMN (Business Process Model and Notation) models for car diagnostics and testing. It introduces BPMN4CARS, a tailored BPMN workflow engine for high-performance vehicular computers, addressing the specific needs of car testing and diagnostics in production lines. The engine is designed to execute diagnostic processes modeled in BPMN, bridging the gap between business processes and automotive IT systems. The paper evaluates two workflow engines, highlighting the efficiency of the BPMN4CARS engine in terms of startup time, CPU, and memory consumption. It concludes that BPMN4CARS meets the identified requirements for an in-car workflow management system of Section 2.6.4, making it suitable for future car production lines.

The engine is evaluated against a conventional workflow engine showcasing its superiority in startup time, CPU, and memory usage. With this work, the step towards executing BPMN process models in software-defined cars is done.

Individual contribution of the candidate

The contribution to this paper was the conceptual design of the thesis work regarding the manufacturing domain. The candidate collected literature sources and made conclusions for the domain. She wrote the initial version of the manuscript.

This subsection was published as *Paper5*: (S. König, Vogel-Heuser, Wilch, et al., 2023)

S. König et al., "BPMN4CARS: A Car-Tailored Workflow Engine," 2023. IEEE 21st International Conference on Industrial Informatics (INDIN), Lemgo, Germany, 2023, pp. 1-6, doi: 10.1109/INDIN51400.2023.10218082.

4.6. Executing Automotive Test Workflows with BPMN and Web Service Orchestration in Software-defined Car Manufacturing (*Paper6*)

Simone König, Birgit Vogel-Heuser, Adam Hradecky, Sophie Horn and Michael Hahn

Summary of the article

The article presents a streamlined approach to integrate lean automation in the production of software-focused vehicles, which is pivotal for fostering a digital infrastructure following the requirements of Section 2.6.3 and 2.6.4. It advocates for the adoption of business process management and execution semantics to ensure a fluid connection between operational activities and IT systems, particularly in the realm of software testing for vehicle manufacturing. This strategy promotes continuous engineering activities that are not tied to any specific software platforms.

This research emphasizes the importance of employing BPMN alongside web service orchestration to oversee testing operations in the realm of collaborative vehicle production. The proposed system facilitates a seamless integration and management of production processes both within the corporate infrastructure and the vehicles, with the flexibility to orchestrate web services that cater to various vehicle models and connect with sophisticated automotive diagnostic tools.

The approach is supported by real test examples and a case study involving an actual software-defined vehicle component. Insights from a focus group of automotive industry experts highlight the benefits of having integrated processes and point to potential areas for future investigation. The importance of creating intuitive modeling tools is stressed, as these are essential for cooperation within multidisciplinary teams dealing with intricate systems. For the manufacturing of software-driven vehicles, it is advisable to create a dedicated web service interface to deepen the integration of IT within vehicle systems.

The paper further recommends that this approach is assessed across different diagnostic sectors, including vehicle design, production, and post-sale services. An inter-manufacturer study could potentially uncover more advantages of standardizing the software-support technologies discussed in this study.

This paper shows the system architecture for executable BPMN in flexible software-defined car manufacturing and builds the conclusion for the work of Paper1, Paper2, Paper3, Paper4 and Paper5 in terms of this dissertation work.

Individual contribution of the candidate

The contribution to the paper includes the overall setup of the project. The candidate collected data, software, and literature from various sources. She drew the interpretation of the results and wrote the initial version of the manuscript.

This subsection is submitted for publication as *Paper6*: (S. König et al., 2024)

> König, S., Vogel-Heuser, B., Hradecky, A. *et al.* Executing automotive test workflows with BPMN and web service orchestration in software-defined car manufacturing. *Production Engineering Research and Development.* (2024). https://doi.org/10.1007/s11740-024-01302-1

In the following chapter, the research contribution of this work is discussed.

5. Assessment of the Fulfillment of the Requirements

In Chapter 4, the findings of the evaluation case studies and the fulfillment of the stated requirements from Chapter 2 are presented and discussed. Most requirements are evaluated positively in separate case studies. Test scheduling and in-car orchestration is flexible in terms of the presented results. Moreover, experimental results and expert assessments proved the suitability of the approach on executable BPMN in test automation in software-defined car manufacturing. The main points are summarized in Table 7.

However, concerns arise around the current lack of knowledge in the BPMN standard. Standards from information technology are still in the beginning in the automotive manufacturing domain. The high level of abstraction in the programming interface is problematic for production employees when realizing measurement scripts. They still prefer programming lines of codes, even though they prefer the concept of BPMN over OTX. A user study with focus on the integration of BPMN Script Tasks into SOVD Scripts can be worthwhile.

Table 7: Summary of the rating of requirement fulfilment based on Table 4 (+ fulfilled, ° partially fulfilled, - not fulfilled, grey: not relevant)

Requirements		**Research Activity**			**Rating**	**Detailed Rating & Reference to Publication**
		Qualitative	**Quantitative**	**Tool Support**		
$R_{Complexity}$	$R_{Quality}$	X		R-based script to plan test cases under the usage of hierarchical clustering. Industrial integration is achieved with a script integrated by an external service provider into an existing software tool.	+	This requirement is investigated in *Paper1* with a qualitative analysis at an OEM including tool support. It is shown that complexity in test planning can be reduced, and quality can be improved during car ramp-up phases by the provided clustering-based approach. The software artifact is still applied at the OEM.

$R_{Variants}$	$R_{Optimize}$		X	Python-based script using the CP-SAT-solver of Google OR Tools (Google OR-Tools, 2024)	+	The automated scheduling approach for tests and diagnosis procedures (see *Paper3*) are improved considering online scheduling on an in-car HPC. This approach minimized schedule run-time as highlighted in *Paper4*. The software artifact of Paper3 is applied at the automotive manufacturing line of the OEM.
	$R_{Orchestration}$		X	Python-based script calling the Event managing component and the process engine to orchestrate tests in-car	+	
$R_{Continuous}$	$R_{Diagnostics}$			OEM-internal BPMN modeling tool	°	The listed requirements are investigated in *Paper2* and *Paper6***.** Support of graphical modeling and execution of test and diagnosis use cases for software-defined cars in manufacturing lines is established with BPMN. However, there is lack in transferability to the overall testing and diagnosis scenario. Especially in *Paper6*, a scenario has been demanded to perform a larger study on executable BPMN in car development, manufacturing lines and after sales services. The software artifacts on executable BPMN are part of the OEM internal SofDCar demonstrator.
	R_{Low}	X			°	
	$R_{Scheduling}$		X	OEM-internal BPMN modeling tool interacts via the Event Managing Component with the Process Engine and Orchestrator	+	
	$R_{No-code}$			X	+	
	$R_{Seamless}$		X		+	
	$R_{Analysis}$			Mock-Up based on the OEM-internal BPMN modeling tool	°	
	$R_{Manufact}$	X		OEM-internal BPMN modeling tool	+	
R_{System}	R_{Re-Use}	X		OEM-internal BPMN modeling tool and Git-based repository	+	A system architecture for executable BPMN in automotive embedded applications is presented and evaluated in

	$R_{SofDCar}$	X		Software artifacts that can be integrated in a software-defined car	+	*Paper6*, whereas the process engine is discussed in *Paper5***.** The artifacts are applied in a prototypical setup at the OEM for a software-defined car.
	R_{In-Car}		X		∘	
	R_{API}	X			∘	

6. Summary and Outlook

To conclude the thesis, a summary of the research contribution is presented. It relates the findings of the publications to the hypotheses presented in the Introduction. Afterwards, it is stated if the requirements are fulfilled, and an outlook on future work is given.

Modern car architectures challenge proven technologies in car manufacturing lines. Software updates become more important and thus, streamlining internal engineering processes are of interest as well. Traditional solutions are not enough, new approaches are needed to interconnect software-defined cars with software-defined manufacturing lines.

In this thesis, an approach is developed that can be used to optimize engineering process of testing and diagnostics in the future automotive production of modern cars. Emphasis is placed on the executability of BPMN process models and the flexibilization of test processes in planning and scheduling phases.

Throughout the thesis, the focus was on four hypotheses.

Hypothesis H1*:* An unsupervised machine learning method can improve test quality of automotive testing processes during car ramp-up.

- ➔ A procedure to plan tests in car ramp-up phases with hierarchical clustering is presented in *Paper1*. The results of a quantitative case study are presented in *Paper1*. The concept is relevant to the OEM, that is why it conducted a patent application for the concept (S. König & Schnittger, 2020).

Hypothesis H2*:* The concept on in-car orchestration of tests supports flexible car manufacturing w.r.t. software-defined cars and improves the existing scheduling process.

- ➔ The mathematical notations for this test scheduling approach are presented in *Paper3*. A patent is registered by the OEM (Ahrens, Finger, et al., 2021). The concept on online scheduling is validated in *Paper4*. It is developed by the candidate in the course of the work and incorporated into a patent (Ahrens, Gelinski Abujamra, et al., 2021).

Hypothesis H3*:* Applying business process management with execution semantic approaches can bridge operational technology (OT) and information technology (IT) for automotive testing processes for software-defined cars in manufacturing to enable continuous engineering processes without vendor-specific software frameworks.

➔ The approach to model test workflows is introduced by an evaluated guideline in *Paper2*. This guideline convinced the OEM to further study executable BPMN.

***Hypothesis H4**:* The system architecture supports the software driven car architecture and its central components to integrate the engineering processes of software-defined cars by multi-disciplinary teams, which are composed of engineers, IT architects, computer scientists, workers and business analysts.

➔ The system architecture is introduced and validated in *Paper6*. It supports flexible scheduling as it is presented in *Paper4* and develops the integration of executable BPMN as in *Paper5*. A patent is registered by the OEM in Germany (Hahn et al., 2022) and the respective content is also registered in the United States of America, Japan, Republic of Korea, and China.

The qualitative and quantitative findings in the included papers present a comprehensive concept for flexibly automating test planning and scheduling in software-driven car manufacturing. This includes a procedure for planning tests during ramp-up phases (*H1*) and an in-car orchestration method that supports flexible test scheduling on high-performance computers to adapt in real-time during test execution (*H2*). The research also introduces an approach using business process management with executable BPMN to bridge operational and information technology, facilitating continuous engineering processes without relying on vendor-specific frameworks (*H3*). Additionally, an OEM-validated system architecture is proposed to integrate engineering processes for software-defined cars, supporting multidisciplinary teams *(H4)*.

In car ramp-up projects, the test planning approach is applied at a German OEM plant. The test scheduling approach will be applied at a future car ramp-up and series manufacturing line at a German OEM plant. The online test scheduling approach is considered as a further development. The concept of executable BPMN is in a prototype phase and further user studies are planned with the car development and after sales departments.

In future work, the approach of this thesis could be extended to cover the entire process chain of test and vehicle diagnostics in an OEM - from development to production and after-sales, including service providers. Improved modeling tools can be investigated as well (Contreras et al., 2022). Process refactoring is relevant for software-defined manufacturing, too (Durán & Salaün, 2022). Testing the workflows is relevant as well (Lopes & Guerreiro, 2023). Of course, scheduling plays here an important role (Land et al., 2024).

Model-based safety and security engineering is further an ongoing research topic (Nigam et al., 2018).

In addition, the use of Generative Artificial Intelligence in interaction with the engineers should be evaluated as engineers have accepted tools such as OpenAI (OpenAI, 2024) as a programming aid within a very short time.

7. Literature

Abouzid, I., Bekali, Y. K., & Saidi, R. (2022). Modelling IoT Behaviour in Supply Chain Business Processes with BPMN: A Systematic Literature Review. *Journal of ICT Standardization.* Advance online publication. https://doi.org/10.13052/jicts2245-800X.1035

Adler, N., Otten, S., Schwär, M., & Müller-Glaser, K. D. (2014). Managing Functional Safety Processes for Automotive E/E Architectures in Integrated Model-Based Development Environments. *SAE International Journal of Passenger Cars - Electronic and Electrical Systems*, *7*(1), 103–114. https://doi.org/10.4271/2014-01-0208

Ahrens, K., Finger, D. E., Gelinski Abujamra, F., Hradecky, A., Kopp, O., König, S., Novy, A., Reihn, M., Schick, M., & Schmid-Eilber, S. (2021). Verfahren zur Ablaufplanung durchzuführender Testprozesse (DE102021002302A8).

Ahrens, K., Gelinski Abujamra, F., Hradecky, A., Kopp, O., König, S., Novy, A., Schick, M., & Schmid-Eilber, S. (2021). Verfahren zur situationsabhängigen Planung von in einer Fertigungsumgebung einer Fahrzeugproduktion durchzuführenden Montageumfängen (DE102021006088A1).

ASAM e.V. (2024a). https://www.asam.net/, last accessed: October 31, 2024

ASAM e.V. (2024b). *Service-Oriented Vehicle Diagnostics*. https://www.asam.net/standards/detail/sovd/, last accessed: October 31, 2024

Aziza, M., Jaswadi, J., & Indah Riwajanti, N. (2022). Modeling and Implementation of the Village Secretary Accounting Information System based on BPMN. *Journal of Applied Business, Taxation and Economics Research*, *2*(2), 114–127. https://doi.org/10.54408/jabter.v2i2.120

Barba, I. (2013). Constraint-based planning and scheduling techniques for the optimized management of business processes. *AI Communications*, *26*(2), 251–253. https://doi.org/10.3233/AIC-130554

Bartels, J.-H., & Zimmermann, J. (2009). Scheduling tests in automotive R&D projects. *European Journal of Operational Research*, *193*(3), 805–819. https://doi.org/10.1016/j.ejor.2007.11.010

Behrendt, S., Ungen, M., Fisel, J., Hung, K.-C., May, M.-C., Leberle, U., & Lanza, G. (2023). Improving Production System Flexibility and Changeability Through Software-Defined Manufacturing. In M. Liewald, A. Verl, T. Bauernhansl, & H.-C. Möhring (Eds.), *Lecture Notes in Production Engineering. Production at the Leading Edge of Technology* (pp. 705–716). Springer International Publishing. https://doi.org/10.1007/978-3-031-18318-8_70

Behrendt, S., Martin, M., Puchta, A., Ströbel, R., Fisel, J., May, M. C., Gönnheimer, P., Fleischer, J., & Lanza, G. (2023). Software-Defined Manufacturing for the Entire Life Cycle at Different Levels of Production. In N. Kiefl, F. Wulle, C. Ackermann, & D. Holder (Eds.), *ARENA2036. Advances in Automotive Production Technology – Towards Software-Defined Manufacturing and Resilient Supply Chains* (pp. 25–34). Springer International Publishing. https://doi.org/10.1007/978-3-031-27933-1_3

Bickelhaupt, S., Hahn, M., Nuding, N., Morozov, A., & Weyrich, M. (2023). Challenges and Opportunities of Future Vehicle Diagnostics in Software-Defined Vehicles. In S. Bickelhaupt, M. Hahn, N. Nuding, A. Morozov, & M. Weyrich (Eds.), *SAE Technical Paper Series, SAE Technical Paper Series.* SAE International400 Commonwealth Drive, Warrendale, PA, United States. https://doi.org/10.4271/2023-01-0847

Błażewicz, J., Ecker, K. H., Pesch, E., Schmidt, G., & Węglarz, J. (Eds.). (2007). *International handbooks on information systems*. *Handbook on scheduling: From theory to applications*. Springer. http://bvbr.bib-bvb.de:8991/F?func=service&doc_library=BVB01&doc_number=014182618&line_number=0002&func_code=DB_RECORDS&service_type=MEDIA

Bogner, J., Kotstein, S., & Pfaff, T. (2023). Do RESTful API design rules have an impact on the understandability of Web APIs? *Empirical Software Engineering*, *28*(6). https://doi.org/10.1007/s10664-023-10367-y

Böhlen, B., Meyer, O., Alrifaee, B., Beerwerth, J., Kampmann, A., Kowalewski, S., Konersmann, M., Rumpe, B., & Steinfurth, F. (2023). Software-Defined Vehicle–Herausforderungen in der Diagnose dienstorientierter Fahrzeugarchitekturen. In B. Bäker (Ed.), *Diagnose in mechatronischen Fahrzeugsystemen XVI: Software-Defined Vehicle, SOVD, Maschinelles Lernen und KI, Standardisierung, HU und ADAS.* TUDpress.

Bouroumi, J. E. L., Guermah, H., & Nassar, M. (2022). Business process Execution: a contextual approach. In *2022 International Conference on Artificial Intelligence of Things (ICAIoT)* (pp. 1–5). IEEE. https://doi.org/10.1109/ICAIoT57170.2022.10121836

Brandstetter, A., Himmelstoss, H., Risling, M., Schel, D., & Oberle, M. (2023). *Towards A Design Of A Software-Defined Manufacturing System Based On A Systematic Literature Review For Enabling A Decentralised High-Rate Electrolyser Production*. Hannover: publish-Ing. https://doi.org/10.15488/13466

Bronner, M., Kalt, S., Koberstaedt, S., & Soltes, M. (2018). A methodological approach for selecting the assembly concept for mounting all-electric vehicles within existing assembly lines. In *2018 IEEE 9th International Conference on Mechanical and Intelligent Manufacturing Technologies (ICMIMT)* (pp. 110–116). IEEE. https://doi.org/10.1109/ICMIMT.2018.8340431

Broy, M., Pretschner, A., Salzmann, C., & Stauner, T. (2006). Software-Intensive Systems in the Automotive Domain:Challenges for Research and Education. In *SAE Technical Paper Series, SAE Technical Paper Series.* SAE International400 Commonwealth Drive, Warrendale, PA, United States. https://doi.org/10.4271/2006-01-1458

Brunninghaus, M., & Rocker, C. (2022). Low-Code Development in Worker Assistance Systems: Improving Flexibility and Adaptability. In *2022 IEEE 20th International Conference on Industrial Informatics (INDIN)* (pp. 366–373). IEEE. https://doi.org/10.1109/INDIN51773.2022.9976178

Brust, M. R., Andronache, A., & Rothkugel, S. (2007). WACA: A Hierarchical Weighted Clustering Algorithm Optimized for Mobile Hybrid Networks. In *2007 Third International Conference on Wireless and Mobile Communications (ICWMC'07)* (p. 23). IEEE. https://doi.org/10.1109/ICWMC.2007.93

Bundesministerium für Digitales und Verkehr. (2023). *EU verabschiedet Data Act*. https://bmdv.bund.de/DE/Themen/Digitales/Digitale-Gesellschaft/EU-Data-Act/eu-data-act.html, last accessed: October 31, 2024

Carlson, R., Do, H., & Denton, A. (2011). A clustering approach to improving test case prioritization: An industrial case study. In *2011 27th IEEE International Conference on Software Maintenance (ICSM)* (pp. 382–391). IEEE. https://doi.org/10.1109/ICSM.2011.6080805

Cheng, H., Kang, G., Liu, J., Wen, Y., Cao, B., & Wang, Z. (2022). BPMN++: Comprehensive Business Process Modeling for Industrial Internet Application. In *2022 IEEE Intl Conf on Parallel & Distributed Processing with Applications, Big Data & Cloud Computing, Sustainable Computing & Communications, Social Computing & Networking (ISPA/BDCloud/SocialCom/SustainCom)* (pp. 548–555). IEEE. https://doi.org/10.1109/ISPA-BDCloud-SocialCom-SustainCom57177.2022.00076

Compagnucci, I., Corradini, F., Fornari, F., Polini, A., Re, B., & Tiezzi, F. (2023). A systematic literature review on IoT-aware business process modeling views, requirements and notations. *Software and Systems Modeling*, *22*(3), 969–1004. https://doi.org/10.1007/s10270-022-01049-2

Contreras, A., Falcone, Y., Salaün, G., & Zuo, A. (2022). WEASY: A Tool for Modelling Optimised BPMN Processes. In S. L. Tapia Tarifa & J. Proença (Eds.), *Lecture notes in computer science. Formal Aspects of Component Software* (Vol. 13712, pp. 110–118). Springer International Publishing. https://doi.org/10.1007/978-3-031-20872-0_7

Corea, C., Kampik, T., & Delfmann, P. (2024). Empirical Evidence of DMN Errors in the Wild - An SAP Signavio Case Study. In J. de Weerdt & L. Pufahl (Eds.), *Lecture notes in business information processing. Business Process Management Workshops* (Vol. 492, pp. 326–336). Springer Nature Switzerland. https://doi.org/10.1007/978-3-031-50974-2_25

David, G., Zmaranda, D. R., Győrödi, R.-Ş., & Győrödi, C. A. (2023). Exploring the Impact of Workflow Engines on Business Process Management in Enterprise Applications. A case-study: Camunda. In *2023 17th International Conference on Engineering of Modern Electric Systems (EMES)* (pp. 1–4). IEEE. https://doi.org/10.1109/EMES58375.2023.10171706

Dietrich, D., Neubauer, M., Lechler, A., & Verl, A. (2024). Iterative Planning as a Holistic Framework for Production System-Wide Optimization Control Loops. In F. J. G. Silva, A. B. Pereira, & R. D. S. G. Campilho (Eds.), *Lecture Notes in Mechanical Engineering. Flexible Automation and Intelligent Manufacturing: Establishing Bridges for More Sustainable Manufacturing Systems* (pp. 611–621). Springer Nature Switzerland. https://doi.org/10.1007/978-3-031-38241-3_69

Dörmer, J., Günther, H.-O., & Gujjula, R. (2015). Master production scheduling and sequencing at mixed-model assembly lines in the automotive industry. *Flexible Services and Manufacturing Journal*, *27*(1), 1–29. https://doi.org/10.1007/s10696-013-9173-8

Dubani, Z., Soh, B., & Seeling, C. (2010). A Novel Design Framework for Business Process Modelling in Automotive Industry. In *2010 Fifth IEEE International Symposium on Electronic Design, Test & Applications* (pp. 250–255). IEEE. https://doi.org/10.1109/DELTA.2010.48

Dumas, M., & Pfahl, D. (2016). Modeling Software Processes Using BPMN: When and When Not? In M. Kuhrmann, J. Münch, I. Richardson, A. Rausch, & H. Zhang (Eds.), *Managing Software Process Evolution* (pp. 165–183). Springer International Publishing. https://doi.org/10.1007/978-3-319-31545-4_9

Durán, F., & Salaün, G. (2022). Optimization of BPMN Processes via Automated Refactoring. In J. Troya, B. Medjahed, M. Piattini, L. Yao, P. Fernández, & A. Ruiz-Cortés (Eds.), *Lecture notes in computer science. Service-Oriented Computing* (Vol. 13740, pp. 3–18). Springer Nature Switzerland. https://doi.org/10.1007/978-3-031-20984-0_1

Egyed, A., Grünbacher, P., Linsbauer, L., Prähofer, H., & Schaefer, I. (2023). Variability in Products and Production. In B. Vogel-Heuser & M. Wimmer (Eds.), *Digital Transformation* (pp. 65–91). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-662-65004-2_3

Elstermann, M., Dittmar, A., & Lederer, M. (Eds.). (2023). *Communications in Computer and Information Science*. *Subject-Oriented Business Process Management. Models for Designing Digital Transformations*. Springer Nature Switzerland. https://doi.org/10.1007/978-3-031-40213-5

Erasmus, J., Vanderfeesten, I., Traganos, K., & Grefen, P. (2020). Using business process models for the specification of manufacturing operations. *Computers in Industry*, *123*, 103297. https://doi.org/10.1016/j.compind.2020.103297

Fernandes, J., Reis, J., Melão, N., Teixeira, L [Leonor], & Amorim, M. (2021). The Role of Industry 4.0 and BPMN in the Arise of Condition-Based and Predictive Maintenance: A Case Study in the Automotive Industry. *Applied Sciences*, *11*(8), 3438. https://doi.org/10.3390/app11083438

Ferreira, F., Marques, A. L., Faria, J., & Azevedo, A. (2016). Hybrid Process Management: A Collaborative Approach Applied to Automotive Industry. *Procedia CIRP*, *56*, 539–544. https://doi.org/10.1016/j.procir.2016.10.106

Flechsig, C., Völker, M., Egger, C., & Weske, M. (2022). Towards an Integrated Platform for Business Process Management Systems and Robotic Process Akönig utomation. In A. Marrella, R. Matulevičius, R. Gabryelczyk, B. Axmann, V. Bosilj Vukšić, W. Gaaloul, M. Indihar Štemberger, A. Kő, & Q. Lu (Eds.), *Lecture notes in business information processing. Business Process Management: Blockchain, Robotic Process Automation, and Central and Eastern Europe Forum* (Vol. 459, pp. 138–153). Springer International Publishing. https://doi.org/10.1007/978-3-031-16168-1_9

Fuchß, D., Kühn, T., Wortmann, A., Pfeiffer, J., & Koziolek, A. (2023). *An Expert Survey on the Use of Informal Models in the Automotive Industry.* https://doi.org/10.5445/IR/1000162389

Funk, D. (2023). Creating a Low-Code Business Process Execution Platform With Python, BPMN, and DMN. *IEEE Software*, *40*(1), 9–17. https://doi.org/10.1109/MS.2022.3212033

Geiger, M., Harrer, S., Lenhard, J., & Wirtz, G. (2018). BPMN 2.0: The state of support and implementation. *Future Generation Computer Systems*, *80*, 250–262. https://doi.org/10.1016/j.future.2017.01.006

Gelgfren, J. M., Arvis, H., Hagemann, S., & Wenzel, S. (2023). Mixed-Integer Programming Model for Scheduling of Modular Automotive Body-In-White Production Systems. In N. Kiefl, F. Wulle, C. Ackermann, & D. Holder (Eds.), *ARENA2036. Advances in Automotive Production Technology – Towards Software-Defined Manufacturing and Resilient Supply Chains* (pp. 68–77). Springer International Publishing. https://doi.org/10.1007/978-3-031-27933-1_7

German Federal Ministry for Economic Affairs and Climate Action (Ed.). (2024a). *SDM4FZI: Software-defined Manufacturing - the future manufacturing paradigm*. https://www.sdm4fzi.de/

German Federal Ministry for Economic Affairs and Climate Action (Ed.). (2024b). *SofDCar: Software-Defined Car*. https://sofdcar.de/language/de/

Giacomo, G. de, Dumas, M., Maggi, F. M., & Montali, M. (2015). Declarative Process Modeling in BPMN. In J. Zdravkovic, M. Kirikova, & P. Johannesson (Eds.), *Lecture notes in computer science. Advanced Information Systems Engineering* (Vol. 9097, pp. 84–100). Springer International Publishing. https://doi.org/10.1007/978-3-319-19069-3_6

Google OR-Tools. (2024). *CP-SAT-Solver* [Computer software]. Google for Developers. https://developers.google.com/optimization/cp/cp_solver?hl=de, last accessed: October 31, 2024

Grambow, G., Hieber, D., Oberhauser, R., & Pogolski, C. (2022). *Arpf - an Augmented Reality Process Framework for Context-Aware Process Execution in Industry 4.0 Processes*. Hochschule Aalen. https://doi.org/24698

Gülke, T., Rumpe, B., Jansen, M., & Axmann, J. (2014). High-Level Requirements Management and Complexity Costs in Automotive Development Projects: A Problem Statement. Advance online publication. https://doi.org/10.48550/arXiv.1409.2634

Haas, R., Elsner, D., Juergens, E., Pretschner, A., & Apel, S. (2021). How can manual testing processes be optimized? developer survey, optimization guidelines, and case studies. In D. Spinellis, G. Gousios, M. Chechik, & M. Di Penta (Eds.), *Proceedings of the 29th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering* (pp. 1281–1291). ACM. https://doi.org/10.1145/3468264.3473922

Haberle, T., Charissis, L., Fehling, C., Nahm, J., & Leymann, F. (2015). The Connected Car in the Cloud: A Platform for Prototyping Telematics Services. *IEEE Software*, *32*(6), 11–17. https://doi.org/10.1109/MS.2015.137

Hahn, M., Kopp, O., König, S., & Sturm, R. (2022). Verfahren zur Durchführung einer Funktionsdiagnose zumindest einer Fahrzeugkomponente und Diagnosesystem (DE102022001254B8).

Hasic, F., Smedt, J. de, & Vanthienen, J. (2018). Redesigning Processes for Decision-Awareness: Strategies for Integrated Modelling. In *2018 11th International Conference on the Quality of Information and Communications Technology (QUATIC)* (pp. 247–250). IEEE. https://doi.org/10.1109/QUATIC.2018.00043

Hu, B., Shi, Y [Yinbin], Cao, Z., & Zhou, M. (2023). A Hybrid Scheduling Framework for Mixed Real-Time Tasks in an Automotive System With Vehicular Network. *IEEE Transactions on Cloud Computing*, *11*(3), 2231–2244. https://doi.org/10.1109/TCC.2022.3194713

Huang, H., & Zhou, S. (2018). An efficient SAT algorithm for complex job-shop scheduling. In *Proceedings of the 2018 8th International Conference on Manufacturing Science and Engineering (ICMSE 2018)*. Atlantis Press. https://doi.org/10.2991/icmse-18.2018.126

International Organization for Standardization (2011). *Road vehicles — Open Test sequence eXchange format (OTX)* (ISO 13209). https://www.iso.org/standard/53507.html

ISO (2020). *Road vehicles — Unified diagnostic services (UDS)*. https://www.iso.org/standard/72439.html, last accessed: October 31, 2024

Jain, A. K., & Dubes, R. C. (1988). *Algorithms for clustering data*. *Prentice Hall advanced reference series*. Prentice-Hall.

Jolak, R., Savary-Leblanc, M., Dalibor, M., Wortmann, A., Hebig, R., Vincur, J., Polasek, I., Le Pallec, X., Gérard, S., & Chaudron, M. R. V. (2020). Software engineering whispers: The effect of textual vs. graphical software design descriptions on software design communication. *Empirical Software Engineering*, *25*(6), 4427–4471. https://doi.org/10.1007/s10664-020-09835-6

Jordan, C., Foth, P., Pretschner, A., & Fruth, M. (2022). Unreliable test infrastructures in automotive testing setups. In M. Harman & H. Miller (Eds.), *Proceedings of the 44th International Conference on Software Engineering: Software Engineering in Practice* (pp. 307–308). ACM. https://doi.org/10.1145/3510457.3513069

Jost, F., & Sinz, C. (2024). Handling Automotive Hardware/Software Co-Configurations with Integer Difference Logic. In T. Kehrer, M. Huchard, L. Teixeira, & C. Birchler (Eds.), *Proceedings of the 18th International Working Conference on Variability Modelling of Software-Intensive Systems* (pp. 103–111). ACM. https://doi.org/10.1145/3634713.3634728

Kalach, V. (2014). *Modellierung von REST-Service-Kompositionen.* https://doi.org/10.18419/opus-3468

Kampmann, A., Luer, M., Kowalewski, S., & Alrifaee, B. (2022). Optimization-based Resource Allocation for an Automotive Service-oriented Software Architecture. In *2022 IEEE Intelligent Vehicles Symposium (IV)* (pp. 678–687). IEEE. https://doi.org/10.1109/IV51971.2022.9827429

Kampmann, A., Mokhtarian, A., Rogalski, J., Kowalewski, S., & Alrifaee, B. (2020). Agile Latency Estimation for a Real-time Service-oriented Software Architecture. *IFAC-PapersOnLine*, *53*(2), 5795–5800. https://doi.org/10.1016/j.ifacol.2020.12.1619

Keller, U., Ostertag, T., Gitt, C., Nietfeld, F., Metzakis, N., & Sturm, J. (2013). Intelligent Hybrid. *ATZextra*, *18*(5), 156–161. https://doi.org/10.1365/s35778-013-0065-z

Kirchhof, J. C., Nieke, M., Schaefer, I., Schmalzing, D., & Schulze, M. (2021). Variant and Product Line Co-Evolution. In W. Böhm, M. Broy, C. Klein, K. Pohl, B. Rumpe, & S. Schröck (Eds.), *Model-Based Engineering of Collaborative Embedded Systems* (pp. 333–351). Springer International Publishing. https://doi.org/10.1007/978-3-030-62136-0_18

Kirikkayis, Y., Gallik, F., & Reichert, M. (2022a). IoTDM4BPMN: An IoT-Enhanced Decision Making Framework for BPMN 2.0. In *2022 International Conference on Service Science (ICSS)* (pp. 88–95). IEEE. https://doi.org/10.1109/ICSS55994.2022.00022

Kirikkayis, Y., Gallik, F., & Reichert, M. (2022b). Modeling, Executing and Monitoring IoT-Driven Business Rules with BPMN and DMN: Current Support and Challenges. In J. P. A. Almeida, D. Karastoyanova, G. Guizzardi, M. Montali, F. M. Maggi, & C. M. Fonseca (Eds.), *Lecture notes in computer science. Enterprise Design, Operations, and Computing* (Vol. 13585, pp. 111–127). Springer International Publishing. https://doi.org/10.1007/978-3-031-17604-3_7

Kirikkayis, Y., Gallik, F., Winter, M., & Reichert, M. (2023). BPMNE4IoT: A Framework for Modeling, Executing and Monitoring IoT-Driven Processes. *Future Internet*, *15*(3), 90. https://doi.org/10.3390/fi15030090

Kocher, A., Da Silva, L. M. V., & Fay, A. (2022). Modeling and Executing Production Processes with Capabilities and Skills using Ontologies and BPMN. In *2022 IEEE 27th International Conference on Emerging Technologies and Factory Automation (ETFA)* (pp. 1–8). IEEE. https://doi.org/10.1109/ETFA52439.2022.9921564

König, M., Bein, L., Nikaj, A., & Weske, M. (2020). Integrating Robotic Process Automation into Business Process Management. In A. Asatiani, J. M. García, N. Helander, A. Jiménez-Ramírez, A. Koschmider, J. Mendling, G. Meroni, & H. A. Reijers (Eds.), *Lecture notes in business information processing. Business Process Management: Blockchain and Robotic Process Automation Forum* (Vol. 393, pp. 132–146). Springer International Publishing. https://doi.org/10.1007/978-3-030-58779-6_9

König, S., Reihn, M., Abujamra, F. G., Novy, A., & Vogel-Heuser, B. (2023). Flexible scheduling of diagnostic tests in automotive manufacturing. *Flexible Services and Manufacturing Journal*, *35*(2), 320–342. https://doi.org/10.1007/s10696-021-09438-3

König, S., & Schnittger, D. (2020). Verfahren zur Testplanung einer erweiterten Funktionsprüfung von Fahrzeugen (DE102020003625A1).

König, S., Vogel-Heuser, B., Fieg, E., Hahn, M., & Kopp, O. (2021). Modelling Production Workflows in Automotive Manufacturing. In *2021 IEEE 23rd Conference on Business Informatics (CBI)* (pp. 39–46). IEEE. https://doi.org/10.1109/CBI52690.2021.10053

König, S., Vogel-Heuser, B., Hradecky, A., Horn, S., & Hahn, M. (2024). Executing automotive test workflows with BPMN and web service orchestration in software-defined car manufacturing. *Production Engineering.* Advance online publication. https://doi.org/10.1007/s11740-024-01302-1

König, S., Vogel-Heuser, B., Karg, F., Land, K., Carbone, E., Hradecky, A., Hahn, M., & Kopp, O. (2023). Online Test Scheduling in Car Production Lines. In *2023 IEEE Intelligent Vehicles Symposium (IV)* (pp. 1–8). IEEE. https://doi.org/10.1109/IV55152.2023.10186671

König, S., Vogel-Heuser, B., Mäckel, R., & Schnittger, D. (2021). Improve Test Quality by Applying a Clustering-based Test Planning Procedure for Customer Experience Vehicle Functions. In *2021 22nd IEEE International Conference on Industrial Technology (ICIT)* (pp. 736–743). IEEE. https://doi.org/10.1109/ICIT46573.2021.9453674

König, S., Vogel-Heuser, B., Wilch, J., Unger, T., Hahn, M., Soldo, S., & Kopp, O. (2023). BPMN4CARS: A Car-Tailored Workflow Engine. In *2023 IEEE 21st International Conference on Industrial Informatics (INDIN)* (pp. 1–6). IEEE. https://doi.org/10.1109/INDIN51400.2023.10218082

Kopp, O., Martin, D., Wutke, D., & Leyman, F. (2009). The Difference Between Graph-Based and Block-Structured Business Process Modelling Languages. Advance online publication. https://doi.org/10.18417/emisa.4.1.1 (3-13 Pages / Enterprise Modelling and Information Systems Architectures, Vol 4, No 1 (2009)).

Kriebel, S., Markthaler, M., Salman, K. S., Greifenberg, T., Hillemacher, S., Rumpe, B., Schulze, C., Wortmann, A., Orth, P., & Richenhagen, J. (2018). Improving model-based testing in automotive software engineering. In F. Paulisch & J. Bosch (Eds.), *Proceedings of the 40th International Conference on Software Engineering: Software Engineering in Practice* (pp. 172–180). ACM. https://doi.org/10.1145/3183519.3183533

Krüger, J., Mahmood, W., & Berger, T. (2020). Promote-pl. In R. E. Lopez-Herrejon (Ed.), *Proceedings of the 24th ACM Conference on Systems and Software Product Line: Volume A - Volume A* (pp. 1–12). ACM. https://doi.org/10.1145/3382025.3414970

Kuhn, A. M., Christ, M., Kuhn, C. B., Liu, P., Tekouo, W., & Kestler, H. A. (2023). Towards the Smart Factory: Process Optimization in Virtual Commissioning. In N. Kiefl, F. Wulle, C. Ackermann, & D. Holder (Eds.), *ARENA2036. Advances in Automotive Production Technology – Towards Software-Defined Manufacturing and Resilient Supply Chains* (pp. 210–218). Springer International Publishing. https://doi.org/10.1007/978-3-031-27933-1_20

Land, K., Vogel-Heuser, B., & Off, R. (2024). Test-based behaviour model derivation and adaption to enable automated test case scheduling for automated production systems. *Production Engineering.* Advance online publication. https://doi.org/10.1007/s11740-024-01289-9

Lauber, R., & Göhner, P. (1999). *Modellierungskonzepte und Automatisierungsverfahren, Softwarewerkzeuge für den Automatisierungsingenieur, Vorgehensweise in den Projektphasen bei der Realisierung von Echtzeitsystemen*. *Prozessautomatisierung / Rudolf Lauber Peter Göhner: Vol. 2*. Springer.

Leymann, F. (2010). BPEL vs. BPMN 2.0: Should You Care? In J. Mendling, M. Weidlich, & M. Weske (Eds.), *Lecture notes in business information processing. Business Process Modeling Notation* (Vol. 67, pp. 8–13). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-16298-5_2

Leymann, F., & Roller, D. (2000). *Production workflow: Concepts and techniques*. Prentice Hall.

Lin, J. T., Chiu, C.-C., Huang, E., & Chen, H.-M. (2018). A Multi-Fidelity Model Approach for Simultaneous Scheduling of Machines and Vehicles in Flexible Manufacturing Systems. *Asia-Pacific Journal of Operational Research*, *35*(01), 1850005. https://doi.org/10.1142/S0217595918500057

Liu, Z., Zhang, W., & Zhao, F. (2022). Impact, Challenges and Prospect of Software-Defined Vehicles. *Automotive Innovation*, *5*(2), 180–194. https://doi.org/10.1007/s42154-022-00179-z

Lohrasbinasab, I., Acharya, P. B., & Colomo-Palacios, R. (2020). BizDevOps: A Multivocal Literature Review. In O. Gervasi, B. Murgante, S. Misra, C. Garau, I. Blečić, D. Taniar, B. O. Apduhan, A. M. A. C. Rocha, E. Tarantino, C. M. Torre, & Y. Karaca (Eds.), *Lecture notes in computer science. Computational Science and Its Applications – ICCSA 2020* (Vol. 12254, pp. 698–713). Springer International Publishing. https://doi.org/10.1007/978-3-030-58817-5_50

Lopes, T., & Guerreiro, S. (2023). Assessing business process models: a literature review on techniques for BPMN testing and formal verification. *Business Process Management Journal*, *29*(8), 133–162. https://doi.org/10.1108/BPMJ-11-2022-0557

Lübke, D. A BPMN Profile for Test Case Execution Visualization. In *16th ZEUS Workshop, ZEUS 2024, Ulm, Germany, 29 February - 1 March 2024.* http://ceur-ws.org (Original work published 2024)

Lübke, D., & Pautasso, C. (2019). Empirical Research in Executable Process Models. In D. Lübke & C. Pautasso (Eds.), *Empirical Studies on the Development of Executable Business Processes* (pp. 3–12). Springer International Publishing. https://doi.org/10.1007/978-3-030-17666-2_1

Mangler, J., & Rinderle-Ma, S. (2022). *Cloud Process Execution Engine: Architecture and Interfaces.* https://doi.org/10.48550/arXiv.2208.12214

Marius, B., Lisa, A., Marko, P., & Manfred, R. (2024). *Transforming Object-Centric Process Models into BPMN 2.0 Models in the PHILharmonicFlows Framework.* https://doi.org/10.18420/modellierung2024_009

Mendling, J [J.], Reijers, H. A [H. A.], & van der Aalst, W [W.M.P.] (2010). Seven process modeling guidelines (7PMG). *Information and Software Technology*, *52*(2), 127–136. https://doi.org/10.1016/j.infsof.2009.08.004

Minguillon, F. E., & Lanza, G [Gisela] (2019). Coupling of centralized and decentralized scheduling for robust production in agile production systems. *Procedia CIRP*, *79*, 385–390. https://doi.org/10.1016/j.procir.2019.02.099

Mülle, J., Tex, C., & Böhm, K. (2019). A practical data-flow verification scheme for business processes. *Information Systems*, *81*, 136–151. https://doi.org/10.1016/j.is.2018.12.002

Mütsch, F., Gremmelmaier, H., Becker, N., Bogdoll, D., Zofka, M. R., & Zöllner, J. M. (2023, May 23). *From Model-Based to Data-Driven Simulation: Challenges and Trends in Autonomous Driving*. http://arxiv.org/pdf/2305.13960v3

Neubauer, M., Reiff, C., Walker, M., Oechsle, S., Lechler, A., & Verl, A. (2023). Cloud-based evaluation platform for software-defined manufacturing. *At - Automatisierungstechnik*, *71*(5), 351–363. https://doi.org/10.1515/auto-2022-0137

Nguyen, V., Huber, S., & Gambi, A. (2021). SALVO: Automated Generation of Diversified Tests for Self-driving Cars from Existing Maps. In *2021 IEEE International Conference on Artificial Intelligence Testing (AITest)* (pp. 128–135). IEEE. https://doi.org/10.1109/AITEST52744.2021.00033

Nigam, V., Pretschner, A., & Ruess, H. (2018). *Model-Based Safety and Security Engineering.* https://doi.org/10.48550/arXiv.1810.04866

Nordemann, F., & Toenjes, R. Resilient BPMN over Wireless. In *Mobile Communication - Technologies and Applications; 26th ITG-Symposium, Osnabrueck, Germany, 2022.* (Original work published 2022)

OASIS (2020). *TOSCA:* Simple Profile in YAML Version 1.3. Organization for the Advancement. 2020

Object Management Group (2011). *Business Process Model and Notation (BPMN), Version 2.0*. https://www.bibsonomy.org/bibtex/2d71e28d8b21b73a067683ba4bfe0321a/porta

Object Management Group (2020). *Decision Model And Notation (DMN), Version 1.3*.

Ochoa, W., Larrinaga, F., & Pérez, A. (2023). Architecture for managing AAS-based business processes. *Procedia Computer Science*, *217*, 217–226. https://doi.org/10.1016/j.procs.2022.12.217

Ochoa, W., Legaristi, J., Larrinaga, F., & Perez, A. (2023). *Dynamic Context-Aware Workflow Management Architecture for Efficient Manufacturing: A Ros-Based Case Study.* https://doi.org/10.2139/ssrn.4542931

Oechsle, S., Walker, M., Fischer, M., Frick, F., Lechler, A., & Verl, A. (2023). Real-Time Capable Architecture for Software-Defined Manufacturing. In N. Kiefl, F. Wulle, C. Ackermann, & D. Holder (Eds.), *ARENA2036. Advances in Automotive Production Technology – Towards Software-Defined Manufacturing and Resilient Supply Chains* (pp. 3–13). Springer International Publishing. https://doi.org/10.1007/978-3-031-27933-1_1

OMG (2023). *OMG System Modeling Language*. https://www.omg.org/spec/SysML/2.0/Beta1/About-SysML

OpenAI. (2024). *GPT-4 Turbo*. https://platform.openai.com/docs/models/gpt-4-turbo-and-gpt-4, last accessed: October 31, 2024

OpenAPI Initiative. (2022). *OpenAPI Specification*. https://www.openapis.org/, last accessed: October 31, 2024

Ortiz, J., Torres, V., & Valderas, P. (2022). Supporting a Bottom-Up Evolution of Microservice Compositions Based on the Choreography of BPMN Fragments. In E. Insfran, F. González, S. Abrahão, M. Fernández, C. Barry, M. Lang, H. Linger, & C. Schneider (Eds.), *Lecture Notes in Information Systems and Organisation. Advances in Information Systems Development* (Vol. 55, pp. 219–236). Springer International Publishing. https://doi.org/10.1007/978-3-030-95354-6_13

Pautasso, C. (2011). BPMN for REST. In R. Dijkman, J. Hofstetter, & J. Koehler (Eds.), *Lecture notes in business information processing. Business Process Model and Notation* (Vol. 95, pp. 74–87). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-25160-3_6

Pautasso, C., Zimmermann, O., & Leymann, F. (2008). Restful web services vs. "big"' web services. In J. Huai, R. Chen, H.-W. Hon, Y. Liu, W.-Y. Ma, A. Tomkins, & X.

Zhang (Eds.), *Proceedings of the 17th international conference on World Wide Web* (pp. 805–814). ACM. https://doi.org/10.1145/1367497.1367606

Peinl, R., & Perak, O. (2019). BPMN and DMN for Easy Customizing of Manufacturing Execution Systems. In C. Di Francescomarino, R. Dijkman, & U. Zdun (Eds.), *Lecture notes in business information processing. Business Process Management Workshops* (Vol. 362, pp. 441–452). Springer International Publishing. https://doi.org/10.1007/978-3-030-37453-2_36

Pesic, M. (2008). *Constraint-based workflow management systems:Shifting control to users.* https://doi.org/10.6100/IR638413

Pesic, M., & van der Aalst, W. M. P [W. M. P.]. (2006). A Declarative Approach for Flexible Business Processes Management. In D. Hutchison, T. Kanade, J. Kittler, J. M. Kleinberg, F. Mattern, J. C. Mitchell, M. Naor, O. Nierstrasz, C. Pandu Rangan, B. Steffen, M. Sudan, D. Terzopoulos, D. Tygar, M. Y. Vardi, G. Weikum, J. Eder, & S. Dustdar (Eds.), *Lecture notes in computer science. Business Process Management Workshops* (Vol. 4103, pp. 169–180). Springer Berlin Heidelberg. https://doi.org/10.1007/11837862_18

Pesl, R. D., Stötzner, M., Georgievski, I., & Aiello, M. (2024). Uncovering LLMs for Service-Composition: Challenges and Opportunities. In F. Monti, P. Plebani, N. Moha, H. Paik, J. Barzen, G. Ramachandran, D. Bianchini, D. A. Tamburri, & M. Mecella (Eds.), *Lecture notes in computer science. Service-Oriented Computing – ICSOC 2023 Workshops* (Vol. 14518, pp. 39–48). Springer Nature Singapore. https://doi.org/10.1007/978-981-97-0989-2_4

Pfeiffer, J., Fuchß, D., Kühn, T., Liebhart, R., Neumann, D., Neimöck, C., Seiler, C., Koziolek, A., & Wortmann, A. (2024). Modeling Languages for Automotive Digital Twins: A Survey Among the German Automotive Industry. In A. Egyed, M. Wimmer, M. Chechik, & B. Combemale (Eds.), *Proceedings of the ACM/IEEE 27th International Conference on Model Driven Engineering Languages and Systems* (pp. 92–103). ACM. https://doi.org/10.1145/3640310.3674100

Pilar, F., Costa e Silva, E., & Borges, A. (2023). Optimizing Vehicle Repairs Scheduling Using Mixed Integer Linear Programming: A Case Study in the Portuguese Automobile Sector. *Mathematics*, *11*(11), 2575. https://doi.org/10.3390/math11112575

Piller, C. (2023). Comparing BPMN 2.0 and PASS: A Review and Analysis of Previous Research. In M. Elstermann, A. Dittmar, & M. Lederer (Eds.), *Communications in Computer and Information Science. Subject-Oriented Business Process Management. Models for Designing Digital Transformations* (Vol. 1867, pp. 163–179). Springer Nature Switzerland. https://doi.org/10.1007/978-3-031-40213-5_12

Poss, L., & Schönig, S. (2023). A Generic Approach Towards Location-Aware Business Process Execution. In H. van der Aa, D. Bork, H. A. Proper, & R. Schmidt (Eds.), *Lecture notes in business information processing. Enterprise, Business-Process and Information Systems Modeling* (Vol. 479, pp. 103–118). Springer Nature Switzerland. https://doi.org/10.1007/978-3-031-34241-7_8

Pretschner, A., Broy, M., Kruger, I. H., & Stauner, T. (2007). Software Engineering for Automotive Systems: A Roadmap. In *Future of Software Engineering (FOSE '07)* (pp. 55–71). IEEE. https://doi.org/10.1109/FOSE.2007.22

Pretschner, A., Prenninger, W., Wagner, S., Kjanel, C., Baumgartner, M., Sostawa, B., Zölch, R., & Stauner, T. (2017). One evaluation of model-based testing and its automation. Advance online publication. https://doi.org/10.48550/arXiv.1701.06815

Roy Thomas Fielding. (2000). *Architectural Styles and the Design of Network-based Software Architectures.* University of California, Irvine.

Schäffer, E., Stiehl, V., Schwab, P. K., Mayr, A., Lierhammer, J., & Franke, J. (2021). Process-Driven Approach within the Engineering Domain by Combining Business Process Model and Notation (BPMN) with Process Engines. *Procedia CIRP*, *96*, 207–212. https://doi.org/10.1016/j.procir.2021.01.076

Schäuffele, J., & Zurawka, T. (2024). *Automotive Software Engineering*. Springer Fachmedien Wiesbaden. https://doi.org/10.1007/978-3-658-43543-1

Schindewolf, M., Kuebler, M., Pett, T., Hettich, L., Agh, H., Lorenz, J., Wagner, S., Weyrich, M., Schaefer, I., Düser, T., & Sax, E. (2024). Integrated Approach for High-quality Software Development of Upgradeable Vehicles. In A. C. Kulzer, H.-C. Reuss, & A. Wagner (Eds.), *Proceedings. 2024 Stuttgart International Symposium on Automotive and Engine Technology* (pp. 202–217). Springer Fachmedien Wiesbaden. https://doi.org/10.1007/978-3-658-45010-6_13

Shi, Y., Reich, D., Epelman, M., Klampfl, E., & Cohn, A. (2017). An analytical approach to prototype vehicle test scheduling. *Omega*, *67*, 168–176. https://doi.org/10.1016/j.omega.2016.05.003

Silver, B. (2009). *BPMN Method and Style*. Cody-Cassidy Pr.

Sivri, S. D., & Krallmann, H. (2015). Process-oriented Knowledge Management within the Product Change Systems of the Automotive Industry. *Procedia Engineering*, *100*, 1032–1039. https://doi.org/10.1016/j.proeng.2015.01.463

Slama, D., Nonnenmacher, A., & Irawan, T. (2023). *The Software-Defined Vehicle: A Digital-First Approach to Creating Next-Generation Experiences*. https://www.bosch-mobility.com/media/global/mobility-topics/mobility-topics/software-defined-vehicle/download/231019_sdv_ebook.pdf

Soleimani Malekan, H., Shafahi, M., Ayat, N., & Afsarmanesh, H. (2018). Enhancing Robust Execution of BPMN Process Diagrams: A Practical Approach. In L. M. Camarinha-Matos, H. Afsarmanesh, & Y. Rezgui (Eds.), *IFIP Advances in Information and Communication Technology. Collaborative Networks of Cognitive Systems* (Vol. 534, pp. 230–243). Springer International Publishing. https://doi.org/10.1007/978-3-319-99127-6_20

Staron, M. (2017). *Automotive Software Architectures*. Springer International Publishing. https://doi.org/10.1007/978-3-319-58610-6

Stiehl, V., Danei, M., Elliott, J., Heiler, M., & Kerwien, T. (2019). Effectively and Efficiently Implementing Complex Business Processes: A Case Study. In D. Lübke & C. Pautasso (Eds.), *Empirical Studies on the Development of Executable Business Processes* (pp. 33–57). Springer International Publishing. https://doi.org/10.1007/978-3-030-17666-2_3

Stötzner, M., Breitenbücher, U., Pesl, R. D., & Becker, S. (2024). Managing the Variability of Component Implementations and Their Deployment Configurations Across Heterogeneous Deployment Technologies. In M. Sellami, M.-E. Vidal, B. van Dongen, W. Gaaloul, & H. Panetto (Eds.), *Lecture notes in computer science. Cooperative*

Information Systems (Vol. 14353, pp. 61–78). Springer Nature Switzerland. https://doi.org/10.1007/978-3-031-46846-9_4

Stütz, L., König, T., Bader, R., & Kley, M. (2023). Improvement of the Scheduling of Automotive Testing Processes Based on Production Scheduling Methods. In N. Kiefl, F. Wulle, C. Ackermann, & D. Holder (Eds.), *ARENA2036. Advances in Automotive Production Technology – Towards Software-Defined Manufacturing and Resilient Supply Chains* (pp. 59–67). Springer International Publishing. https://doi.org/10.1007/978-3-031-27933-1_6

Thames, L., & Schaefer, D. (2016). Software-defined Cloud Manufacturing for Industry 4.0. *Procedia CIRP*, *52*, 12–17. https://doi.org/10.1016/j.procir.2016.07.041

Traganos, K., Grefen, P., Vanderfeesten, I., Erasmus, J., Boultadakis, G., & Bouklis, P. (2021). The HORSE framework: A reference architecture for cyber-physical systems in hybrid smart manufacturing. *Journal of Manufacturing Systems*, *61*, 461–494. https://doi.org/10.1016/j.jmsy.2021.09.003

Valderas, P., Torres, V., & Serral, E. (2022). Modelling and executing IoT-enhanced business processes through BPMN and microservices. *Journal of Systems and Software*, *184*, 111139. https://doi.org/10.1016/j.jss.2021.111139

Valencia-Parra, Á., Parody, L., Varela-Vaca, Á. J., Caballero, I., & Gómez-López, M. T. (2019). DMN for Data Quality Measurement and Assessment. In C. Di Francescomarino, R. Dijkman, & U. Zdun (Eds.), *Lecture notes in business information processing. Business Process Management Workshops* (Vol. 362, pp. 362–374). Springer International Publishing. https://doi.org/10.1007/978-3-030-37453-2_30

van der Aalst, W. M. P [Wil M. P.] (2013). Business Process Management: A Comprehensive Survey. *ISRN Software Engineering*, *2013*, 1–37. https://doi.org/10.1155/2013/507984

van Kempen, R., Lampe, B., Leuffen, M., Wirtz, L., Thomsen, F., Bilkei-Gorzo, G., Busch, J.-P., Feger, I., Geller, C., Kehl, C., Uszynski, O., Wagner-Douglas, L., Zanger, L., Eckstein, L., Klüner, D., Beerwerth, J., Alrifaee, B., Kowalewski, S., Konersmann, M., . . . Dietmayer, K. (2023). *Autotech.Agil: Architecture and Technologies for Orchestrating Automotive Agility*. RWTH Aachen University. https://doi.org/10.18154/RWTH-2023-09783

Vector Informatik GmbH. (2024). *Our Software Factory – Developed to Enable You to Develop*. https://www.vector.com/de/de/produkte/solutions/software-defined-vehicle/software-factory/, last accessed: October 31, 2024

Verbruggen, C., & Snoeck, M. (2021). Model-Driven Engineering: A State of Affairs and Research Agenda. In A. Augusto, A. Gill, S. Nurcan, I. Reinhartz-Berger, R. Schmidt, & J. Zdravkovic (Eds.), *Lecture notes in business information processing. Enterprise, Business-Process and Information Systems Modeling* (Vol. 421, pp. 335–349). Springer International Publishing. https://doi.org/10.1007/978-3-030-79186-5_22

Vogel-Heuser, B. (2016). *Handbuch Industrie 4.0 Bd.4: Allgemeine Grundlagen* (2nd ed.). *VDI Springer Reference*. Springer Berlin Heidelberg. http://gbv.eblib.com/patron/FullRecord.aspx?p=4748961

Vogel-Heuser, B., Kleinert, T., & Schaefer, I. (2023). Methods, approaches and applications in software-driven manufacturing. *At - Automatisierungstechnik*, *71*(5), 327–329. https://doi.org/10.1515/auto-2023-0033

Vogel-Heuser, B., Reif, J. A. M., Passoth, J.-H., Huber, C., Brodbeck, F. C., Maasen, S., Lindemann, U., & Hujo, D. (2022). BPMN++ to support managing organisational, multiteam and systems engineering aspects in cyber physical production systems design and operation. *Design Science*, *8.* https://doi.org/10.1017/dsj.2021.29

Vogel-Heuser, B., Schütz, D., Frank, T., & Legat, C. (2014). Model-driven engineering of Manufacturing Automation Software Projects – A SysML-based approach. *Mechatronics*, *24*(7), 883–897. https://doi.org/10.1016/j.mechatronics.2014.05.003

Vogel-Heuser, B., & Wimmer, M. (2023). *Digital Transformation*. Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-662-65004-2

Vogel-Heuser, B., Zhang, M., Lahrsen, B., Landler, S., Otto, M., Stahl, K., & Zimmermann, M. (2024). SysML' – incorporating component properties in early design phases of automated production systems. *At - Automatisierungstechnik*, *72*(1), 59–72. https://doi.org/10.1515/auto-2023-0099

Walker, M., Tasci, T., Lechler, A., & Verl, A. (2023). Analysis of Real-Time Execution Models for Container-Based Control Applications. In N. Kiefl, F. Wulle, C. Ackermann, & D. Holder (Eds.), *ARENA2036. Advances in Automotive Production Technology – Towards Software-Defined Manufacturing and Resilient Supply Chains* (pp. 14–24). Springer International Publishing. https://doi.org/10.1007/978-3-031-27933-1_2

Weller, M., Stötzner, M., Klinaku, F., & Becker, S. Developing the Software of Future Cars: A Car DevOps Approach. In *Softwaretechnik-Trends, Band 43, Gesellschaft für Informatik e.V.* https://dl.gi.de/server/api/core/bitstreams/05009d50-4d07-410b-9681-cb165a1fb28a/content (Original work published 2023)

Wen, Z., Liu, X., Xu, Y., & Zou, J. (2016). A RESTful framework for Internet of things based on software defined network in modern manufacturing. *The International Journal of Advanced Manufacturing Technology*, *84*(1-4), 361–369. https://doi.org/10.1007/s00170-015-8231-7

Weyrich, M. (2024). *Industrial Automation and Information Technology*. Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-662-69243-1

Windpassinger, H. (2022). On the Way to a Software-defined Vehicle. *ATZelectronics Worldwide*, *17*(7-8), 48–51. https://doi.org/10.1007/s38314-022-0779-z

Wiśniewski, P., Kluza, K., Ślażyński, M., & Ligęza, A. (2018). Constraint-Based Composition of Business Process Models. In E. Teniente & M. Weidlich (Eds.), *Lecture notes in business information processing. Business Process Management Workshops* (Vol. 308, pp. 133–141). Springer International Publishing. https://doi.org/10.1007/978-3-319-74030-0_9

Witkowski, T., Antczak, P., & Antczak, A. (2010). Solving the Flexible Open-Job Shop Scheduling Problem with GRASP and Simulated Annealing. In *2010 International Conference on Artificial Intelligence and Computational Intelligence* (pp. 437–442). IEEE. https://doi.org/10.1109/AICI.2010.212

Xu, D., & Tian, Y. (2015). A Comprehensive Survey of Clustering Algorithms. *Annals of Data Science*, *2*(2), 165–193. https://doi.org/10.1007/s40745-015-0040-1

Xu, L., Chen, L., Gao, Z., Moya, H., & Shi, W. (2021). Reshaping the Landscape of the Future: Software-Defined Manufacturing. *Computer*, *54*(7), 27–36. https://doi.org/10.1109/MC.2021.3074851

Yan, Z., Reijers, H. A [Hajo A.], & Dijkman, R. M. (2010). An Evaluation of BPMN Modeling Tools. In J. Mendling, M. Weidlich, & M. Weske (Eds.), *Lecture notes in business information processing. Business Process Modeling Notation* (Vol. 67, pp. 121–128). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-16298-5_12

zur Muehlen, M., & Recker, J. (2019). How Much Language Is Enough? Theoretical and Practical Use of the Business Process Modeling Notation. In R. King (Ed.), *Notes on Numerical Fluid Mechanics and Multidisciplinary Design. Active Flow and Combustion Control 2018* (Vol. 141, pp. 465–479). Springer International Publishing. https://doi.org/10.1007/978-3-540-69534-9_35

8. List of Figures

9. List of Tables

10. List of Abbreviations

Abbreviation	*Corresponding description in this thesis*
BPMN	Business Process Model and Notation
CPU	Central processing unit
DMN	Decision Model and Notation
DSL	Domain specific language
E/E	Electrical/Electronics
ODX	Open Diagnostic Data Exchange
OEM	Original equipment manufacturer
OMG	Object Management Group
OTX	Open Test sequence eXchange
R	Requirement
RAM	Random-access memory
REST	Representational State Transfer
RQ	Research question
SOVD	Service-Oriented Vehicle Diagnostics
SysML	Systems Modeling Language
UML	Unified Modeling Language
WLAN	Wireless Local Area Network
w.r.t.	with respect to

Appendix A. Included Publications

The Appendix consists of six included paper publications:

[*Paper1*] S. König, B. Vogel-Heuser, R. Mäckel and D. Schnittger, "Improve Test Quality by Applying a Clustering-based Test Planning Procedure for Customer Experience Vehicle Functions," 2021 22nd IEEE International Conference on Industrial Technology (ICIT), Valencia, Spain, 2021, pp. 736-743, doi: 10.1109/ICIT46573.2021.9453674.

[*Paper2*] S. König, B. Vogel-Heuser, E. Fieg, M. Hahn and O. Kopp, "Modelling Production Workflows in Automotive Manufacturing," 2021 IEEE 23rd Conference on Business Informatics (CBI), Bolzano, Italy, 2021, pp. 39-46, doi: 10.1109/CBI52690.2021.10053.

[*Paper3*] S. König, M. Reihn, F.G. Abujamra *et al.* Flexible scheduling of diagnostic tests in automotive manufacturing. *Flexible Services and Manufacturing Journal* **35**, 320–342 (2023). https://doi.org/10.1007/s10696-021-09438-3

[*Paper4*] S. König et al., "Online Test Scheduling in Car Production Lines," 2023 IEEE Intelligent Vehicles Symposium (IV), Anchorage, AK, USA, 2023, pp. 1-8, doi: 10.1109/IV55152.2023.10186671.

[*Paper5*] S. König et al., "BPMN4CARS: A Car-Tailored Workflow Engine," 2023 IEEE 21st International Conference on Industrial Informatics (INDIN), Lemgo, Germany, 2023, pp. 1-6, doi: 10.1109/INDIN51400.2023.10218082.

[*Paper6*] S. König, B. Vogel-Heuser, A. Hradecky *et al.* Executing automotive test workflows with BPMN and web service orchestration in software-defined car manufacturing. *Production Engineering Research and Development.* (2024). https://doi.org/10.1007/s11740-024-01302-1

Paper1

Improve Test Quality by Applying a Clustering-based Test Planning Procedure for Customer Experience Vehicle Functions

Simone König, Birgit Vogel-Heuser (Senior Member IEEE)
Chair of Automation and Information Systems
Technical University of Munich
Garching, Germany
simone.k.koenig@daimler.com, vogel-heuser@tum.de

Rainer Mäckel, Dominik Schnittger
Mercedes-Benz AG
Sindelfingen, Germany

2021 22nd IEEE International Conference on Industrial Technology (ICIT) | 978-1-7281-5730-6/20/$31.00 ©2021 IEEE | DOI: 10.1109/ICIT46573.2021.9453674

***Abstract*—The automotive industry is undergoing technological change that is not only shaped by electro mobility, but also by the rising importance of customer experience. It puts the customers first and manages their journeys with the help of software. These software-driven customer experience vehicle functions can promote competitive technological advantages for the automotive company. In the ramp-up phase of new car models, testing of electric and electronic customer experience vehicle functions is conducted on prototypes. The functional availability of these functions depends on various requirements. Thus, test planning quickly becomes difficult to conduct manually. In order to optimize the test planning procedure for these functions, we developed a data-driven procedure that automates test planning on test objects. In this paper we introduce a procedure that comprises interaction of hierarchical clustering, weighted parameters and ranking. We evaluated a proof of concept at an Original Equipment Manufacturer and conclude that the procedure leads to higher quality in testing. This use case is a successful application of unsupervised machine learning in the automotive industry.**

***Index Terms*—automotive, customer experience, test planning, application of hierarchical clustering, weighting, ranking**

I. Introduction - Test Planning in Automotive

The car of the future will be electrified, autonomous, shared, connected and updated annually [1]. Vehicle software functions enhance the customer's travel quality and the range of these functions is increasing while development times are dropping, which means that overall, automotive development complexity is growing.

To enable vehicle software functions, modern premium cars can have more than 50 electronic control units. Electric and electronic functions, such as engine control, are real-time functions. Nearly all other functions demand at least soft real-time behavior, such as the high beam assistant. The high beam assistant can be activated and deactivated automatically by vehicle sensors, giving the customer a pleasant driving experience. This function is complex and needs to be tested.

All of this explains why testing in automotive software architectures is increasingly important in the car development process. Original Equipment Manufacturers (OEMs) and suppliers differentiate five testing phases in automotive software development (compare Fig. 1).

Customer functions are tested on prototypes during the ramp-up phase of new car models. The testing phase focuses on verifying that the functions of the system work according to defined specifications concerning customer experience. Customer testing is conducted via test cases that consist of a collection of separate test steps. Whereas component or system tests are run with high automation, automotive engineers use sophisticated measurement technology to manually run customer tests. Thus, customer testing is the most effort-intensive type of automotive software testing.

To plan the testing phase, engineers should ideally take six requirements $R_1 - R_6$ into account:

R_1: Testing of a software release is limited to a certain number of prototypes. These prototypes are assembled within different phases based on production scheduling dates. Prototypes vary in equipment codes. Equipment codes are three or four digits and indicate the specification of a prototype. They are important because the protoype is assembled and manufactured during production with respect to these codes. Testing a test case on a prototype is dependent on a certain combination of equipment codes.

R_2: Prototypes that are assembled first should be tested first. An early test planning date means that anomalies can be detected earlier, allowing fault recovery process to be started at an earlier stage.

R_3: There are equipment codes that are more important for testing than others. They need to be prioritized during the test planning phase, such as when there is a special interest in testing autonomous vehicle functions.

R_4: Testing should be performed on prototypes with the greatest difference in terms of all equipment codes. This ensures that maximal number of functional relationships through signals in the electrical system occur. Anomalies can be detected and fault recovery process can be started.

R_5: The more tests are performed in a testing session on a prototype the better. Exhaustive testing means that measurement technology has been used more efficiently.

978-1-7281-5730-6/21/$31.00 ©2021 IEEE

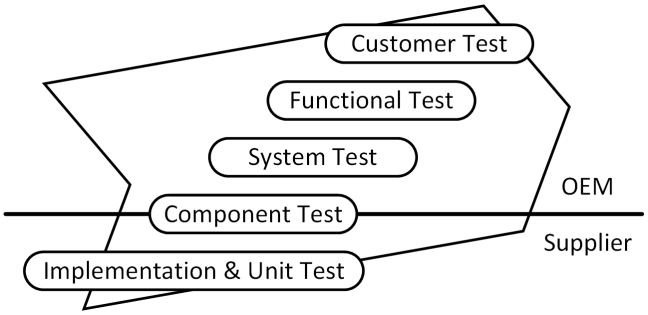


Fig. 1. Testing phases in automotive software development [2].

R_6: Test cases are conducted a certain number of times in a software release until the threshold number, manually defined by engineers, is reached.

Ensuring these requirements means that testing is performed on selected prototypes where maximal functional relationships through signals in different system networks occur. By testing on maximally different test platforms in prototypes, we get more errors due to equipment code dependent system influences. Testing on maximally different prototypes can determine error severity as well. There is no conflict of objectives between the requirements.

If faults occur and customer functions do not work properly due to a combination of different boundary conditions, the fault recovery process will be able to speed up the degree of maturity of the car line. This enables the engineers to ensure the quality of their testing work during the ramp-up phase: the testing results are meaningful for the maturity process.

However, in today's manual test planning procedure the engineers do not make use of all requirements: A prototype is selected for the test plan if it has an early production scheduling date, i.e., R_1, R_2 and R_6 need to be fulfilled. This can lead to a small covered set of all functional relationships since potential influences of different electric equipment codes on the test behaviour are not considered. As a result, the quality of testing for the maturity assurance is reduced. This fact is not satisfactory for the engineers.

We understand that customer experience vehicle function test planning is a multi-dimensional problem and that it is difficult to carry out manually. Since artificial intelligence methods help to optimize and automate data-driven processes, we apply these methods to automate test planning with requirements $R_1 - R_6$. In order to organize unlabeled data, i.e., prototypes as test objects, into dissimilarity groups and thus, to fulfill R_4, clustering methods are a modul for handling the test planning problem. To fulfill all requirements, we use weighted clustering and ranking parameters as components of the new procedure. Assuming that fulfillment of the outlined requirements $R_1 - R_6$ leads to a higher quality of testing, we formulate the research question RQ as follows:

RQ: Can we improve the quality of customer experience vehicle function testing by the application of an automated cluster-based, weighted and ranked procedure for test planning with respect to a quantitative metric?

The main contribution of this paper is an evaluated procedure that helps engineers to improve test quality with optimized test planning of customer experience vehicle functions in the automotive industry using hierarchical clustering with weighted and ranked requirements.

The remainder of this paper is structured as follows: In Section II, the fundamentals of clustering are introduced because the new test planning procedure will be based on clustering. In Section III, we propose the procedure for test planning of customer experience vehicle functions. In Section IV, we refine the research question and present the implementation. Section V considers the evaluation results at an OEM. We conclude and outline further work in Section VI.

II. Fundamentals of Clustering

This section provides a brief introduction to clustering as part of unsupervised machine learning, weighted clustering and ranking as well as selected references essential for the differentiation to related work of this paper.

A. Clustering

Clustering is the process of dividing objects into subsets that belong together in the context of a particular problem [3]. Xu and Tian [4] provide a survey on clustering algorithms. The clustering algorithms have in common that they need a proximity measure (similarity or dissimilarity measure), criterion function to evaluate a clustering and an algorithm to compute clustering. Popular examples of clustering are Centroid-based Clustering with K-means, Distribution-based Clustering with Gaussian Mixture Model, Density-based Clustering with DBSCAN and Connectivity-based Clustering with Hierarchical Clustering. We refer to the standard literature of clustering theory since the given information are basic as shown in [3] or [4].

To choose a concrete clustering algorithm to fulfill requirement R_4 we need to take the data and sample size into account as well as the time behavior of the algorithm: Since test planning of customer experience vehicle functions is conducted for every software release, the sample size of test objects is rather small. Due to the manual development process explained in the introduction of this paper, real-time behaviour of the test planning algorithm is not necessary. Moreover, a clustering method which shows hierarchy and can be interpreted by engineers being no mathematicians is preferred and thus, we proceed with the theory of Hierarchical Clustering.

1) Agglomerative Hierarchical Clustering: Hierarchical clustering aims to construct a hierarchical relationship among data in order to cluster. In this paper the agglomerative hierarchical clustering, also known as the "bottom-up" approach, is applied as shown in [3]. Each observation starts in its own cluster, and pairs of clusters are merged as one moves up the hierarchy.

2) Dissimilarity Measure: A measure of dissimilarity between observations is used to decide which clusters to combine as shown in [4]. It comprises an appropriate distance metric and a linkage criterion.

TABLE I
CLASSIFICATION OF RELATED WORK (SYMBOL "X" MEANS APPLICABLE, SYMBOL "-" MEANS NON APPLICABLE OR NOT SPECIFIED)

Criteria	Reference							
	Vong, Wong and Ip [5]	Satyendra and Prasad [6]	Ramler and Klammer [7]	Larprattanakul and Suwannasart [8]	Khalid and Qamar [9]	Arafeen and Do [10]	Barhate [11]	This paper
Application field	Test Case Prioritization for Industrial Software	Functional Sorting and Prioritization of Approval Queues	Acceptance Testing for Industrial Software	Model for Regression Test Case Selection	Test Case Prioritization	Test Case Prioritization for Software	Automotive Testing Strategy	Automotive Test Planning
Test Planning of Customer Experience Vehicle Functions	-	-	-	-	-	-	-	X
Sample Size	large	small	-	-	large	large	large	small
Selection Approach	Assemble clustering (K-means, Hierarchical Clustering)	Multi-objective optimization techniques	Model-based Testing with Acceptance Test-Driven Development	Object Dependency Graph	K-means and K-medoids	Requirements-based Clustering with K-means	Taguchi method, prioritization, automation	Hierarchical Clustering
Multi-dimensional Requirements	X	X	X	-	X	X	X	X
Weighted Clustering	-	X	-	-	X	X	X	X
Ranking	X	X	-	-	X	X	X	X

a) Distance Metric: A distance metric is a function to measure the distances between sets of observations. It influences the shape of these clusters, as some elements might be closer or further away to each other depending on the distance.

We define a distance metric d on a set X with

$$d : X \times X \to \mathbb{R}$$

For all $x, y, z \in X$ a distance metric d meets the following conditions called non-negativity, symmetry and triangle inequality:

$$d(x, y) \geq 0 \;\; \& \;\; d(x, y) = 0 \iff x = y \tag{1}$$

$$d(x, y) = d(y, x) \tag{2}$$

$$d(x, y) \leq d(x, z) + d(z, y) \tag{3}$$

b) Linkage Criterion: The linkage criterion specifies the dissimilarity as a function of the pairwise distances of the observations.

3) Performance Evaluation Criteria: To evaluate the results of clustering using different distance metrics and linkage criteria, there are popular performance evaluation criteria, e.g. Silhouette Coefficient, as shown in [4]. A higher Silhouette Coefficient score relates to a model with better clustering.

B. Weighted Clustering

Weighted Clustering helps us to fulfill requirement R_3. Weight in the sense of clustering means that a data point is associated with a real-valued weight representing its "importance". This notion of weight generalizes the notion of data point duplication. There are combinations in clustering which are weight-sensitive, weight-considering and weight-robust [12]. Choosing a weight-robust algorithm results in invariance of data point duplication.

C. Ranking

We integrate a function into the clustering procedure to maximize the outcome of requirements R_2, R_5: The multiplicative score function can help us to measure the fulfillment of attributes that need to be maximized in the sense of "more is better" [13]. For two attributes X_i, X_j with $i \neq j$ the multiplicative score function MS is defined as

$$MS_{rp_i, rp_j}(X_i, X_j) = X_i^{rp_i} \cdot X_j^{rp_j} \tag{4}$$

where $rp_i, rp_j \in \mathbb{R}$ are the ranking parameters for X_i, X_j.

D. Literature

We identified the criteria relevant for classification of related work (compare Table I). They are directly derived from the technical background. We apply hierarchical clustering of a small group of test objects, multi-dimensional requirements, weighting and ranking as core elements of the procedure we will develop. There is no appropriate literature for test planning of customer experience vehicle functions with help of machine learning. Acceptance testing and test case prioritization are not focused in our work. For a small data set, hierarchical clustering is a well-established method in machine learning. For both test planning with hierarchical clustering of multi-dimensional weighted requirements and ranking, and automotive customer experience vehicle function test planning, an appropriate approach is missing.

III. New Test Planning Procedure

In the introduction, we established relevant requirements $R_1 - R_6$ that should be met in the test plan of customer experience vehicle functions. We will be focusing on the procedure with respect to these requirements in the following (compare Fig. 2).

A. Data Preprocessing

This section addresses the introduced requirements on data preprocessing ($R_1 - R_6$).

1) Requirements on Data concerning R_1: There are vehicle prototypes $V = \{v_1, \ldots, v_n\}$ of the same model series as test objects where $n \in \mathbb{N}$ is the sample size. For each test object v_i with $i \in \{1, \ldots, n\}$ the relevant information for test planning is defined via characteristics, i.e., equipment codes $O = \{o_1, \ldots, o_q\}$ with $q \in \mathbb{N}$ and vehicle body l as special part of the equipment codes representing a combination of the model series, design variant, engine and steering selected by the customer. We map the prototype information with $m \in \mathbb{N}$ software-release dependent test cases $TC = \{tc_1, \ldots, tc_m\}$ dependent on equipment codes combinations. The test cases tc_k for $k \in \{1, \ldots, m\}$ are split up into test steps. We identify the scheduling dates for the prototypes.

2) Requirements on Data concerning R_3, R_4, R_6: These requirements need to be specified by the business unit. There are equipment codes being especially relevant for test planning, i.e., these important equipment codes should make a difference in clustering. The $p \in \mathbb{N}$ important equipment codes are the set $E = \{e_1, \ldots, e_p\}$ being a subset of all equipment codes O, i.e., $E \subset O$. The vehicle body l of a prototype is an important characteristic as well. Both information are higher weighted (weights are introduced as semi-manual requirement). We introduce threshold $t \in \mathbb{N}$ defining the number of testing times to be performed for a test case.

3) Requirements on Data concerning R_2, R_5: These requirements are specified by a data scientist in discussion with the business unit. There are weighting parameters $W = \{w_1, w_2\}$ with $w_1, w_2 \in \mathbb{N}$ for the set of important equipment codes E and the vehicle body l. With help of the set of ranking parameters $RP = \{rp_2, rp_5\}$ with $rp_2, rp_5 \in \mathbb{R}$ for the multiplicative score function MS defined in (4), we prioritize a test object with an earlier scheduling date (attribute X_2 for requirement R_2), ranked with r_2, and aim to maximize the number of tests in a testing session (attribute X_5 for requirement R_5), ranked with r_5. Test cases should be tested on test objects where both attributes are maximized such that requirements R_2, R_5 are fulfilled, i.e.,

$$\max_{(X_2, X_5)} MS_{r_2, r_5} \tag{5}$$

It is worth a larger study to evaluate the fixation of parameters for a car line by a data scientist.

B. Proposed Procedure of Test Planning

We are prepared to conduct planning for each test case tc_k.

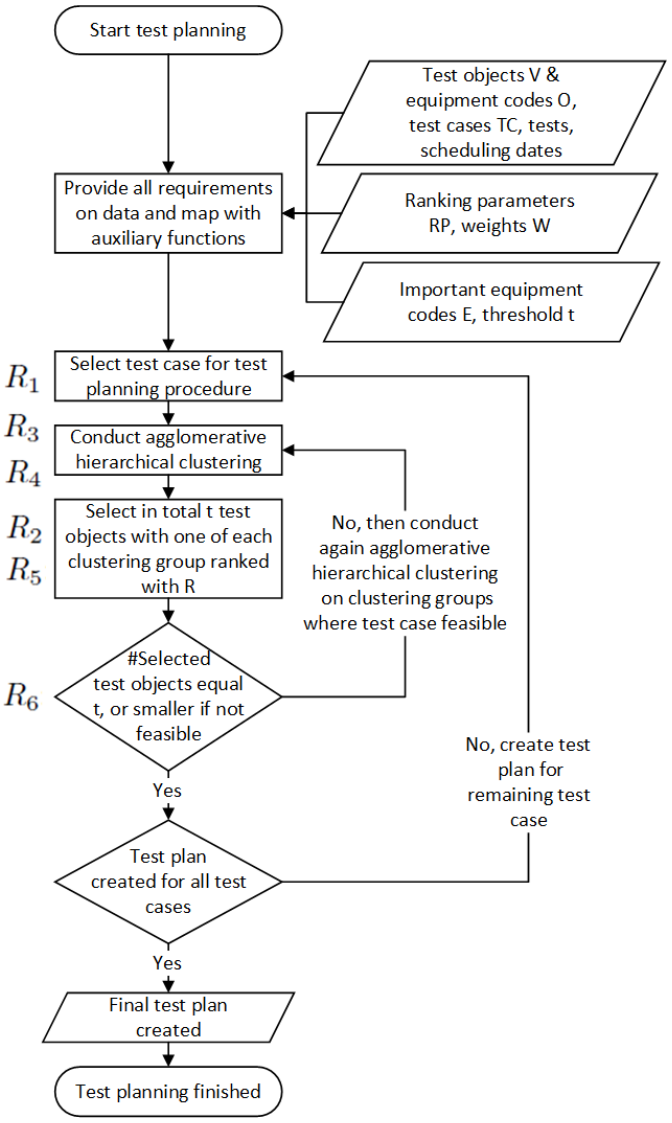


Fig. 2. Flow chart of clustering-based test planning procedure for customer experience vehicle functions to improve quality of testing including requirements $R_1 - R_6$.

1) Hierarchical Clustering to meet R_1, R_3, R_4: Perform hierarchical clustering (R_4) with the test object information mapped with (important) equipment codes and the vehicle body in consideration of the weighting parameters (R_1, R_3). In hierarchical clustering, the threshold t equals the number of cluster groups of test objects since we aim to select t test objects for each test case. As result we identify threshold t cluster groups of test objects.

2) Select Test Objects for Test Cases with Ranking to meet R_2, R_5, R_6: Next, the procedure finds out which test objects are most suitable for test planning with respect to requirements R_2, R_5. To evaluate the degree of fulfillment, the multiplicative score function (5) is calculated for each of the test objects. It is based on the delta of days between actual and latest scheduling date (R_2) and on the number of possible tests that can be assigned to the test objects (R_5). Each test case can now be

assigned to test objects from ideally each of the t clusters calculated prior by prioritizing the highest multiplicative score.

If there are one or more cluster without test objects that can apply the current test case, the clustering needs to be repeated iteratively on the remaining clusters until there are either t clusters with matching test objects or t or less test objects left in total. For the latter case, the prioritization by multiplicative score is inapplicable and all of the test objects are assigned. This fulfills requirement R_6.

This part of the procedure guarantees to find dissimilar test objects for each test case while taking the scheduling date and the number of tests into account.

3) Output for Test Planning: The test plan contains maximum threshold t of test objects for every test case tc_k with requirements $R_1 - R_6$ being fulfilled.

4) Quantitative Quality Assessment: To evaluate the new test planning result we need to apply a quantitative assessment. In regard of the procedure there are mainly three parameters that can quantitively measure the difference between a baseline model and the new test planning approach.

For each test case tc_k and threshold t: Are the maximum t test objects selected from t different cluster groups achieved by hierarchical clustering? Is there a variance of the important equipment codes in the t test objects? Is there a variance in the vehicle body of the t test objects?

In the quality metric the focus is on

- t test object clusters represented by requirement R_4 and
- maximal t vehicle bodies represented by requirement R_3

since the variance of important equipment codes E is dependent on the construction of the test objects and would bring complexity into the sense of the quality metric.

Becoming more specific, we define the quantity for threshold $t = 3$ referring to the experience of our business partners. For other t it is easy to adapt following vector notation. If there are less than t potential test objects to choose from a test case, the quantity is adapted as well, e.g., for $t_{less} = t - 1$. For each test case tc_k we create a vector b_{new} that includes the following parameters for the new test planning procedure:

$$b_{new} = \begin{pmatrix} u_1 \\ u_2 \end{pmatrix} \in \mathbb{N}^2$$

with

$$u_1 = \begin{cases} 1 & \text{, 3 test objects in 3 cluster groups} \\ \frac{2}{3} & \text{, 3 test objects in 2 cluster groups} \\ \frac{1}{3} & \text{, 3 test objects in 1 cluster group} \end{cases}$$

explaining the variance of the $t = 3$ test objects in the cluster groups of hierarchical clustering and

$$u_2 = \begin{cases} 1 & \text{, each test object has a separate vehicle body} \\ \frac{2}{3} & \text{, 2 test objects have the same vehicle body} \\ \frac{1}{3} & \text{, 3 test objects have the same vehicle body} \end{cases}$$

explaining the variance of the vehicle body in the $t = 3$ test objects. We define vector b_{old} in the corresponding way interpreted as variance of test objects in cluster groups and vehicle bodies of the baseline model. For the set of all test cases $TC = \{tc_1, \ldots, tc_m\}$ and $k \in \{1, \ldots, m\}$ we define d as

$$\begin{aligned} d_{QQM}(tc_k) &= \sum_{\forall \text{elements in } b} \mathbb{1}_{(b_{new}(tc_k) - b_{old}(tc_k) > 0)} \\ &= \mathbb{1}_{(b_{new,1}(tc_k) - b_{old,1}(tc_k) > 0)} \\ &\quad + \mathbb{1}_{(b_{new,2}(tc_k) - b_{old,2}(tc_k) > 0)} \end{aligned} \tag{6}$$

d_{QQM} is interpreted as the sum of indicator functions of whether the elements of b_{new} in rows are greater than b_{old}. It validates whether the test plan contains variance of u_1 and u_2 in favour of the new procedure represented by b_{new}.

We now define the quantitative quality metric QQM for the quantification of the quality gain of the presented clustering procedure compared with the baseline model as

$$QQM = \frac{\sum_{TC} d_{QQM}}{m} \tag{7}$$

with $m \in \mathbb{N}$ the sample size of test cases TC. For $t = 3$ it holds that $b_{old}, b_{new} \in \mathbb{N}^2$, $d_{QQM} : \mathbb{N}^2 \times \mathbb{N}^2 \to \{0, 1, 2\}$ and for QQM it holds that $QQM \in [0, 2]$.

A weakened QQM is called $QQMW$ with greater or equal-sign instead of greater-sign as specified for QQM. Due to the greater and greater or equal-signs, QQM and $QQMW$ do not fulfill the symmetric characteristic of a metric stated in (2), but do fulfill (1) and (3). Since we want to measure the improvement of fulfilling requirements $R_1 - R_6$ by the new test planning compared to the baseline model, the asymmetry is naturally intended and not a mathematical obstacle.

IV. Refined Research Question and Implementation

We refine the research question RQ stated in the introduction, that we would like to evaluate with the results of the implementation.

A. Refinement of Research Question

In the testing scenario described above, we have a small sample size of test objects. We assume that fulfillment of requirements $R_1 - R_6$ leads to a higher quality of testing. We refine the research question RQ by selecting the procedure in Fig. 2 as well as QQM and $QQMW$ as quantitative metrics.

RQ: Can we improve the quality of customer experience vehicle function testing by the application of the cluster-based, weighted and ranked procedure in Fig. 2 for test planning with respect to the asymmetric metrics QQM and $QQMW$?

B. Implementation of the Proposed Procedure

We describe the implementation of the procedure for the use case stated in the introduction of this paper as proof of concept at an OEM. Essential requirements on data are specified by the engineer respresenting the business unit and a data scientist (compare Table II).

TABLE II
IMPLEMENTATION DETAILS FOR INDUSTRIAL PROOF OF CONCEPT

Criteria	Proof of Concept at OEM
Agglo. Hierarchical Clustering	Package AgglomerativeClustering
Distance Metric	Manhattan
Linkage Criterion	Complete-linkage
Performance Evaluation Criterion	Silhouette Coefficient
Weighted Clustering	$w = 3$ for important equipment codes and vehicle body, 1 for everything else
Ranking	Multiplicative score function with parameter $rp = 0.5$

1) Details on Computational Implementation Environment: We implemented the procedure in Python 3.7 via Jupyter Notebook. For agglomerative hierarchical clustering we used scikit-learn 0.22.1.

2) Details on Data Requirements concerning R_1: Test planning is conducted on vehicle prototypes $V = \{v_1, \ldots, v_n\}$ as test objects with $n = 25$ defined by ascending identifiers, e.g., $v_1 = xxxx0556$ of data type string. There are equipment codes $O = \{o_1, \ldots, o_{86}\}$ with $q = 86$ and vehicle bodies l_1, l_2, l_3, l_4, e.g., $o_1 = P43$ representing a special comfort package and $l_1 = xxx13212$. Both identifiers are of data type string. Testing is conducted on $m = 160$ software-release dependent test cases $TC = \{tc_1, \ldots, tc_{160}\}$ of data type string, e.g., $tc_{24} = 1079977648$ representing a specific test case on the high beam assistant. Prototypes are assembled on 16 different scheduling days in the interval of October 22nd, 2019 until November 11th, 2019. The scheduling days are real dates of data type date.

3) Details on Data Requirements concerning R_3, R_4, R_6: Important equipment codes $E = \{e_1, \ldots, e_{13}\}$ with $p = 13$ are manual requirements to be specified by the engineer which are especially important for customer experience vehicle function testing, e.g., code 200 for an autonomous vehicle application. The data type is string. The testing threshold shall be $t = 3$.

4) Details on Data Requirements concerning R_2, R_5: Since important equipment codes and the vehicle body are equally important to us in test planning, we introduce $w = w_1 = w_2$. Following Table II they are higher weighted with weights $w_1 = w_2 = 3$, all other equipment codes with 1. The ranking attributes X_2, X_5 are equally important to us, such that we introduce the ranking parameter $rp = rp_2 = rp_5 = 0.5$.

The multiplicative score function MS (4) becomes

$$MS_{rp,rp}(X_2, X_5) = MS_{0.5,0.5}(X_2, X_5) = X_2^{0.5} \cdot X_5^{0.5} \quad (8)$$

where attribute X_2 is the number of tests in a test case and attribute X_5 is the number of days available to test a software release starting from the scheduling date. Test cases should be tested on test objects where both attributes are maximized in

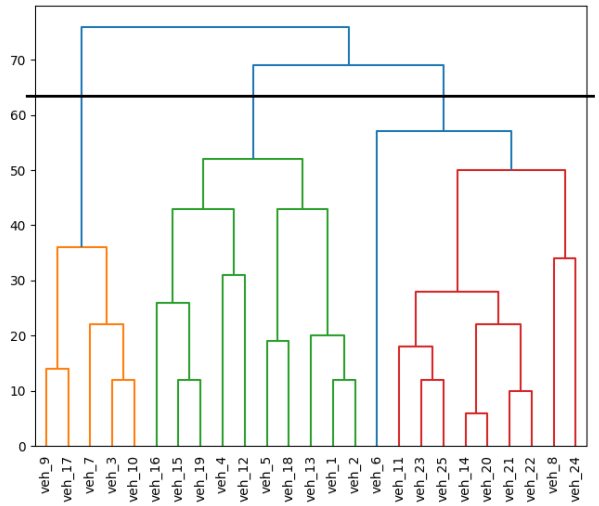


Fig. 3. Dendrogram of $n = 25$ test objects having performed agglomerative hierarchical clustering to meet requirement R_4, whereby requirements R_1, R_3 are fulfilled by data preprocessing. The solid horizontal line divides the test objects into $t = 3$ cluster groups.

(5) such that requirements R_2, R_5 are fulfilled, i.e.,

$$\max_{(X_2,X_5)} MS_{0.5,0.5} \quad (9)$$

5) Details on Baseline Model: In the baseline model for test planning, only an early scheduling date is relevant, i.e., requirements R_1, R_2 and R_6 are fulfilled.

V. RESULTS AND DISCUSSION OF PROOF OF CONCEPT AT OEM

In the following the results of the proof of concept at the OEM based on the implementation and the findings for the research question are presented. The discussion of the results concludes the section.

A. Results of Proof of Concept

1) Result of Hierarchical Clustering: A dendrogram is a popular way to illustrate hierarchical clustering results (compare Fig. 3). There are multiple clusters of test objects. Three clusters of test objects contain 5, 10 and 10 prototypes.

2) Result of Selection of Prototypes for Test Cases with Ranking: Our data set includes $m = 160$ test cases for the $n = 25$ test objects. The final question is which maximal t test objects can we select for each test case with help of the multiplicative score function MS.

Exemplary, we have a closer look at test case tc_{24}, before we progress to the general analysis of all test cases. Testing of tc_{24} is not possible on four prototypes due to build-up constraints: v_1, v_2, v_8, v_{13}. The proposed test planning procedure selects $t = 3$ constructible prototypes with respect to the equipment codes and these are v_3, v_5 and v_6 being part of all three cluster groups (compare Fig. 3). It selects the prototypes of groups with maximal ranking value of the multiplicative score MS

TABLE III
RESULTS ON RANKING WITH DESCENDING MULTIPLICATIVE SCORE MS LIMITED TO 21 PROTOTYPES FOR TEST CASE tc_{24}.

Multiplicative score MS	***Vehicle prototypes*** V ***as test objects***
158.8	v_5
147.2	v_6
134.7	v_{18}
134.1	v_3
118.2	v_{24}
115.0	v_{10}
108.7	v_{23}
100.96	v_4
100.49	v_{20}
100.47	v_9
94.95	v_{25}
92.26	v_{19}
91.7	v_{14}
87.98	v_{21}
73.3	v_{17}
72.7	v_{16}
68.9	v_{11}
65.8	v_{15}
47.7	v_{22}
26.98	v_7
0.0	v_{12}

function (9) (compare the results in Table III). Test objects v_3, v_5 and v_6 have the hightest MS in the cluster groups shown in Fig. 3 and thus, they are chosen in test plan for test case tc_{24}. For tc_{24} the new test plan considers prototypes with several important equipment codes and two different vehicle bodies $l_{v_3} \neq l_{v_5} = l_{v_6}$ (compare Table IV).

The baseline model selects v_5, v_6 and v_{18} out of two cluster groups due to the fact that only an early scheduling date is important. By choosing prototype v_{18}, the baseline model accepts an earlier scheduling date than it is for v_3 in the new test plan. However, the baseline model selects prototypes with identical vehicle bodies $l_{v_5} = l_{v_6} = l_{v_{18}}$ (compare Table IV) such that requirement R_4 is not ensured.

3) Results of Quantitative Quality Assessment: To quantify the distribution of the presented procedure in comparison with the baseline model we evaluate QQM and $QQMW$ for tc_{24}.

Following (6) for test case tc_{24}, it holds that

$$\begin{aligned} d_{QQM}(tc_{24}) &= \mathbb{1}_{(b_{new,1}(tc_{24}) - b_{old,1}(tc_{24}) > 0)} \\ &+ \mathbb{1}_{(b_{new,2}(tc_{24}) - b_{old,2}(tc_{24}) > 0)} \\ &= \mathbb{1}_{((1-\frac{2}{3})>0)} + \mathbb{1}_{((\frac{2}{3}-\frac{1}{3})>0)} \\ &= 2 \end{aligned}$$

The evaluation of QQM in (7) and $QQMW$ for tc_{24} shows us quantitatively what we already described qualitatively

$$QQM_{tc_{24}} = QQMW_{tc_{24}} = 2$$

TABLE IV
RESULTS OF TEST PLANNING FOR TEST CASE tc_{24}: THE NEW TEST PLAN COVERS ALL NECESSARY REQUIREMENTS R_2, R_3, R_4, R_5, R_6. R_1 IS FULFILLED BY PREPROCESSING OF HIERARCHICAL CLUSTERING.

Requirements	**New Test Plan**	**Baseline Model**
R_6**: $t \leq 3$ Test objects**	v_3, v_5, v_6	v_5, v_6, v_{18}
R_4**: Cluster groups**	$t = 3$	$t - 1 = 2$
R_2**: Early scheduling date**	28.10.19, 22.10.19, 25.10.19	22.10.19, 25.10.19, 24.10.19
R_3**: Focus on important equipment codes**	Focused by weights W	Not focused
Special case: difference in vehicle bodies	$l_{v_3} \neq l_{v_5} = l_{v_6}$	$l_{v_5} = l_{v_6} = l_{v_{18}}$
R_5**: Larger number of tests per test object**	Focused by MS function	Not focused

For test case tc_{24} the presented procedure shows a high variability in comparison to the baseline model, both for the distribution of test objects to different cluster groups and for the choice of test objects with different vehicle bodies. Thus, QQM and $QQMW$ take the maximum value and the shown procedure can achieve the fulfillment of testing requirements $R_1 - R_6$ in the test plan, which in turn leads to a higher test quality. The proposed test planning procedure is preferable for test case tc_{24} and an optimized test planning procedure.

This is transferable to other test cases, as we can see in the stacked barplot of the Euclidean Norm of both b_{old} and b_{new} for all test cases TC with $m = 160$ (compare Fig. 4).

For all test cases the results are summarized in Table V.

B. Discussion of Results Explaining Higher Test Quality

The presented test planning procedure values the dissimilarity of prototypes expressed by equipment codes as well as different vehicle bodies out of t clusters when selecting test objects (compare Fig. 4). It fulfills all requirements $R_1 - R_6$ (compare Table IV) forming the basis for a higher test quality. To sum up, the procedure is preferable with respect to all test cases for QQM and $QQMW$ in comparison to the baseline model (compare Table V).

By early testing on maximally different test platforms in prototypes, engineers can achieve more errors due to equipment code dependent system influences. This enables the engineers to ensure a higher quality of their testing work by meaningful test results for the maturity process of the car line.

The new procedure for test planning helps to attain a higher quality in testing of customer experience vehicle functions. It is fast to conduct, easy to adapt, makes use of all necessary requirements $R_1 - R_6$ for test planning and the test distribution is more profound. The algorithm is integrated into the test management tool which is used by engineers to collaborate across the OEM. That is why there is a productive application of the procedure at the OEM.

1) Transferability: Transferability of the approach is ensured to any test planning scenario with a small set of

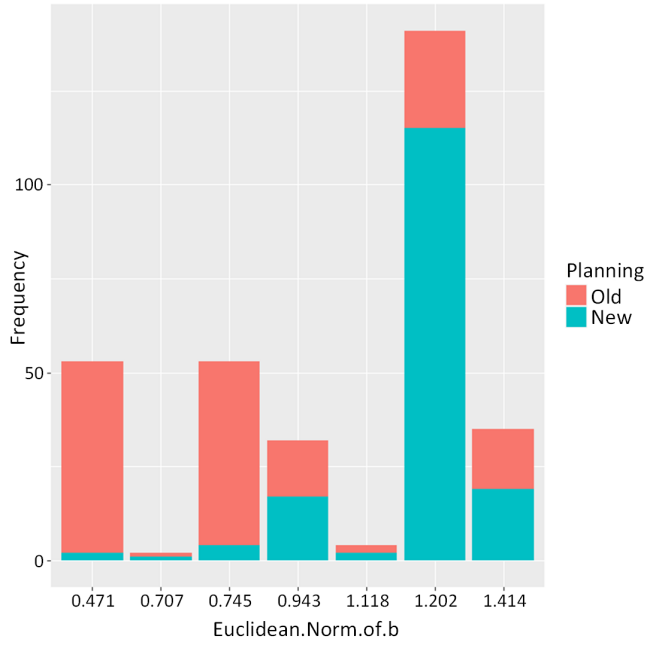


Fig. 4. Stacked barplot of Euclidean Norm of vectors b_{old} and b_{new} for all test cases: A higher Euclidean Norm of vector b_{new} than b_{old} stands for fulfillment of test plan requirements with respect to the proposed procedure. In general, the Euclidean Norm of b_{new} is higher than of b_{old} for the considered test cases. This ensures the fulfillment of requirements R_3, R_4 recommending the presented procedure.

TABLE V
RESULTS OF QQM AND $QQMW$

	Parameters				
	m	$\sum_{TC} d_{QQM}$	***QQM***	$\sum_{TC} d_{QQMW}$	***QQMW***
Values	160	189	1.181	320	2

test objects, weighted parameters and attributes that shall be ranked. The data science approach is not restricted to automotive software testing or automotive industry.

2) Scalability: Scalability is dependent on computational complexity of algorithms expressed in Big $\mathcal{O}$-notation. Hierarchical clustering requires a lot of computing resources if the sample size n of test objects is large because of at least $\mathcal{O}(n^2)$-runtime complexity depending on the concrete algorithm. Data scientists should study clustering algorithms with respect to efficiency if the sample size gets larger. Scalability is no issue for us since customer test planning is part of a human-driven process not being subject to real-time execution.

Due to the results of the presented procedure we can improve test quality by clustering-based customer experience vehicle function test planning with respect to the asymmetric metric QQM. This confirms our research question.

VI. CONCLUSION AND OUTLOOK

Artificial intelligence offers added value in many areas of automotive industry. In this paper, we described the development, implementation and successful evaluation of a procedure for a previously manual test planning of customer experience vehicle functions in the ramp-up phase of new car models to be automated by hierarchical clustering, weighted influencing parameters and ranked requirements with the goal to improve quality of testing. The procedure can be transfered to other data-driven scenarios in order to organize unlabeled data with requirements and ranking, because we did not stipulate any limitations. The individual parameters on clustering need to be adapted according to the concrete use case to achieve scalability. The presented procedure is a use case of how multi-dimensional test planning improves in speed and reproducibility with the help of machine learning.

There is a productive application of the procedure at the OEM. Engineers will use upcoming ramp-ups studying the extent to which the procedure will shorten the maturity cycles in the development process and thus, can save costs.

REFERENCES

[1] PricewaterhouseCoopers, "Five trends transforming the automotive industry," Tech. Rep., 2017-2018. [Online]. Available: https://www.pwc.com/gx/en/industries/automotive/assets/pwc-five-trends-transforming-the-automotive-industry.pdf
[2] M. Staron, *Automotive software architectures: An introduction*. Cham, Switzerland: Springer, 2017.
[3] A. K. Jain and R. C. Dubes, *Algorithms for clustering data*, ser. Prentice Hall advanced reference series. Englewood Cliffs: Prentice Hall, 1988.
[4] D. Xu and Y. Tian, "A comprehensive survey of clustering algorithms," *Annals of Data Science*, vol. 2, no. 2, pp. 165–193, 2015.
[5] C. M. Vong, P. K. Wong, and W. F. Ip, "Case-based classification system with clustering for automotive engine spark ignition diagnosis," in *2010 IEEE/ACIS 9th International Conference on Computer and Information Science*. IEEE, 2010, pp. 17–22.
[6] N. B. Satyendra and N. Nagaraja Prasad, "Functional sorting and prioritization of approval queues using multi objective optimizationtechniques," in *2019 IEEE International Conference on Electrical, Computer and Communication Technologies (ICECCT)*. IEEE, 2019, pp. 1–5.
[7] R. Ramler and C. Klammer, "Enhancing acceptance test-driven development with model-based test generation," in *2019 IEEE 19th International Conference on Software Quality, Reliability and Security Companion (QRS-C)*. IEEE, 2019, pp. 503–504.
[8] A. Larprattanakul and T. Suwannasart, "An approach for regression test case selection using object dependency graph," in *2013 5th International Conference on Intelligent Networking and Collaborative Systems*. IEEE, 2013, pp. 617–621.
[9] Z. Khalid and U. Qamar, "Weight and cluster based test case prioritization technique," in *2019 IEEE 10th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON)*. IEEE, 2019, pp. 1013–1022.
[10] M. J. Arafeen and H. Do, "Test case prioritization using requirements-based clustering," in *2013 IEEE Sixth International Conference on Software Testing, Verification and Validation*. IEEE, 2013, pp. 312–321.
[11] S. S. Barhate, "Effective test strategy for testing automotive software," in *2015 International Conference on Industrial Instrumentation and Control (ICIC)*. IEEE, 2015, pp. 645–649.
[12] M. Ackerman, S. Ben-David, S. Brânzei, and D. Loker, "Weighted clustering," 2011. [Online]. Available: http://arxiv.org/pdf/1109.1844v2
[13] C. Tofallis, "Add or multiply? a tutorial on ranking and choosing with multiple criteria," *INFORMS Transactions on Education*, vol. 14, no. 3, pp. 109–119, 2014.

Paper2

Modelling Production Workflows in Automotive Manufacturing

Simone König*,†, Birgit Vogel-Heuser (Senior Member IEEE)*, Etienne Fieg*, Michael Hahn†, Oliver Kopp†

*Chair of Automation and Information Systems, Technical University of Munich, Garching, Germany
†Mercedes-Benz AG, Sindelfingen, Germany

{simone.koenig, vogel-heuser, etienne.fieg}@tum.de, {michael.a.hahn, oliver.kopp}@daimler.com

2021 IEEE 23rd Conference on Business Informatics (CBI) | 978-1-6654-2069-3/21/$31.00 ©2021 IEEE | DOI: 10.1109/CBI52690.2021.10053

***Abstract*—Interest in applications based on business process modelling in Industry 4.0 is on the rise, with corporations worldwide shifting from simply exploring the potential of process models to creating productive business cases with executable process models in cloud computing. However, despite numerous applications existing in process modelling, engineers are still lacking proper guidelines for modelling production workflows in automotive diagnostics. To address the issue, we evaluated Business Process Model and Notation (BPMN) and Decision Model and Notation (DMN) as modelling notations for production workflows in assembly lines. Based on the results, this paper presents an evaluated guideline for modelling production car diagnostics workflows in BPMN 2.0 and DMN 1.3 in the automotive industry. Our work contributes to science, as we provide an overview and analysis of business process modelling in automotive testing and diagnostics.**

***Index Terms*—Automotive testing, manufacturing, production workflows, BPMN, DMN, guideline**

I. Introduction – Automotive Assembly Lines

In automotive manufacturing, cars are not only assembled, they also undergo a large number of diagnostic tests and commissioning of electronic components and control units (ECU) after mechanical completion for assembly protection [1]. Testing and commissioning include, e.g., checking whether the correct software is installed on an ECU or calibrating various sensors in the car. In doing so, the car passes through several individual process steps of diagnostic test and commission at up to 15 stations in the assembly line. These testing processes are directly related to the configuration of a car, e.g., a car without head-up display does not need head-up display calibration. Thus, the configuration set yields to different subsets of processes for testing and commissioning of a car. Executing these business processes in production workflows can bridge the business-production gap [2, 3]: The workflows use service tasks to send diagnostic commands to and retrieve diagnostic data from a car. Human-machine interaction between a workman and a car can be realised with user-guided tasks. As the business processes in automotive diagnostics can be generally modelled as flow diagrams, a visual representation of production workflows is a starting point. Thus, there is a basic interest in a guideline on how to model production testing workflows with business process modelling languages.

The main contribution of this paper is an evaluated guideline that helps engineers in automotive testing to model production workflows. The user of the guideline that will be created in this work is the engineer who will model the workflow. We identified Business Process Model and Notation (BPMN) [4] and Decision Model and Notation (DMN) [5] as graphical representations for specifying automotive manufacturing business processes that fulfil the business use case best. Based on business requirements and a SWOT analysis, we provide the users with a guideline to educate them modelling of the testing workflow. Thus, we formulate the research question RQ of this work as follows:

RQ: How is a guideline structured for modelling production workflows in testing and securing automotive diagnostics?

The remainder of this paper is structured as follows: In Section II, challenges and requirements in modelling production workflows are introduced. Section III considers related work. We identify the strengths, weaknesses, opportunities, and threats of the graphical representation of production workflows in automotive business processes in Section IV. In Section V, we propose a guideline for modelling production workflows in automotive testing and diagnostics. In Section VI, we evaluate and discuss the guideline. We conclude and outline further work in Section VII.

II. Challenges and Requirements in Modelling Production Workflows

In this section, we identify BPMN as the modelling notation standard for production workflows in automotive manufacturing. Furthermore, we identify requirements for a guide to model production workflows in automotive testing and diagnostics.

A. BPMN as Modelling Notation for Production Workflows

As diagnostic testing in automotive assembly lines is a business process to secure electrical and electronic assembly processes, we consider graph-based business process modelling languages. The underlying business process notation should follow an industry stand for executable process models. Information technology support and open source tools for workflow modellers and engines are important as well. This includes that the standard should be already applied in other industries as well. As production workflows are scheduled and orchestrated with respect to the car configuration, operations should have built-in parallelism and loops. The diagnostic workflows are realised in service tasks and send diagnostic commands to and retrieve diagnostic data from a car. An interaction with web services over Web Services Description Language (WSDL)

978-1-6654-2069-3/21/$31.00 ©2021 IEEE
DOI 10.1109/CBI52690.2021.10053

or http-based RESTful services is preferable considering the value of cloud computing in the automotive industry [6]. To include additional business processes with data in the assembly line and offer all process types, data objects, manual and user actions should be supported by the modelling notation as well.

In automotive diagnosis, ISO 13209 defines a standard for tester-independent, XML-based data exchange format for the documentation and formal description of test sequences in Open Test sequence eXchange (OTX) [7]. However, to keep open the possibility of integrating future cloud computing use cases in the automotive assembly line, we would like to take a more general notation. Thus, we choose BPMN 2.0 as the standard business process modelling notation for production workflows in automotive testing. In addition, DMN 1.3 assists in managing variants within testing workflows, as with BPMN complementary decision modelling notation.

B. Requirements R_1–R_8 for Modelling Guideline

To identify challenges in modelling production workflows, we interviewed four representative experts in automotive diagnostics. The experts are engineers working at an automotive manufacturing line and engineering development with expert knowledge in diagnostic testing and manual program scheduling. The experts identified relevant modelling challenges based on their expert knowledge. Building upon these challenges, eight requirements R_1–R_8 were derived for the modelling guideline of production workflows.

The production workflow is based on the corresponding test specification, e.g., written in current text documents and based on implicit engineer experience. Graphical flow charts of the workflows only exist in a few cases. Automated transformation from textual programming to graphical representation is difficult to be carried out due to various information sources and not fully standardised textual programming coding styles. The workflow must be modelled with the help of its specification and its source code as well as with the help of specialist knowledge (requirement R_1).

The variant handling of individual production workflows for manufacturing is carried out by a programmer with the help of the source code and the constraints of several test specifications. Building variants of workflows depends on experts' knowledge and textual programming for each car line (requirement R_2).

Production workflows can be executed and run both automatically and workman-guided. Workflows should cover manual, workman-guided, and automated process steps. A combination of the types is also possible. A graphically modelled workflow should link workman-guided tests to human-machine interaction (requirement R_3).

Production workflows are scheduled for a car. The constraints needed for scheduling workflows are available in implicit form in the current source code and the specification of the workflows. Today, engineers have no explicit overview of all criteria. The criteria might be extracted with data science methods or in the ramp-up phase of new car models. Criteria that are relevant for the schedule must be integrated into the process model. Based on the criteria, a schedule can be generated and which is optimised with respect to run time (requirement R_4).

There might exist no technically implemented control instance that prevents logical errors in the schedule. Logical implemented rules to guide the scheduling of test processes do not exist. The criteria on which the decision model is based must be complete and free of contradictions so that logical errors cannot occur (requirement R_5).

The variety of car equipment results in variants within the production workflow. The complexity of the model cannot be reduced at will, since executable code shall be generated from the process models directly. The complexity and the number of variants of the production workflow should be represented in the modelling without compromising clarity. The models should have multiple levels of hierarchy for different stakeholders. Users demand several hierarchies of modelling processes in workflows, i.e., call of diagnosis functions, standard modules like error reading, communication with human-machine interface (requirement R_6).

The process of modelling production testing workflows could take more time than traditional source code implementation. The modelling must occur with the help of predefined, detailed, modelled tasks at hierarchical level. If necessary, these tasks can be adjusted during modelling (requirement R_7).

Changes in textual programming of production workflows and schedules must be accepted by the workforce. Engineers concern that graphics is slower than textual programming. There already exists confirming historical literature on that hypothesis [8]. Modelling of production testing workflows should be faster, clearer, simpler, and easier to maintain to find acceptance. Engineers should be involved from an early starting point to avoid the rejection of the graphical modelling approach (requirement R_8).

Assuming the outlined requirements R_1–R_8, we develop a guideline to educate automotive engineers on the process of modelling production workflows. The engineer is the user of the guideline, who creates workflows for the assembly line either based on existing testing programs or created from scratch. The user crafts a BPMN process by modelling an automotive test or commissioning sequence.

III. Related Work

Current literature covers several guidelines that describe the process of model development in various application areas. Chae et al. [9] discuss the process for the treatment of coronary heart disease that is modelled with the help of BPMN and DMN, whereas Sooter et al. [10] discuss the medical decision-making process on whether the three-month injection is a suitable contraceptive method. Sooter et al. [10] model with the notations BPM and DM. Both medical processes are of lower complexity compared to the use case presented in this paper and can be reasonably modelled without the use of a hierarchy model. Dubani et al. [11] model a use case in the automotive industry: the process of component production in the interaction between an OEM and a supplier is extracted and then modelled with BPMN. This is a business process with

lower complexity, which is prototypically modelled at a high level without technical details on the executability of models.

Furthermore, literature exists that shows various possibilities of integrating DMN into BPMN models: Biard et al. [12] compare the decision processes modelled with BPMN with those in which DMN and BPMN are combined. Batoulis et al. [13] present various possibilities for reducing the complexity of models by using DMN in BPMN models. Complex BPMN structures are unambiguously assigned to DMN structures, which increase the clarity of the model with the same function. Three different approaches to the symbiosis of DMN and BPMN models for decision-based processes are presented by Hasic et al. [14]. In comparison with the presented paper, [12], [14], and [13] are not based on a concrete application.

Several collections of best practices of BPMN modelling exist in the current literature, e.g., by Silver [15]. The quality of existing models is assessed and, based on this, seven practical guidelines for improving model quality are presented by Mendling et al. [16]. However, the guidelines presented are generally valid and therefore not detailed enough for the use case at hand. Witsch [17] discusses a Manufacturing Execution System to increase the efficiency, quality, and cost of production processes. This use case is similar to the use case of this paper. A modelling language is presented for the Manufacturing Execution System developed, which is only based on BPMN. The modelling language is developed by Witsch and Vogel-Heuser [18] based on a theoretical model while comparing modelling languages such as BPMN and UML with it.

A summary of the related work is provided in Table I. There is no concept that describes the process of modelling production test and diagnostic workflows using BPMN and DMN in the automotive context. In the next section, we conduct a SWOT analysis to identify Strengths, Weaknesses, Opportunities, and Threats (acronym SWOT) of graphical modelling of production workflows in the automotive industry.

TABLE I
CLASSIFICATION OF RELATED WORKS (SYMBOL 'X' MEANS APPLICABLE, '(X)' PARTLY APPLICABLE, '–' MEANS NONAPPLICABLE OR NOT SPECIFIED)

Reference	Guideline	Application	BPMN	DMN	DMN instantiation for BPMN
Batoulis et al. [13]	–	Universal	–	x	–
Biard et al. [12]	–	Universal	–	x	–
Chae et al. [9]	x	Medical	x	x	–
Dubani et al. [11]	x	Automotive	x	–	–
Hasic et al. [14]	–	Universal	x	x	–
Mendling et al. [16]	–	Universal	x	–	–
Sooter et al. [10]	x	Medical	x	x	–
Witsch and Vogel-Heuser [18]	x	Industrial production	(x)	–	–
Witsch [17]	–	Industrial production	(x)	–	–
This paper	x	Automotive	x	x	(x)

IV. SWOT ANALYSIS: GRAPHICAL MODELLING OF PRODUCTION WORKFLOWS

A SWOT analysis is a strategic planning technique to assess a project from different points of view. Therefore, strengths and weaknesses (internal factors), and opportunities and threats (external factors) are analysed. In this paper, the aim of the analysis is to benefit from the strengths of graphical modelling for the production environment and to develop measures to circumvent existing weaknesses. By combining the strengths, weaknesses, opportunities and threats to each other in a matrix, recommendations for action can be derived, which can be used to optimise the transformation of test routine development. In the proceeding section, the guideline for modelling production workflows in automotive testing and diagnostics provides solutions for the requirements R_1–R_8 with respect to the results of the SWOT analysis. We performed structured interviews with experts on diagnostic testing in automotive manufacturing.

A. Combination of strengths and opportunities

The engineering process of test routine software should be adapted from status quo to exploit the full potential of graphical modelling. Time savings and reduced complexity in the software development of test routines, as well as automated generation of machine-readable code for car models can reduce workforce capacity in automotive companies. Executable process models can bridge the business-production gap.

B. Combination of strengths and threats

The workflow execution system should be adapted to the interface of the workflow management system in which the modelling is implemented to enable optimal integration. This end-to-end integration bridges the business-IT gap. The modelling of test routines should be started as the first part of the overall transformation process, as this step is particularly time-consuming when there is a large volume of different test routines.

C. Combination of weaknesses and threats

Extensive staff training on modelling and uniform guidelines must be established in traditional automotive companies. This is the only way to ensure a consistently high quality of the models and acceptance in the long term.

D. Combination of weaknesses and opportunities

As many different models as possible should be made accessible in a repository to ensure continuity and consistent model quality. At the same time, the modelling duration of a workflow can be shortened in this way. Furthermore, the knowledge is made accessible in an explicit form in the repository, e.g., for new car lines.

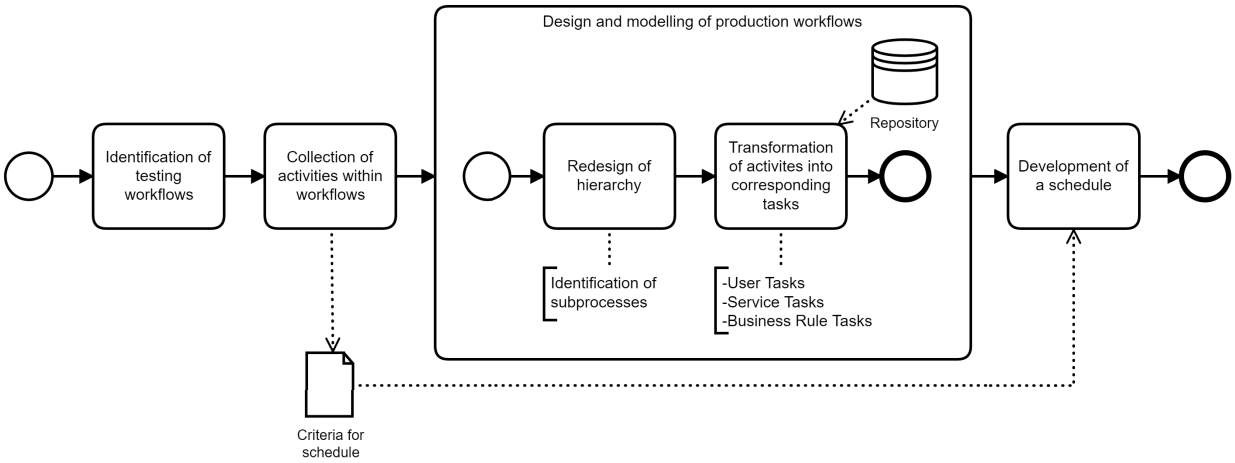


Fig. 1. Guideline for modelling production testing and diagnostic workflows

V. Guideline for Modelling Production Workflows in Automotive Testing and Diagnostics

In the following, the guideline for modelling production workflows in automotive testing in assembly lines is explained. The guideline is built on BPMN 2.0 and DMN 1.3 as modelling notations, based on the requirements R_1–R_8 and the action portfolio of the SWOT analysis. Fig. 1 illustrates the corresponding guideline steps represented in BPMN for the user.

A. Identification of Testing Workflows

Identifying a suitable testing process to be modelled and executed in a proceeding step, the responsible engineer captures the information needed to model. The information can be found in textual programs, text documents, and in expert discussions. Test processes with a similar technical purpose should be combined in a model as a standard, while large test processes should be divided into several models.

B. Collection of Activities within Workflows

To get the content of the identified testing workflows in detail, the testing process should be collected in the context of an expert discussion with the responsible developer. Based on this, the testing process is divided into individual activities. Furthermore, all criteria required for variant management and scheduling in the decision model should be extracted from this.

C. Design and Modelling of Production Workflows

The production testing workflows are modelled using the BPMN 2.0 standard. A guide to descriptive BPMN modelling is illustrated by Silver [15]. The activities are transformed into corresponding BPMN tasks and then linked to a model. When designing the testing workflow, a structure must be chosen that fits the modelling, but also correctly and completely maps the functional scope of the existing testing process. Workman-guided activities should be modelled using user tasks in BPMN, automated activities using service tasks in BPMN, and decisions using business rule tasks in gateways or in DMN.

For a better overview and better understanding of the technical hierarchies in the graphics involved, the modelling should be divided into several hierarchical levels. At the top level, the business process of the testing workflow is roughly outlined. With the level of detail increasing over several levels, the level of detail required for implementation is achieved on the lowest level, which requires extensive technical understanding. Collapsed subprocesses in connection with call activities should be used for the implementation of hierarchies.

Frequently occurring subprocesses should be modelled separately from the testing workflow and made available in a repository of a modelling palette. The integration into the testing workflow again occurs on a subprocess with the call activity. When modelling the subprocesses, it must be ensured that they can be parameterised. Calling the subprocess, the workflow-dependent parameters are transferred. Variant handling can be expressed with DMN.

D. Development of a Schedule

The criteria extracted from the production testing workflows are the basis for a decision model that can assist the development of an optimised schedule of workflows. The decision model is two-stage, with the first part of instantiation for BPMN using the DMN standard. The second part should be implemented with the help of an algorithm to solve optimisation problems and will not be considered further in this paper.

The Decision Requirements Graph (DRG) [5] of the first part consists only of a decision table at the decision logic level. For each production testing workflow, a corresponding row should be created in the decision table and the extracted criteria should be entered as input in columns. The ID of the respective model is used as output. In each case, Friendly Enough Expression Language (FEEL) [5] is used describing input and output, Collect should be used as a hit policy. For

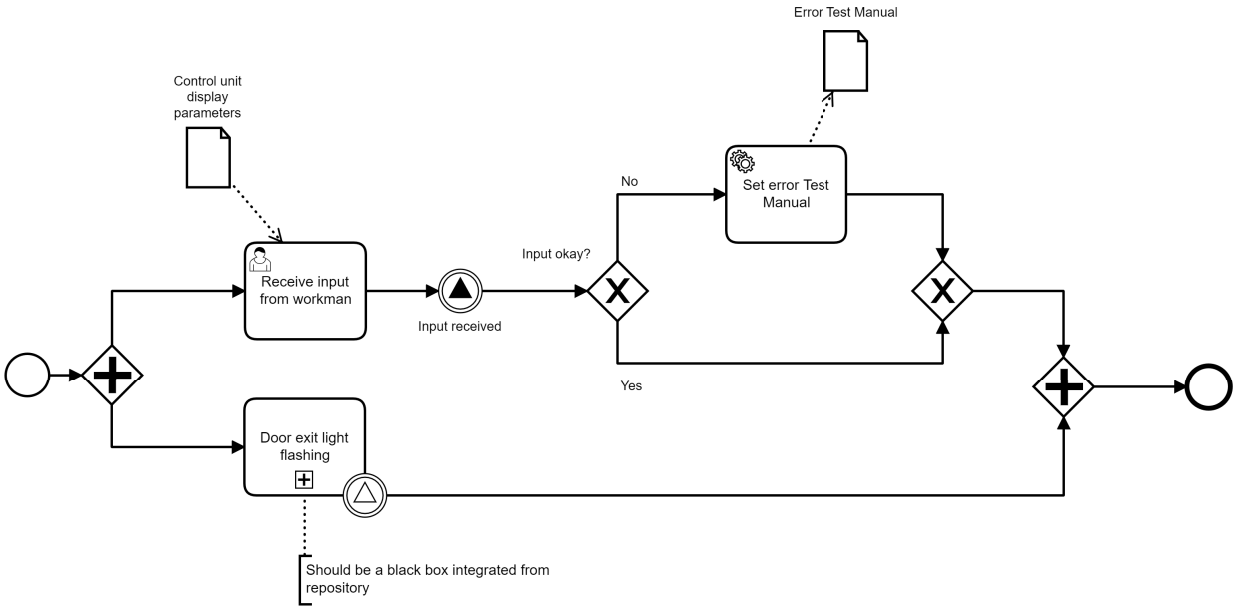


Fig. 2. Reference model of *manual* test process *door exit light* for qualitative validation of the presented guideline

any point in time in the schedule, the decision model outputs the IDs of all workflows that can be carried out based on the characteristics of the criteria. If the decision model is called sequentially for several points in time, a schedule results from sequencing the outputs. If the output consists of several IDs at a certain point in time, the explicit schedule must be determined by the second part of the decision model.

VI. Evaluation of Guideline

The presented guideline for modelling production workflows in automotive testing has been evaluated experimentally. Camunda Modeler, as part of the open source modelling platform, was used for modelling processes [19]. To conclude this section, the guideline is assessed against the formulated requirements of Section II-B on a per-concept basis.

A. Description of the Experimental Design

A two-stage test design validates qualitatively and quantitatively the results of automotive experts:

- **Two-stage test design** with qualitative and quantitative validation of the results
- **Four representative experts** from Mercedes-Benz AG who are users of the guideline as test persons
- **Qualitative evaluation** of production testing workflows, modelled by the test persons using Camunda Modeler, the source code, the specification of the test, and their own specialist knowledge. Therefore, comparison with the reference model has been done alongside the individual steps of the guideline and the requirements of Section II-B.
- **Quantitative evaluation** of the guideline using an assessment of structured statements by the test persons concerning the scope, level of detail, and quality of the guideline.

B. Results on Two-Stage Test Design

In the evaluation of the guideline, a diagnostic test called *door exit light* test has been chosen since it contains diagnosis, automated and workman-guided elements. Whether the test is performed or not, it depends on the car's configuration set. We build a reference model that helps us to compare the test persons' results. The reference model of the manual subprocess is illustrated in Fig. 2. In the first step of the *door exit light* test, it is either checked manually by a workman whether the *door exit lights* are active when the corresponding function in ECU *ecu1* for all doors is activated or, in the case of an automatic test, it is determined whether the errors are set in the error memory of the corresponding control unit during activation. To differ between automation or workman guidance for each car dependent on a special car configuration code *code1* and the selected front door *door left front (DLF)* or *door right front (DRF)*, a DMN decision model to instantiate the BPMN model is created in Fig. 3. The DMN model is integrated via a business rule task. The automated and workman-guided tests are modelled as collapsed subprocesses with call activity. The process is carried out for all four doors one after another. Within the manual part, the *door exit light* flashes until the workman makes an input to the control unit. A possible error of the *door exit light*, which is derived from the workman's input, is documented and then, the submodel is terminated.

C. Experts as Subjects

The sample size of four subjects is rather small. To achieve a representative result, particular attention was paid to their

DMN instantiation: Automated or workman-guided | Hit Policy: Rule order

	When Order_Code string	And Name_ECU string	Then Test_Type string	Annotations
1	code1	ecu1-DLF	auto	Test on DLF can be automated when code1 is present.
2	code1	ecu2-DRF	auto	Test on DRF can be automated when code1 is present.
3	-	-	manual	In other cases, test is manually executed by workman.
+				

Fig. 3. DMN instantiation of the BPMN workflow *door exit light* test for a car and presented in Camunda Modeler – DLF means *door left front* and DRF *door right front)*

characteristics when selecting representative subjects. This includes age, gender, academic background, and professional experience. All subjects can serve as users of the guideline because of their profession. Since comparable characteristic structures are established in many companies in the European automotive industry, it can be assumed that the present result would be similar for various European automotive OEMs.

Two subjects could not make the models, but were available for the final discussion. Accordingly, only the models of two subjects are used for the qualitative evaluation. Thus, in the quantitative evaluation, anomalies occur in the results of the structured interviews.

D. Qualitative Validation

Identification of testing workflows

Correctly identifying suitable tests could only be checked to a small extent in this validation. For reasons of comparability of the models between the test persons, all test persons were provided with the source code and the specification of only one test. On the basis of this information, all subjects decided to model the content of the existing test in a single workflow.

Collection of activities within workflows

Due to their expert knowledge, the test persons did not have to conduct any further expert discussions to get a better understanding of the content of the test process. However, the extracted activities differ between the models of the subjects: Compared to the reference model, one subject has advanced to a deeper level of detail for some activities, as shown in Fig. 4. Furthermore, compared to the reference model, the subject made a distinction when performing the test between a right-hand drive and a left-hand drive.

Design and modelling of production workflows

Both test persons did not fully map the functional scope of the *door exit light* test, but only displayed the manual part of the test. Furthermore, when transforming the activities into tasks, both test persons did not specify the tasks more precisely. Thus, user tasks, service tasks, and business rule tasks remained unused. Both test persons did not split their models into several hierarchical levels. It should be noted at this point that test persons did this on purpose due to the lack of a workflow engine for execution in the evaluation setting.

Development of a schedule

The development of a test schedule was not considered in the context of this validation as the external algorithm for run time optimisation has not been integrated yet.

E. Quantitative Validation

Table II shows the statements that the test persons evaluated in the context of quantitative validation. In Fig. 5, the evaluations of the test persons are assigned to the statements on a scale from 0 (complete rejection) to 5 (complete agreement).

F. Discussion of Evaluated Guideline

It can be stated that the guideline and the underlying concept were good to understand for the test persons except for one outlier (statement 1). All test persons also stated that they were able to apply the concept presented in the guideline very well in the context of modelling (statement 2). In comparison with the reference model, however, it is noticeable that there are discrepancies between the subjects' models and the reference model. Reasons can be the lack of understanding of the concept summarised in the guideline including the standard BPMN elements, missing modelling palettes, and the use of Camunda Modeler as a software platform. Clear assistance for the modelling could mostly be extracted from the guideline (statement 3). Reasons against a higher approval are that the guideline neither covers graphical modelling in detail nor detailed diagnostic service implementations.

There is a large discrepancy between the subjects regarding open questions in the modelling (statement 4). The reason for the strong discrepancy, which is shown by the high variance of the results of statement 4, could be the subjective perception of the experts' own understanding of the guideline.

The test persons stated that the level of detail in the guideline is sufficiently high for modelling (statement 5). In the models of the subjects, however, the essential details that were presented in the guideline, such as the specification of the tasks or the use of several hierarchical levels, are not taken into account. This could be due to a lack of understanding of the graphical modelling theory used in the guide. One test person rated the scope of the guideline as clearly too small (statement 6). The reason for this was primarily the personal expectation of the

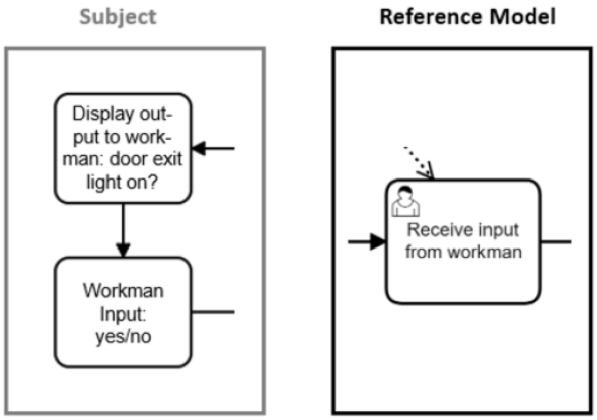


Fig. 4. Comparison of activities classified with different levels of detail between a subject and the reference model

test person. The contents, ideas, and concepts presented in the guidelines were already known, at least in part to the subjects (statement 7). The reason for this could be that some test persons have already come into contact with the automotive industry standard OTX.

Test persons' options are split in two, whether the concept behind the guideline is suitable or not as a basis for the redesign of the test procedure development (statement 8): This is shown by the high variance of the results of Statement 8 in Fig. 5. On the one hand, the models can be used both for documentation and as a basis for the execution of production testing workflows, which means significantly less effort in development and maintenance compared to the previous process. On the other hand, providing an execution engine is complex and employees specialised in modelling are required as well.

G. Assessment of Requirements for the Guideline

In this section, the developed guideline is assessed against the formulated requirements R_1–R_8 of Section II-B on a per-concept basis. Table III provides a summary of the assessment.

Production workflows for testing and commissioning can be fully implemented on a theoretical and conceptual level with the help of the standards BPMN and DMN (requirements R_1, R_2), apart from the more complex scheduling problem (requirements R_4). DMN assists in variant handling with respect to car configuration codes. Scheduling of production workflows needs many boolean rules that are input for an optimisation algorithm, such that DMN cannot assist efficiently. To achieve a run time-optimised schedule of workflows for each car, an optimisation algorithm needs to be implemented based on Boolean rules and integrated externally.

It has been shown during the validation that both workman-guided and automated checks can be modelled with the underlying concept. The specification of a check and the existing source code are sufficient for modelling the check module. Requirement R_3 is fulfilled.

TABLE II
REPRESENTATION OF STATEMENTS THAT SUBJECTS RATED DURING GUIDELINE VALIDATION

No.	Statement
1	The concept for graphical modelling of workflows summarised in the guideline was understandable for me.
2	I was able to apply the concept presented in the guide to my modelling style.
3	I was able to extract clear assistance for the modelling from the guide.
4	I did not have any open questions about the modelling.
5	The guide has a sufficiently high level of detail as an aid to modelling.
6	The scope of the guideline is sufficient as an aid for the modelling.
7	The guide contains ideas, concepts and information that I was not aware of before.
8	I consider the guideline – and the concept of executable graphical process models behind it – to be a suitable basis for redesigning test procedure software development.

Not all contents presented in the concept, such as the hierarchy level model for the differentiated representation of the model complexity, are self-explanatory. Without special user training, it is not possible to meet requirement R_6. A successful integration of the diagnostics back end is of great relevance for the practical benefit and user acceptance of the concept as well (requirement R_5): A suitable execution engine for the process models is of further research interest.

The user-friendly model implementation depends a lot on the software application used for graphical modelling. In this work, Camunda Modeler has been used. Evaluation showed that a modelling palette is necessary and should be integrated in a suitable modelling tool. Users reported that there is potential in software ergonomics of integrating subprocesses: navigation through the hierarchy and subprocesses can be realised, e.g., by a double click of the computer mouse. Thus, there is potential for further research to fully meet the requirement R_7. The appropriate training of the employees is of great relevance – both regarding the software application and regarding the concept developed in this work. The evaluated guideline in this work is a starting point (requirement R_8). To fulfil both requirements R_7 and R_8 in the transformation process precisely, the concepts of user experience design and software ergonomics should be included as outlined in the interpretation of the SWOT analysis. Variant handling in DMN may be one possibility in BPMN models such that there is more research to be done.

It can be concluded that the guideline can be used to introduce and guide through graphical modelling of testing routines in an automotive manufacturing system. The guideline supports the transformation process from textual programming to graphical modelling.

Without the theoretical basics of the current processes of test routine development, the modelling standards BPMN and DMN as well as in-depth knowledge of the used software

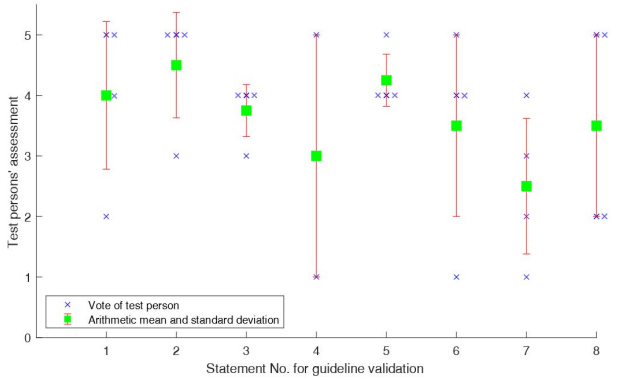


Fig. 5. Representation of the test persons' assessment of statements with the following ascending scale: 0 representing full rejection and 5 full agreement

TABLE III
SUMMARISED ASSESSMENT OF THE GUIDELINE

No.	R_1	R_2	R_3	R_4	R_5	R_6	R_7	R_8
Guideline	√	√	√	-	-	√	o	√

Legend: √ Requirement fulfilled
o Requirement partly fulfilled
- Requirement not fulfilled

application, a sufficient understanding of the guideline is not possible. It should be noted that the modelling of tests on the basis of the specification and source code generally has a high degree of freedom. The validation showed that both workman-controlled and automated tests can be modelled. The expert specification of a test and the existing source code are sufficient for modelling of production testing workflows.

The guideline shows the structure of how modelling production workflows in testing and securing automotive diagnostic processes in manufacturing lines can be performed. Thus, the research question RQ is discussed with additional insights into further research topics given.

VII. Conclusion and Outlook

During the fourth industrial revolution, automotive companies are facing considerable challenges. Since Industry 4.0 solutions for manufacturing transit gradually over time, the interaction between legacy production systems and new cloud computing solutions is indispensable. To provide an end-to-end integration of the modelling and execution of a diagnostic test using graphically modelled processes, engineers demand proper guidelines for modelling production workflows in automotive commissioning and testing.

This paper filled the gap by presenting an evaluated guideline for engineers to model production workflows in BPMN 2.0 and DMN 1.3 in automotive assembly lines. We performed a SWOT analysis with structured expert interviews that demands a transformation process from textual programming to graphical modelling of production workflows. The workflows are using BPMN service tasks and can send diagnostic commands to and retrieve diagnostic data from a car. DMN assists in variant handling with respect to configuration codes. The major challenge within this work was to discuss an integration concept for existing testing modules and schedules using graphical programming. Different hierarchy levels realised in subprocesses play a crucial role as well as the integration of service tasks in model palettes.

In future work, we study modelling tools and create palettes for production workflows in the automotive industry.

References

[1] M. A. Omar, *The automotive body manufacturing systems and processes*. Chichester: Wiley, 2011.

[2] F. Leymann and D. Roller, *Production Workflow: Concepts and Techniques*. Prentice Hall PTR, 2000. [Online]. Available: https://books.google.de/books?id=Xc8eAQAAIAAJ

[3] M. Wieland, O. Kopp, D. Nicklas, and F. Leymann, "Towards context-aware workflows," in *CAiSE07 proc. of the workshops and doctoral consortium*, vol. 2, no. S 25. Citeseer, 2007, p. 78.

[4] Object Management Group (OMG), "Business Process Model And Notation (BPMN), Version 2.0," 2011.

[5] Object Management Group, "Decision Model And Notation (DMN), Version 1.3," 2020.

[6] C. Pautasso, "Bpmn for rest," in *Business Process Model and Notation*, ser. Lecture Notes in Business Information Processing, R. Dijkman, J. Hofstetter, and J. Koehler, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, vol. 95, pp. 74–87.

[7] "Road vehicles Open Test sequence eXchange format – ISO 13209," 2012.

[8] M. Petre, "Why looking isn't always seeing: Readership skills and graphical programming," *Commun. ACM*, vol. 38, no. 6, p. 3344, Jun. 1995. [Online]. Available: https://doi.org/10.1145/203241.203251

[9] J. Chae, B. H. Park, M. Jones, M. Ward, and J. Nebeker, "Converting clinical pathways to bpm+ standards: A case study in stable ischemic heart disease," in *2020 IEEE 33rd International Symposium on Computer-Based Medical Systems (CBMS)*. IEEE, 72020, pp. 453–456.

[10] L. J. Sooter, S. Hasley, R. Lario, K. S. Rubin, and F. Hasić, "Modeling a clinical pathway for contraception," *Applied clinical informatics*, vol. 10, no. 5, pp. 935–943, 2019.

[11] Z. Dubani, B. Soh, and C. Seeling, "A novel design framework for business process modelling in automotive industry," in *2010 Fifth IEEE International Symposium on Electronic Design, Test & Applications*. IEEE, 012010, pp. 250–255.

[12] T. Biard, A. Le Mauff, M. Bigand, and J.-P. Bourey, "Separation of decision modeling from business process modeling using new "decision model and notation" (dmn) for automating operational decision-making," in *Risks and Resilience of Collaborative Networks*, ser. IFIP Advances in Information and Communication Technology, L. M. Camarinha-Matos, F. Bénaben, and W. Picard, Eds., vol. 463. Cham: Springer International Publishing, 2015, pp. 489–496.

[13] K. Batoulis, A. Meyer, E. Bazhenova, G. Decker, and M. Weske, "Extracting decision logic from process models," in *Advanced Information Systems Engineering*, ser. Lecture Notes in Computer Science, J. Zdravkovic, M. Kirikova, and P. Johannesson, Eds. Cham: Springer International Publishing, 2015, vol. 9097, pp. 349–366.

[14] F. Hasic, J. de Smedt, and J. Vanthienen, "Redesigning processes for decision-awareness: Strategies for integrated modelling," in *2018 11th International Conference on the Quality of Information and Communications Technology (QUATIC)*. IEEE, 92018, pp. 247–250.

[15] B. Silver, *BPMN method and style*. Aptos, CA: Cody-Cassidy Pr, 2009.

[16] J. Mendling, H. A. Reijers, and W. van der Aalst, "Seven process modeling guidelines (7pmg)," *Information and Software Technology*, vol. 52, no. 2, pp. 127–136, 2010.

[17] M. Witsch, *Funktionale Spezifikation von Manufacturing Execution Systems im Spannungsfeld zwischen IT, Geschäftsprozess und Produktion*, 1st ed. Göttingen: Sierke, 2014.

[18] M. Witsch and B. Vogel-Heuser, "Towards a formal specification framework for manufacturing execution systems," *IEEE Transactions on Industrial Informatics*, vol. 8, no. 2, pp. 311–320, 2012.

[19] "Camunda inc." (last accessed: 11-August-2021). [Online]. Available: https://camunda.com/products/camunda-platform/modeler/

Paper3

Material from: S. König, M. Reihn, F.G. Abujamra *et al.,* “Flexible scheduling of diagnostic tests in automotive manufacturing”, *Flexible Services and Manufacturing Journal, 2023, Springer Nature.*

Flexible Services and Manufacturing Journal (2023) 35:320–342
https://doi.org/10.1007/s10696-021-09438-3

Flexible scheduling of diagnostic tests in automotive manufacturing

Simone König[1,2] · Maximilian Reihn[2] · Felipe Gelinski Abujamra[2] · Alexander Novy[2] · Birgit Vogel-Heuser[1]

Accepted: 6 December 2021 / Published online: 16 January 2022

Abstract

The car of the future will be driven by software and offer a variety of customisation options. Enabling these customisation options forces modern automotive manufacturers to update their standardised scheduling concepts for testing and commissioning cars. A flexible scheduling concept means that every chosen customer configuration code must have its own testing procedure. This concept is essential to provide individual testing workflows where the time and resources are optimised for every car. Manual scheduling is complicated due to constraints on time, predecessor-successor relationships, mutual exclusion criteria, resources and status conditions on the car engineering and assembly line. Applied methods to handle the mathematical formulation for the corresponding industrial optimisation problem and its implementation are not yet available. This paper presents a procedure for automated and non-preemptive scheduling in the testing and commissioning of cars, which is built on a Boolean satisfiability problem on parallel and identical machines with temporal and resource constraints. The presented method is successfully implemented and evaluated on a variant assembly line of an automotive Original Equipment Manufacturer. This paper is the starting point for an automated workflow planning and scheduling process in automotive manufacturing.

Keywords Non-preemptive scheduling · Multiple constraints · Boolean satisfiability problem · Automotive testing · Car manufacturing

✉ Simone König
simone.k.koenig@daimler.com

[1] Institute of Automation and Information Systems, Technical University of Munich, Munich, Germany

[2] Mercedes-Benz AG, Böblingen, Germany

1 Introduction and motivation

The automotive industry is affected by the increasing importance of automotive software (Ebert and Favaro 2017). Automotive software allows customers to use more assisted, software-linked car functions during their car journey. These software-linked customer car functions are related to components and electronic control units (ECUs), such as the head-up display. The functions support new concepts for automotive software engineering in manufacturing: the Original Equipment Manufacturers (OEM) offer their customers flexible customisation possibilities and equip the manufacturing processes accordingly.

Automotive manufacturing should therefore be able to produce, test, and commission software-linked, customised cars. As every chosen configuration code may need its own testing procedure to complete the car, such as the commissioning of a seat massage, it is essential to provide time and resource-optimised individual testing workflows for any given car. Those testing workflows consist of software and worker-guided process steps. Nowadays, all of these tests are written into a schedule by hand with a preferably short test time. To minimise factory costs, the schedule of the test procedures could be optimised with respect to the total test time within a factory by applying an automated scheduling procedure.

The main contribution of this paper is a method for the automated and flexible scheduling of diagnostic tests on cars subject to several constraints. The constraints relate to time, direct predecessors, set of all predecessors, mutual exclusion criteria, resource and status conditions for the car engineering and assembly line. The corresponding mathematical model can be formulated as a Boolean satisfiability problem on parallel identical machines. This is the first use case to describe flexible scheduling on parallel identical machines with complex constraints for testing and commissioning in car manufacturing.

The paper is structured as follows: Section 2 introduces the background to scheduling, business challenges and deduces resulting requirements of scheduling for testing and commissioning in automotive manufacturing systems. Section 2 hereby outlines the research questions of this work. The related work is discussed in Section 3. Section 4 describes the mathematical model for scheduling diagnostic tests. A novel automated and flexible scheduling procedure based on the mathematical model is presented in Section 5. The computational study is illustrated in Section 6. The results of the study are presented and discussed in the following Section 6. The final Section 7 concludes this paper.

2 Problem definition

Section 2 introduces the basic concept of production planning in manufacturing. We explain the technical requirements arising from expert discussions that need to be fulfilled for scheduling in testing and commissioning. We can then select an appropriate mathematical concept based on the defined technical requirements.

2.1 Background to scheduling and real-time operating systems

The process of translating customer orders into manufacturing jobs on specific dates is called production planning (Lawlor 1973). Production planning creates a plan that describes the necessary, ordered set of jobs required to accomplish a specific goal in terms of customer orders, given certain starting conditions. Scheduling is a subprocess of production planning, dealing with the allocation of resources to jobs over given time periods with respect to the optimisation of one or more objectives (Błażewicz 2007). A typical optimisation problem in industry is minimising the makespan of a job schedule, which can be interpreted as the time needed from the start to the end of the schedule (Hillier and Herrmann 2006). Scheduling is broadly applied in open, flow and job shop production problems (Brucker 2007).

Scheduling in manufacturing can be interpreted as non-preemptive scheduling in real-time operation systems (Quilliot et al. 2021). To narrow down related work, we continue with the industrial challenges and derive requirements from these.

2.2 Challenges and requirements in the complex scheduling of diagnostic tests

Modern automotive manufacturers offer their customers a wide range of car customisation options. A fixed production plan is therefore no option in flexible manufacturing systems, or in standardised car diagnostic testing schedules for electric and electronic car components (challenge C_1). As every choice of configuration set may need its own testing procedure, manufacturing systems must provide time and resource-optimised individual testing schedules for any given car (challenge C_2).

Fig. 1 shows three exemplary testing scenarios on an automotive assembly production line. In test location 1, the maximum number of tests is conducted on the fully equipped car with seven scheduled tests. A fully equipped car has more electrical and electronic components than others and thus, for cars with less

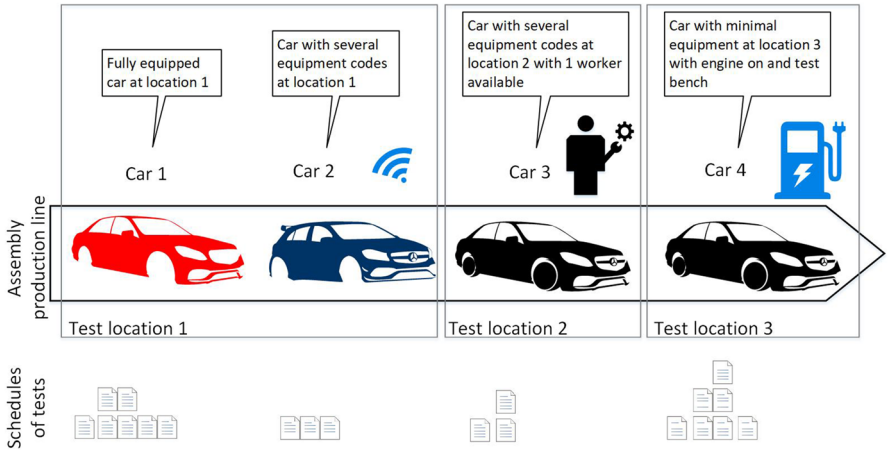


Fig. 1 Scheduling scenarios in automotive assembly lines - a document icon symbolises a diagnostic test

equipment codes, fewer tests need to be scheduled. In test location 2, there is a schedule with a worker-guided test. In test location 3, the smallest scope of tests is conducted on the cars with the least equipment. The tests are performed on a car with motor status *on* and in communication with a test bench.

The tests should be performed in parallel at all test locations to minimise testing run times if feasible. In the following, we will take a closer look at the constraints that decide on the parallel execution of tests.

Expert discussions were held with engineers and programmers to address the challenges C_1 and C_2 and resulted in six requirements for the creation of a procedure to schedule diagnostic tests in automotive manufacturing. We introduced a flexible manufacturing system (FMS) to handle challenge C_2: a FMS is an integrated group of processing computer, numerical control machines and material handling equipment controlled by computers for the automatic processing of parts (ElMaraghy and Caggiano 2016). In an FMS, testing schedules consist of equipment-dependent tests and can be computed event-driven.

Six technical requirements will be formulated with mathematical expressions in the following to handle challenge C_1. These explicit requirements arise from expert knowledge in automotive diagnostic test scheduling. The baseline scheduling model is built by an engineer who writes all tests into a software sequence schedule by hand, preferably with a short test time. It is based on experience.

Schedules were either created on demand or beforehand to plan machine usage better. Thus, the following explicit requirements form the technical basis of the mathematical optimisation model for flexible scheduling in this work. To formulate the constraints, we list all of the basic mathematical notation of this work in Table 1.

Table 1 Relevant notation in this work

$I = \{1, \dots, N\}$	Set of $N \in \mathbb{N}$ tests
$K = \{1, \dots, S\}$	Set of $S \in \mathbb{N}$ machines
$J = \{1, \dots, K\}$	Set of $K \in \mathbb{N}$ resources
$L = \{1, \dots, Z\}$	Set of $Z \in \mathbb{N}$ status conditions of objects
$\{t_i\}_{i \in I}$	$t_i \in \mathbb{R}_{>0}$ units of time that test $i \in I$ takes
$\{pre_i\}_{i \in I}$	$pre_i \in I$ direct predecessor of test $i \in I$
$\{P_i\}_{i \in I}$	$P_i \subset I$ set of predecessors that must end before test $i \in I$ can start
$\{M_i\}_{i \in I}$	$M_i \subset I$ set of tests that may not run parallel to test $i \in I$
$\{r_{i,j}\}_{i \in I, j \in J}$	$r_{i,j} \in [0, 100]$ resources temporarily taken by test $i \in I$ on resource $j \in J$
$\{c_{i,l}\}_{i \in I, l \in L}$	$c_{i,l} \in \{turn\_on, turn\_off, req\_on, req\_off\}$ condition status of test $i \in I$ on kind of condition $l \in L$
$\{s_i\}_{i \in I}$	$s_i \in \mathbb{R}$ start time of test $i \in I$
$\{e_i := s_i + t_i\}_{i \in I}$	$s_i \in \mathbb{R}$ end time of test $i \in I$
$\{T_i := [s_i, e_i]\}_{i \in I}$	$T_i \in \mathbb{R}^2$ time interval of test $i \in I$
$e_{\max} = \max_{i \in I} \quad e_i$	Makespan, the total length of a schedule
C	Set of constraints
$\Theta_C \in \mathbb{R}_{\geq 0}^{\lvert I \rvert}$	Feasible solution for a given set of constraints C

We define the finite set of N individual tests that are scheduled for a car at an automotive assembly line as

$$I = \{1, \ldots, N\}, \quad N \in \mathbb{N},$$

and the finite set of K different resources that may be used by a test $i \in I$ as

$$J = \{1, \ldots, K\}, \quad K \in \mathbb{N}.$$

For example, the specific test $i \in I$ may use 20% of the car engine's resource capacity. The finite set of Z different status conditions, which must be considered in the test schedule, is defined as

$$L = \{1, \ldots, Z\}, \quad Z \in \mathbb{N}.$$

For example, the car engine must be turned *on* before testing the air conditioning. The respective test would require the motor status to be *on* as the status condition.

Without loss of generality, we formulate the requirements R_{time}, R_{dir}, R_{pre}, R_{mutex}, R_{res} and R_{status} for the test $i \in I$.

- Requirement R_{time}: a test has a run time
  Test $i \in I$ has a predicted run time $\{t_i\}_{i \in I},\ t_i \in \mathbb{R}_{>0}$. The prediction can be based on expert knowledge and an analysis of historic tests. The run time t_i is a crucial requirement to minimise with respect to the makespan of the test schedule. Note that the more accurate the time predictions are, the more efficient an optimised test schedule will be in the production line.
- Requirement R_{dir}: a test can have a direct predecessor
  Test $i \in I$ may require a test as a direct predecessor $\{pre_i\}_{i \in I},\ pre_i \in I$. Therefore, the end time of pre_i must equal the start time of i. For example, the engine must be brought up to a certain temperature before it is tested under maximum load. There must be short time between the test of heating and the main test of the engine itself.
- Requirement R_{pre}: a test has a set of predecessors
  Test $i \in I$ may require a set of predecessors $\{P_i\}_{i \in I},\ P_i \subset I$, which must be completed before i starts. For example, an ECU should be flashed by software before it is coded to a specific configuration.
- Requirement R_{mutex}: a test can have mutual exclusive tests
  Test $i \in I$ may require a set of mutual exclusive tests $\{M_i\}_{i \in I},\ M_i \subset I$, which may not run parallel to i. For example, you cannot simultaneously test whether the air conditioning can precisely regulate to a predefined high or low temperature.
- Requirement R_{res}: a test has resource criteria
  Test $i \in I$ may temporarily consume a fixed amount $\{r_{i,j}\}_{i \in I, j \in J},\ r_{i,j} \in [0, 100]$ of automotive bus resource $j \in J$. At no point in the testing schedule may the consumed amount of resources be greater than 100% on any of the resources.
- Requirement R_{status}: a test has status criteria
  A test $i \in I$ may require or change a status of a reference object $l \in L$, i.e., $\{c_{i,l}\}_{i \in I, l \in L},\ c_{i,l} \in \{require\_on,\ require\_off,\ turn\_on,\ turn\_off,\ any\}$. For exam-

ple, if test $i \in I$ has the status $c_{i,l} = require\_on$ for $l = engine$, another test $j \in I$ with status $c_{j,l} = turn\_on$ must be run beforehand. No test $m \in I$ with status $c_{m,l} = turn\_off$ has to be performed in between. For example, before performing a worker-guided test, at least one worker must be present at the test location and thus be set to the state *turn_on*.

The scheduling of N tests is processed on $K = \{1, \ldots, S\}$, which is the set of $S \in \mathbb{N}$ identical and parallel machines. Each test $i \in I$ has a run time $\{t_i\}_{i \in I}$ and can only be processed by one processing unit of the FMS at a time. We specify that the machines are identical, so that processing $i \in I$ takes time $\{t_i\}_{i \in I}$ on any machine (Shmoys et al. 1991).

We define C as a fixed and finite set of requirements and will refer to s_i as the start time and $e_i := s_i + t_i$ as the end time for the test $i \in I$. Therefore, the time interval of test $i \in I$ is denoted as $\{T_i := [s_i, e_i]\}_{i \in I}$ with $T_i \in \mathbb{R}^2$. A feasible solution of the start time $\{s_i\}_{i \in I}$ with respect to constraints in C is called $\Theta_C \in \mathbb{R}^{|I|}_{\geq 0}$. We minimise the schedule with respect to the makespan

$$e_{\max} = \max_{i \in I} \quad e_i. \tag{1}$$

An optimal solution for a set of constraints given an objective is called $\hat{\Theta}_C$.

Every test has a predicted run time, as stated in requirement R_{time} and can inherit any combination of further constraints. Fulfilling those requirements will ensure a faultless testing procedure and exclude the risk of damage to any of the components.

We classify the scheduling problem in terms of the three-field notation of Brucker (2007). A scheduling problem is classified in $\alpha|\beta|\gamma$. α determines the machine environment. In this work, we assume $\alpha = \{\alpha_1 = K\}$ because of identical parallel machines. Job characteristics are determined by $\beta = \{\beta_2 = prec, \beta_3 = const\}$. In this work, directed acyclic graphs are of interest so that $\beta_2 = prec$. The specification of release dates is also of interest, in combination with the car's state of construction on the assembly line, i.e., $\beta_3 = const$. The optimality criterion is the minimal completion time of the test schedule, i.e., $\gamma = e_{\max}$. As proposed by Brucker (2007), we ignore the empty symbols in the three-field notation.

2.3 Research questions

We assume that the fulfilment of the six outlined requirements R_{time}, R_{dir}, R_{pre}, R_{mutex}, R_{res} and R_{status} forms the theoretical basis for a procedure to schedule diagnostic tests flexibly in automotive manufacturing. To investigate the complex scheduling problem, we formulate the research questions $RQ1$ and $RQ2$ as follows:

$RQ1$: Which mathematical model describes the scheduling problem, taking into account the requirements R_{time}, R_{dir}, R_{pre}, R_{mutex}, R_{res} and R_{status}, and how can a numerical solution be found?

$RQ2$: What are the key elements of the procedure for scheduling diagnostic tests during the assembly line production in automotive companies based on the model provided by $RQ1$?

3 Related work

Based on the requirements of Section 2.2, we identified the criteria relevant for the classification of related work (see Table 2).

Scheduling in automotive use cases has been extensively discussed in literature. Shi et al. (2017), for example, developed test plans with tight schedules that combine multiple diagnostic tests on cars to fully utilise all of the available time in prototype car test scheduling. Moreover, Spieckermann et al. (2004) dealt with the scheduling problem of operating paint shop systems with sequence-dependent set-up costs.

Schedules of diagnostic tests in automotive production lines contain safety and certification-relevant test steps. This is why we prefer exact deterministic mathematical methods in this paper. The constraints implicated by the direct predecessor R_{dir} and the set of predecessors R_{pre} are explained exhaustively in any context of linear programming (Vanderbei 2020). The mutual exclusion constraint R_{mut} can be formulated using binary helper variables.

We will develop a method to translate the requirements of Section 2.2 into a piecewise linear model (mixed-integer model) for every requirement except R_{res}, which will be formulated as a logical constraint.

To do so, the mathematical model of this paper will be formulated as a *Boolean satisfiability* (SAT) problem and will be solved accordingly, e.g., with the Generic seaRch Algorithm for the Satisfiability Problem (GRASP) of Marques-Silva and Sakallah (1999). An overview on the theory and applications of SAT problems is provided by Pulina and Seidl (2020). Huang and Zhou (2018), for example, provide an efficient SAT encoding method for complex job-shop scheduling.

The group of SAT problems is proven to be NP-complete by Cook (1971). This means that it is possible to verify any given solution in polynomial time, but no algorithm exists that can find one such solution in polynomial time. As the algorithm engineering of modern SAT solvers has improved over recent years, the real world running time in the present use case of static scheduling is of negligible importance (Alouneh et al. 2019).

Weckenborg et al. (2020) discussed the automotive capacity scheduling of prototype vehicle production at the Volkswagen Pre-Production Center. Resource allocation, the selection and scheduling of orders for prototype vehicle production are of interest here. Moreover, Weckenborg et al. (2020) proposed a spreadsheet-based decision support system for daily capacity scheduling. Nevertheless, the setting is different to our work in terms of the application and the criteria of Table 2.

Buergin et al. (2018) introduced an optimisation model for the local order scheduling of mixed-model aircraft assembly lines covering both assignment to lines as well as sequencing with respect to cost. Requirements R_{time}, R_{pre} and R_{res} are the focus of the work of Buergin et al. (2018).

Dörmer et al. (2015) covered the problem of master production scheduling for high-variant, mixed-model automotive assembly lines. Individual, customer-defined models of a basic product type are assigned to short-term production

Table 2 Classification of related work - symbol "X" means applicable, symbol "-" means not applicable or not specified, P.I.M. = Parallel identical machines for batch scheduling

Reference	Application	Solving method	P.I.M.	R_{time}	R_{dir}	R_{pre}	R_{mutex}	R_{res}	R_{status}
This paper	Car diagnostics testing	GRASP	X	X	X	X	X	X	X
Weckenborg et al. (2020)	Automotive capacity scheduling	Binary integer programming	-	X	-	-	-	X	X
Buergin et al. (2018)	Local order assignment in aircraft manufacturing	Mixed-model sequencing	-	X	-	X	-	X	-
Dörmer et al. (2015)	Automotive master production	Heuristics	-	X	-	-	-	X	-
Wang and Liu (2015)	Production scheduling and preventive maintenance	Genetic	X	X	-	X	-	X	-
Bartels and Zimmermann (2009)	Automotive prototypes	Priority heuristic	-	X	-	X	-	X	-

periods while anticipating the negative impacts of an unbalanced model sequence at the lower planning level. The focus is on workload and processing times at the stations on the assembly lines with respect to cost, thus the requirements R_{time} and R_{res} are fulfilled.

Wang and Liu (2015) described a multi-objective, parallel machine scheduling problem with resources on machines and moulds, and with flexible preventive maintenance activities on resources. The work deals with parallel identical machines and addresses the requirements R_{time}, R_{pre} and R_{res}.

Bartels and Zimmermann (2009) covered an approach to experimental cars based on multi-mode, resource-constrained project scheduling with minimum and maximum time lags as well as renewable and cumulative resources, i.e., requirements R_{time}, R_{pre} as well as R_{res} are studied. However, the objective is different as scheduling is used to minimise the number of automotive prototypes used in Bartels and Zimmermann (2009).

To the best of the authors' knowledge, no literature exists on combining the constraint criteria R_{time}, R_{dir}, R_{pre}, R_{mutex}, R_{res} and R_{status} into a deterministic comprehensible method which optimises the makespan of the test schedule. There is no suitable approach to fulfill all of the criteria in Table 2 for automotive batch scheduling on assembly lines.

4 Model formulation and solution for flexible scheduling

We will build the full model, which describes the input and output data with regard to the conditions and requirements of Section 2. Without loss of generality, the constraints will be formulated for a test $i \in I$.

4.1 Constraints for tests as model input

Referring to the problem description in Section 2.2, the input data can be represented in a table of length $|I|$. Every test $i \in I$ has constraints and requirements that are indicated in this table. There is no specific requirement for the input format or data type as it depends on the realised implementation.

4.2 Objectives for the mathematical model

The objective of the optimisation in this work is to calculate the start time s_i, given the input table, so that the makespan is minimised as defined in Eq. 1, formally

$$\min_{\{s_i\}_{i\in I}} e_{\max} \;=\; \min_{\{s_i\}_{i\in I}} \max_{i\in I} e_i. \tag{2}$$

This ensures that the testing schedules are as cost efficient as possible, whereby cost is related to minimal run time.

To force the constraint R_{dir} of the direct predecessor pre_i of test $i \in I$, we require

$$e_{pre_i} = s_i. \tag{3}$$

Next, we formulate the requirement R_{pre} for the set of predecessors P_i of test $i \in I$,

$$e_p \leq s_i \quad \forall p \in P_i. \tag{4}$$

So far, this is a strictly linear model and a solution Θ_C for such could be found using the Simplex method in linear programming (Vanderbei 2020), given a linear objective function. Note that Eq. 2 is not linear.

The helper variables have to be defined for the mutual exclusion requirements R_{mutex} of test $i \in I$. We define

$$\tilde{M} := 1 + \sum_{i \in I} t_i \tag{5}$$

as a helper variable in order to process the constraints '*either or*': To enforce the mutual exclusion requirement for test $i \in I$, every test in M_i must '*either end before i*' or '*start after i*'. With the help of a binary variable $\tilde{h}_{i,m} \in \{0, 1\}$, the '*either or*' constraint is translated to

$$e_m \leq s_i + \tilde{M} * \tilde{h}_{i,m}, \quad \text{and} \tag{6a}$$

$$e_i \leq s_m + \tilde{M} * (1 - \tilde{h}_{i,m}), \quad \forall m \in M_i. \tag{6b}$$

We forego a formal proof, but will explain shortly: According to Definition 5, it holds that

$$e_{max} \leq \sum_{i \in I} t_i < \tilde{M}. \tag{7}$$

$\sum_{i \in I} t_i$ is the upper limit of the makespan, but note that the trivial chaining of tests is not necessarily a feasible solution. However, every feasible solution Θ_C is shorter or equal in time.

We start with $\tilde{h}_{i,m} = 0$. Eq. 6b is then automatically fulfilled so that test m must end before test i starts.

Vice versa, assuming $\tilde{h}_{i,m} = 1$. As $e_m \leq e_{max}$ and $s_i \geq 0$, it holds that for fixed $i, m \in I$, Eq. 6a is

$$\begin{aligned} & e_m \leq s_i + \tilde{M} \\ \iff \quad & e_m - s_i \leq \tilde{M} \\ \overset{Def.\ 5}{\iff} \quad & e_m - s_i \leq \sum_{i \in I} t_i \ < \ \tilde{M}, \end{aligned}$$

and therefore, the constraint will always be fulfilled no matter what s_i, s_m is. In that case, Eq. 6b is enforced, so that test m must begin only after test i is finished.

We have successfully translated '*either* Eq. 6a *or* Eq. 6b' into the system using the binary helper variable $\tilde{h}_{i,m} \in \{0, 1\}$. The system of constraints containing the

mutual exclusion requirements is no longer a strictly linear model, but rather a mixed-integer linear (MILP) system. It must be solved with suitable methods, e.g., a combination of the Simplex method and Branch and Bound method of Integer Programming (Hu and Kahng 2016).

In the next step, we want to translate the status requirement R_{status} into mathematical constraints, which will keep the system a MILP problem. To do so, we have to interpret what possible status condition the test $i \in I$ can have. As defined in R_{status} for a test $i \in I$ and a status $l \in L$, the test $i \in I$ can either require the status condition to be '*off/on*', but can also turn the condition '*off/on*'. Note that it may also hold the '*any*' requirement, which defines the status condition l to be of no concern to test i. Note that by definition, the status condition, if changing, is updated when the test is completed and that during that test, no other test which is not of status '*any*' can be run. An intuitive explanation will follow for these conditions based on the Pseudo code illustrated in Algorithm 1 of Appendix A.

At the start of the testing procedure, all status conditions are '*off*' by definition. If there is a test $i_1 \in I$ with condition '*require_on*', there are two possible allocations of the test. Either i_1 is executed before any other test with the status condition '*turn_on*' or i_1 is executed after a test with the status condition '*turn_off*', but before any test with '*turn_on*' is executed in between. On the other hand, a test $i_2 \in I$ with status condition '*require_on*' can only run if a test with status condition '*turn_on*' is run beforehand and no test with status condition '*turn_off*' may run in between. Note that those constraints must hold for every kind of status condition $l \in L$.

There may be multiple '*turn_on*'–'*turn_off*' tests, which is why every possible combination must be taken into account by using '*either or*' constraints as a help again. As the '*either or*' in this case can also be '*at least one of many*', we use a binary helper variable for every possible case and the constraint, so that

$$\sum_{j \in |helper^l|} h_{j,l} := |helper^l| - 1. \tag{8}$$

Allow us to illustrate an example: let $i_1, i_2, i_3, j \in I$ and we want j to only start if at least one of the other tests i_1, i_2, i_3 is finished. The system would be

$$e_1 \leq s_j + \tilde{M} * h_1\,, \tag{9a}$$

$$e_2 \leq s_j + \tilde{M} * h_2\,, \tag{9b}$$

$$e_3 \leq s_j + \tilde{M} * h_3\,, \tag{9c}$$

$$\sum_{k \in \{1,2,3\}} h_k = |\{1,2,3\}| - 1 = 2\,. \tag{9d}$$

It can be seen that at least one of h_1, h_2, h_3 must be equal to 0, and therefore one of Eqs. 9a, 9b or 9c are enforced, while the other two equations are of no concern.

For the resource constraint R_{res}, it would be possible to keep the model strictly piecewise linear and solve it with an MILP method. This has the disadvantage of having an exponential number of constraints depending on the number of tests that use a resource. We therefore introduce a logical constraint $\tilde{r}_{t,i,j}$, which forces the model to be solved as a SAT. For every test $i \in I$ we create an interval $T_i = [s_i, e_i]$ and define

$$\tilde{r}_{t,i,j} := \begin{cases} r_{i,j} & \text{if } t \in T_i \\ 0 & \text{else,} \end{cases}$$

to enforce

$$\sum_{i=1}^{|I|} \tilde{r}_{t,i,j} \leq 100 \quad \forall j \in J \quad \forall t \in [0, e_{max}]. \tag{10}$$

Implementing this constraint depends on the SAT solver or library that is used. In this work, the solver used by the implemented library of Perron and Furnon (2019) is the conflict-driven clause learning (CDCL) algorithm, which was developed by Marques-Silva and Sakallah (1999).

We have formally established the constraints of the system and will now be able to use a SAT solver to find an optimal solution.

4.3 Schedule of diagnostic tests as model output

The optimised schedule includes the start times s_i of the tests $i \in I$. As the run time t_i for a test $i \in I$ is only a prediction based on historic data, real-world testing time may depend on many external non-controllable factors such as worker interaction. Thus, an output format is needed that still forces the test schedule to adhere to all requirements. Hence, an adjacency list is introduced which contains a list of predecessors for every test. In order for test i to be started, every test in the predecessor list of test i must have finished. An exemplary adjacency list is illustrated in Table 3. The derived schedule begins with tests 1 and 2, which have no predecessors. Thus, they can be run in parallel. When test 1 is finished, test 3 can start. When tests 2 and 3 are finished, test 4 can finally start.

We provide a Pseudo code in Algorithm 2 of Appendix A to translate the optimised start times to an adjacency list. The makespan of the adjacency schedule is

Table 3 Adjacency list as representation of achieved schedule

Test		Predecessors
1	←	None
2	←	None
3	←	1
4	←	2 3

equal to the makespan of the start time $e_{\max}$, if the predicted times $\{t_i\}_{i \in I}$ are used to calculate the makespan.

5 A novel automated scheduling procedure

Having derived the mathematical scheduler in Section 4, we now describe the overall procedure to schedule diagnostic tests for the testing and commissioning of electronic components in automotive manufacturing. The mathematical model is hereby the central element.

The novel procedure consists of seven steps and two decisions (compare Fig. 2). We run through the procedure with an exemplary set of N tests $I = \{1, \ldots, N\}$ where $I \subset I_{all}$ and I_{all} are the set of all existent tests over all cars. Moreover, we use automation systems as FMS, most of which are structured hierarchically in the so-called automation pyramid according to ISA-95 (ISA 2000).

The procedure for scheduling diagnostic tests on a selected car starts with *Step 1*. The set of tests I, that are performed on a specific car, depends on the configuration codes of the car. The configuration codes are linked to the tests, and this linked information is the input data for the first step. In *Step 2*, the location of the car on the assembly line is determined with the help of a production system at the supervisory level (ISA 2000). In *Step 3*, a production system at the planning level offers the constraints C for the configuration code-dependent tests of the car on the assembly line. The constraints C are necessary to schedule N tests for a car with respect to the business and engineering requirements R_{time}, R_{dir}, R_{pre}, R_{mutex}, R_{res} and R_{status}.

In *Step 4*, the tests are scheduled with the objective of minimising the overall test run time. The mathematical model behind *Step 4* has been highlighted extensively in Section 4. The schedule is represented as an adjacency list.

In *Step 5*, the resulting schedule of N tests for the car on the assembly line is transferred for execution with a controller at the supervisory level. A code repository can support with binaries for each test. In *Step 6*, the results of the executed schedule are analysed and documented. In *Decision 1*, a check is carried out as to whether there is at least one test with a result equal to 'not OK'. If so, *Decision 2* checks whether a repetition of the test or tests with the result equal to 'not OK' is feasible. The repetition can depend on the new position of the car in the assembly line production at the time of *Decision 2*. If the repetition is feasible, the tests with the result equal to 'not OK' are repeated for the car on the given assembly line by going back to *Step 2*. If all test results are 'OK', *Decision 2* is skipped. In *Step 7*, the N test processes have been carried out on the car. The procedure is finished.

6 Industrial evaluation

We would now like to describe the industrial evaluation of the mathematical model of Section 4 and the procedure from Fig. 2 for an automotive case study with an OEM.

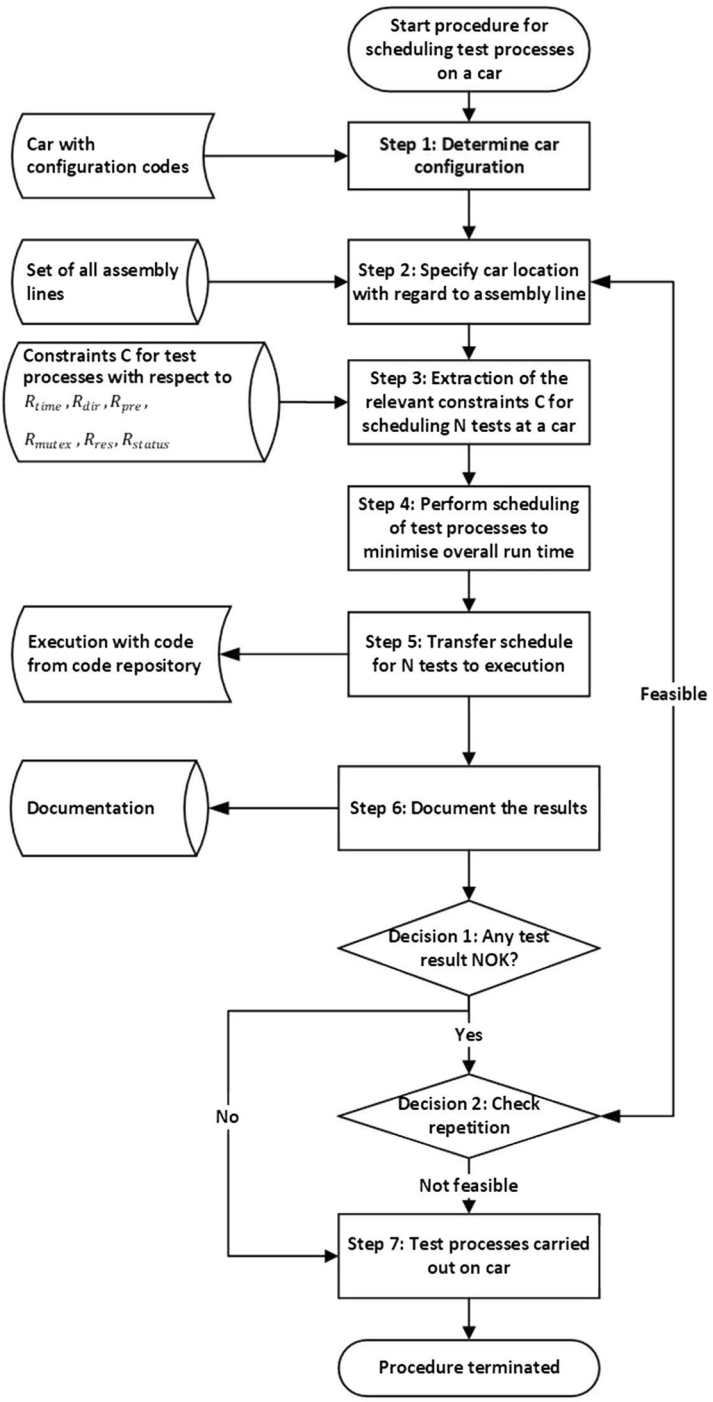


Fig. 2 Flowchart of scheduling procedure to run diagnostic tests more flexibly and reduce overall run time

6.1 Details of computational implementation environment

A prototype of the mathematical model in Section 4 was implemented to validate this work. We used a Lenovo ThinkPad T570 with 24GB RAM and an Intel i5-6300U processor to illustrate the run time efficiency of the method. In the computation environment, we used Google OR - Tools (Perron and Furnon 2019) for implementation in Python 3.8 as the number of constraints may rise to 10^3. As suggested in the benchmark test of Da Col and Teppan (2019), there is no real-world run time problem, even if one would expect the number of constraints to rise gradually with the development of electrified cars and more complex assembly lines.

6.2 Case study-based industrial evaluation

Fig. 2 illustrates three types of input parameters: the car with its configuration codes, the set of all test locations in the factory, and the boundary constraints for the tests that are performed on the car at a test location. In the following real-world case study, we therefore look at the procedure from two angles: first, we select a specific test location at the automotive assembly line. The data related to the tests at the chosen test location, are illustrated in Table 4. This case study allows us to evaluate the mathematical model presented in Sect. 4 in detail.

Second, we look at the associated relationships with the tests and the number of all cars produced in the selected production hall at the German location over a fixed period. We hereby evaluate the procedure illustrated in Fig. 2.

We begin by looking at the test location in the interior installation of the assembly line production that contains representative tests of automatic and worker-guided test scopes. Table 4 gives an overview of the tests for the case study.

We find $I = \{1, \ldots, N = 21\}$ tests. Each of which is described by an indication of time (expressed through the column *time* [*s*]), necessary repetition (column *run*), predecessor-successor relationships (columns *precond* and *previous*), exclusion of related tests (column *mutex*), resources of automotive gateway systems (columns *gate1load_resource* in unit [%] and *gate2load_resource* [%]). The case study contains worker-guided tests 3, 4, 5, 6, 16, and 17 (column *worker_status*).

In this setting, there is only one worker available at the test location, so that only one worker-guided test can be run in parallel. We consider no tests requiring communication with test benches. Automated tests can contain functions such as coding and flashing. Tests 18, 19, 20, and 21 are help tests in the chosen test location: test 18 turns the ignition of the car on and test 19 turns it off. Test 20 requires the existence of a worker and test 21 releases it.

All tests in Table 4 except for tests 18, 19, 20, and 21 are carried out with the ignition on (column *ign_condition_status*). Which test can be tested on a car depends on the car configuration. There are 128 possible schedules in this case, which will be reduced to feasible combinations when matching them with the actual equipment codes of a fixed car volume. On the one hand, the test of the exit lights is performed in tests 9 and 10, and can be carried out automatically depending on a

Table 4 Implementation details for industrial case study on an automotive assembly line

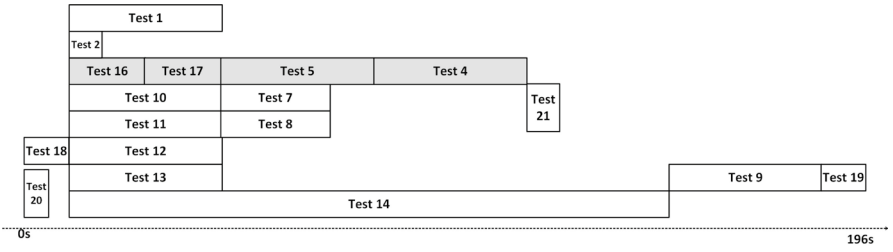
Tests I	time [s]	precond	previous	mutex	gate1load_ resource [%]	gate2load_ resource [%]	worker_status	ign_condition_status
1	10	none	none	none	1	0	any	req_on
2	1	none	none	none	1	0	any	req_on
3	10	4, 12	none	3, 4, 5, 6, 16, 17	1	0	req_on	req_on
4	10	5, 13	none	3, 4, 5, 6, 16, 17	1	0	req_on	req_on
5	10	6, 11	none	3, 4, 5, 6, 16, 17	1	0	req_on	req_on
6	10	10	none	3, 4, 5, 6, 16, 17	1	0	req_on	req_on
7	7	12	none	none	1	0	any	req_on
8	7	10	none	none	1	0	any	req_on
9	10	none	none	none	100	100	any	req_on
10	0	none	none	none	2	0	any	req_on
11	10	none	none	none	2	0	any	req_on
12	10	none	none	none	2	0	any	req_on
13	10	none	none	none	2	0	any	req_on
14	180	none	none	15	20	20	any	req_on
15	10	none	none	14	20	20	any	req_on
16	5	none	none	3, 4, 5, 6, 16, 17	1	0	req_on	req_on
17	5	none	none	3, 4, 5, 6, 16, 17	1	0	req_on	req_on
18	3	none	none	none	0	0	any	turn_on
19	3	1 – 17	none	none	0	0	any	turn_off
20	0	none	none	none	0	0	turn_on	any
21	0	3, 4, 5, 6, 16, 17	none	none	0	0	turn_off	any

specific configuration code. On the other hand, the exit lights are executed manually in tests 3 and 6. In the following, we consider a set of the volume of E Class models produced at a German plant over the period February-May 2021.

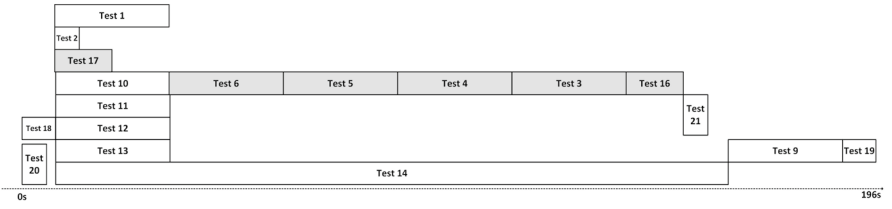
6.3 Results of the case study at the OEM based on the implementation of the scheduling procedure

We highlight *Steps 1-3* of the procedure (see Fig. 3). Considering the fixed car volume in the German production plant, two out of 128 feasible condition sets from Table 4 remain. This means that we have two scheduling variants. We determine, that test 15 is not relevant for the cars on hand in the period since the underlying configuration code is not installed. On the one hand, the test of the exit lights in tests 9 and 10 can be carried out automatically depending on a specific configuration code. On the other hand, the exit lights are executed manually in tests 3 and 6 if the specific configuration code is not present for a car. In 3% of the fixed car volume, the test set $I_1 = \{1, 2, 4, 5, 7, 8, 9, 10, 11, 12, 13, 14, 16, 17, 18, 19, 20, 21\}$ is scheduled in *Step 4*, whereas the set $I_2 = \{1, 2, 3, 4, 5, 6, 7, 8, 11, 12, 13, 14, 16, 17, 18, 19, 20, 21\}$ is scheduled in 97% of the cars.

Having applied the mathematical model presented in Section 4 on test sets I_1, I_2, the resulting schedules can be visualised in Fig. 3. The model terminated with the optimal status, meaning the shortest possible schedule, has been found in both test sets. Note that it is possible for more than one solution to be of optimal length. As already



(a) Version I_1 with two more automated tests 9,10



(b) Version I_2 with two more worker-assisted tests 3,6

Fig. 3 Two schedules as Gantt charts for the presented automotive case study – worker assistance is illustrated in grey, test 14 is not to scale

mentioned, the individual test times are only predictions. If the test times differ from the predictions in a scheduled procedure, a different schedule may have been quicker given the real world times. Hence, it is important to be as precise as possible when predicting the test run times.

The optimisation problem with respect to I_1 is solved on average over 10 runs in $0.0092ms$ and I_2 in $0.0165ms$.

6.4 Discussion

The results of the previous Section 6 are discussed and the work is assessed against the requirements formulated in Section 2.2 on a per-concept basis.

We have created a mathematical model to address the run time of a test (R_{time}), direct predecessors (R_{dir}), the set of all predecessors (R_{pre}), mutual exclusive tests (R_{mutex}), resource constraints (R_{res}) and status conditions (R_{status}) per definition. The resulting schedules of Fig. 3 have been confirmed manually, since the example that is illustrated in Table 4, is rather simple. The research question $RQ1$ is answered by definition.

Without considering the resource constraint, a branch-and-cut would be most appropriate, as has been discussed extensively in Padberg and Rinaldi (1991). The scalability and complexity is also well understood and discussed (Basu et al. 2021). Adding the logical constraint of the resources makes the optimisation more complex. In terms of the scope in the automotive industry, the limits of the model are out of reach as thorough benchmark tests have been performed and analysed in Da Col and Teppan (2019). Hence, even if added complexity is assumed in the future by a more complex car architecture, the scalability will not pose a problem in this scope.

In terms of abstract complexity, scalability is dependent on the computational complexity of algorithms expressed in Big O-notation. General SAT problems are proven to be NP-complete by Cook (1971). Therefore, solving will have complexity $\mathcal{O}(2^N)$ in the worst case, but since no methods exist to reduce complexity and increase efficiency, benchmark tests as in Da Col and Teppan (2019) are of greater significance.

Furthermore, we proposed a procedure with key elements to schedule the tests on an assembly line for automotive companies based on the mathematical model provided in Fig. 2. We decided to apply static scheduling to each car, so that *Steps 1-6* of the procedure are straightforward. In the evaluation study, we were unable to test *Decisions 1 and 2* as the rescheduling policy since our test setting was not event-driven. However, the outlined procedure shows future possibilities to integrate the static scheduler into an event-driven production system. As the given scheduling procedure does not focus on a specific technology, it is transferable to other scenarios in different industries with scheduling processes. This brings us back to the research question $RQ2$ that we analysed as well.

7 Conclusion and future work

In this paper, we have described the development, implementation and successful evaluation of a mathematical optimisation model as basis for the flexible scheduling of automotive diagnostic tests on an assembly production line. The approach considers multiple scheduling constraints on the car engineering and assembly line related to time, direct predecessors, set of all predecessors, mutual exclusion criteria, resource and status conditions.

The model presented here is the main contribution of this work and is formulated as a Boolean satisfiability problem. It has been successfully validated on the final assembly line of a German OEM plant. We embedded the model into a procedure to schedule the tests for each car with configuration codes at the test location. By integrating the model into the procedure, we showed that complex scheduling with multi-constraints can be flexible and automated. In general, the schedules generated with the proposed method were better in terms of run time and work effort than the solutions generated by the current manual procedure. After successful test implementation, the OEM estimates the annual cost savings to be substantial.

This is the first use case that describes flexible scheduling on parallel identical machines with complex constraints for testing and commissioning in car manufacturing. Thereby, this work presents a way to integrate the procedure into an event-driven production system. The paper provides a basis for further research into optimisation methods for automated workflow management in automotive manufacturing. The dynamic scheduling of diagnostic tests will be of interest in future.

Appendix A

Data: Lists/dicts/objects
$\{s_i\}, L, I, \{c_{i,l}\}, \{require\_on^l\}_{l \in L}, \{require\_off^l\}_{l \in L}, \{turn\_on^l\}_{l \in L}, \{turn\_off^l\}_{l \in L}$, model
Result: Constraints forcing the system for status conditions
for *l in L* **do**
  for *i in* $require\_on^l$ **do**
    for *p in* $turn\_on^l$ **do**
      model.Add($s_p + t_p \leq s_i + M * hc_{l,i,p}$);
      for *k in* $turn\_off^l$ **do**
        model.Add($s_k + t_k \leq s_p + M * hc_{l,i,p} + M * hc_{l,i,p,k}$);
        model.Add($s_i + t_i \leq s_k + M * hc_{l,i,p} + M * (1 - hc_{l,i,p,k})$);
      end
    end
    model.Add($\sum_{p \in t\_On_j} hc_{l,i,p} = |turn\_on^l| - 1$);
  end
  ;
  for *i in* $require\_off^l$ **do**
    if $len(turn\_off^l) > 0$ **then**
      for *p in* $turn\_off^l$ **do**
        model.Add($s_p + t_p \leq s_i + M * (1 - hc_{l,i,p}) + M * hc_{l,i,p,off}$);
        for *k in* $turn\_on^l$ **do**
          model.Add($s_i + t_i \leq s_k + M * hc_{l,i,p} + M * hc_{l,i,p,off}$);
          model.Add($s_k + t_k \leq s_p + M * (1 - hc_{l,i,p}) + M * hc_{l,i,p,off} + M * hc_{l,i,p,k}$);
          model.Add($s_i + t_i \leq s_k + M * (1 - hc_{l,i,p}) + M * hc_{l,i,p,off} + M * (1 - hc_{l,i,p,k})$);
        end
      end
      model.Add($\sum_{p \in turn\_off^l} hc_{l,i,p} = |turn\_off^l| - 1$);
    else
      for *p in* $turn\_on^l$ **do**
        model.Add($s_i + t_i \leq s_p$);
      end
    end
  end
end

Algorithm 1: Pseudo code for handling of status conditions

Data: List $\{s_i\}$
Result: Adjacency list as dictionary <key>:<predecessors of key>
$adj\_list = \{\}$;
for i *in* I **do**
  $adj\_list[i] = []$;
  for j *in* I **do**
    if $e_j \leq s_i$ *and* $i \stackrel{!}{=} j$ **then**
      $adj\_list[i] \leftarrow j$;
    else
  end
end

Algorithm 2: Pseudo code translation to adjacency list

Author Contributions All authors contributed to the study conception and design. Material preparation, data collection and analysis were performed by SK, MR and FGA. The first draft of the manuscript was written by SK and MR and all authors commented on previous versions of the manuscript. All authors read and approved the final manuscript.

Funding Open Access funding enabled and organized by Projekt DEAL. No funding was received to assist with the preparation of this manuscript.

Data availability All of the data generated for the scheduling method during this study are included in this published article. Confidential data are not included.

Code availability The code that supports the findings of this study are available from Mercedes-Benz AG, but restrictions apply to the availability of the code. However, the code is available from the authors on reasonable request and with the permission of Mercedes-Benz AG.

Declarations

Conflict of interest The authors declare that they have no conflict of interests.

References

Alouneh S, Abed S, Al Shayeji MH, Mesleh R (2019) A comprehensive study and analysis on sat-solvers: advances, usages and achievements. Artifi Intelli Rev 52(4):2575–2601. https://doi.org/10.1007/s10462-018-9628-0

Bartels JH, Zimmermann J (2009) Scheduling tests in automotive r&d projects. Eur J Operation Res 193(3):805–819. https://doi.org/10.1016/j.ejor.2007.11.010
Basu A, Conforti M, Di Summa M, Jiang H (2021) Complexity of branch-and-bound and cutting planes in mixed-integer optimization - ii. In: Singh M, Williamson DP (eds) Integer Programming and Combinatorial Optimization. Springer, Cham, pp 383–398
Błaewicz J (2007) Handbook on scheduling: From theory to applications. International handbooks on information systems. Springer, Berlin
Brucker P (2007) Scheduling Algorithms, 5th edn. Springer-Verlag GmbH, Berlin Heidelberg, https://doi.org/10.1007/978-3-540-69516-5
Buergin J, Helming S, Andreas J, Blaettchen P, Schweizer Y, Bitte F, Haefner B, Lanza G (2018) Local order scheduling for mixed-model assembly lines in the aircraft manufacturing industry. Product Eng 12(6):759–767. https://doi.org/10.1007/s11740-018-0852-x
Cook SA (1971) The complexity of theorem-proving procedures. In: Harrison MA, Banerji RB, Ullman JD (eds) Proceedings of the third annual ACM symposium on Theory of computing - STOC '71, ACM Press, New York, New York, USA, pp 151–158, https://doi.org/10.1145/800157.805047
Da Col G, Teppan E (2019) Google vs IBM: A constraint solving challenge on the job-shop scheduling problem. Electron Proceed Theoret Comp Sci 306:259–265. https://doi.org/10.4204/eptcs.306.30
Dörmer J, Günther HO, Gujjula R (2015) Master production scheduling and sequencing at mixed-model assembly lines in the automotive industry. Flex Serv Manuf J 27(1):1–29. https://doi.org/10.1007/s10696-013-9173-8
Ebert C, Favaro J (2017) Automotive software. IEEE Software 34(3):33–39. https://doi.org/10.1109/MS.2017.82
ElMaraghy H, Caggiano A (2016) Flexible manufacturing system. In: Produ TIAf, Laperrière L, Reinhart G (eds) CIRP Encyclopedia of Production Engineering, Springer Berlin Heidelberg, Berlin, Heidelberg, pp 1–7, https://doi.org/10.1007/978-3-642-35950-7_6554-4
Hillier FS, Herrmann JW (2006) Handbook of Production Scheduling, vol 89. Springer, US, Boston, MA,. https://doi.org/10.1007/0-387-33117-4
Hu TC, Kahng AB (2016) Linear and Integer Programming Made Easy. Springer International Publishing, Cham,. https://doi.org/10.1007/978-3-319-24001-5
Huang H, Zhou S (2018) An efficient sat algorithm for complex job-shop scheduling. In: Proceedings of the 2018 8th International Conference on Manufacturing Science and Engineering (ICMSE 2018), Atlantis Press, Paris, France, https://doi.org/10.2991/icmse-18.2018.126
ISA (2000) Enterprise-control system integration (isa-95.00.01), models and terminology
Lawlor A (1973) Works Organisation. Macmillan Education UK, London,. https://doi.org/10.1007/978-1-349-01782-9
Marques-Silva J, Sakallah K (1999) Grasp: a search algorithm for propositional satisfiability. IEEE Trans Comput 48(5):506–521. https://doi.org/10.1109/12.769433
Padberg M, Rinaldi G (1991) A branch-and-cut algorithm for the resolution of large-scale symmetric traveling salesman problems. SIAM Rev 33:60–100
Perron L, Furnon V (2019) Or-tools By Google. Version 7:2
Pulina L, Seidl M (2020) Theory and Applications of Satisfiability Testing - SAT 2020, vol 12178. Springer, Cham,. https://doi.org/10.1007/978-3-030-51825-7
Quilliot A, Sarbinowski A, Toussaint H (2021) Vehicle driven approaches for non preemptive vehicle relocation with integrated quality criterion in a vehicle sharing system. Ann Operat Res 298:1–24. https://doi.org/10.1007/s10479-019-03497-4
Shi Y, Reich D, Epelman M, Klampfl E, Cohn A (2017) An analytical approach to prototype vehicle test scheduling. Omega 67:168–176. https://doi.org/10.1016/j.omega.2016.05.003
Shmoys D, Wein J, Williamson D (1991) Scheduling parallel machines on-line. SIAM J comput 24:131–140. https://doi.org/10.1109/SFCS.1991.185361
Spieckermann S, Gutenschwager K, Voß S (2004) A sequential ordering problem in automotive paint shops. Int J Prod Res 42(9):1865–1878. https://doi.org/10.1080/00207540310001646821
Vanderbei RJ (2020) Linear Programming. International Series in Operations Research & Management Science, Springer, Cham,. https://doi.org/10.1007/978-3-030-39415-8
Wang S, Liu M (2015) Multi-objective optimization of parallel machine scheduling integrated with multi-resources preventive maintenance planning. J Manuf Sys 37:182–192. https://doi.org/10.1016/j.jmsy.2015.07.002

Weckenborg C, Kieckhäfer K, Spengler TS, Bernstein P (2020) The volkswagen pre-production center applies operations research to optimize capacity scheduling. INFORMS J Appl Anal 50(2):119–136. https://doi.org/10.1287/inte.2020.1029

Simone König is currently Development Engineer at the research and predevelopment of production technologies at Mercedes-Benz AG with a main focus on data science and business process management. She studied Mathematics and is enrolled in the PhD programme of the TUM School of Engineering and Design at Technical University of Munich.

Maximilian Reihn is currently enrolled in the Master's programme of Mathematical Finance at University Konstanz. During his studies he started as working student to support a R&D team at Mercedes-Benz AG and developed algorithmic code for data science projects. His academic interest focuses on numerical optimisation and optimal control problems.

Felipe Gelinski Abujamra is currently Development Engineer at the research and predevelopment of production technologies at Mercedes-Benz AG, focusing on development concepts for production using process analysis. Felipe studied Business Information Systems at the Stuttgart Technology University of Applied Sciences. At Mercedes-Benz AG, he wrote his graduation thesis in 2019, in which he dealt with automated scheduling in the automotive manufacturing.

Alexander Novy is currently Development Engineer at the Charging Functions R&D Department at Mercedes-Benz AG. His focus is the development of digital services for electric vehicles. Before developing cloud services, Alexander worked on the research and predevelopment of production technologies at Mercedes-Benz with a main focus on business intelligence and data science. Methodologically, he focuses on machine learning methods, artificial intelligence and data modelling.

Birgit Vogel-Heuser is a full professor and director of the Institute of Automation and Information Systems at the Technical University of Munich. Her main research interests are systems engineering, software engineering, and modeling of distributed and reliable embedded systems. She is core member of TUM's MDSI (Munich Data Science Institute), member of TUM's MIRMI (Munich Institute of Robotics and Machine Intelligence), member of the German Academy of Science and Engineering, chair of the VDI/VDE working group on industrial agents and vice chair of the IFAC TC 3.1 computers in control.

Paper4

Online Test Scheduling in Car Production Lines

Simone König, Birgit Vogel-Heuser *IEEE Fellow*,
Florian Karg, Kathrin Land
Institute of Automation and Information Systems
Technical University of Munich
Garching, Germany

Emanuele Carbone, Adam Hradecky,
Michael Hahn, Oliver Kopp

Mercedes-Benz AG
Sindelfingen, Germany

2023 IEEE Intelligent Vehicles Symposium (IV) | 979-8-3503-4691-6/23/$31.00 ©2023 IEEE | DOI: 10.1109/IV55152.2023.10186671

***Abstract*—Cars with software-intensive features such as autonomous driving bring new opportunities to Original Equipment Manufacturers (OEMs) and their manufacturing processes. The complexity of internal engineering processes is increasing as software-intensive functions are connected to electric and electronic car architectures, which need to be tested and enabled for functionality in car manufacturing. However, internal engineering processes can be simplified in car manufacturing by automating previously manual activities and flexibly managing variants. In this paper, we present a concept for managing the online scheduling of diagnostic tests during dynamic car manufacturing. We introduce the online scheduling of diagnostic tests on a high-performance car computer and compare the concept with state-of-the-art concepts for offline scheduling on remote production servers. To evaluate our concept, we conduct an OEM case study that shows that online scheduling not only reduces the total testing time in manufacturing while managing schedule variants more flexibly, but also increases the overall value added in car production by simplifying existing employee role models. We illustrate a scenario of how new car architectures can facilitate digital manufacturing processes.**

***Index Terms*—Software-defined car, diagnostics and testing, online scheduling, variability management, manufacturing.**

I. Introduction to Diagnostics in Service-Oriented Car Architectures

Digital customer functions such as autonomous driving are implemented as software functions in the car. Customer functions need to be updated and upgraded continuously on a flexible automotive run time platform [1]. Appropriate technical concepts are over-the-air applications such as data collection, live diagnostics, and software updates.

These applications require a specific electric and electronic (e/e) car architecture: Instead of conventional domain-oriented architectures with central gateways, software service-oriented architectures with microprocessors called high-performance computers (HPCs) have gained influence [2]. Complex software functions are implemented on the HPCs, while electronic control units (ECUs) continue to assume the role of sensors and actuators. Figure 1 shows a simple schematic setup of a service-driven architecture with an API Server, e.g., SOVD for diagnostics as proposed by ASAM e. V. [3], HPCs, and ECUs.

In car manufacturing, software service-oriented architectures are commissioned via their e/e components, and failures are diagnosed with the help of diagnostic tests. Coding and functionality checking of the ECU in the seat are exemplary procedures. Thus, changes in service-oriented architectures have an influence on car manufacturing as well.

Determining a schedule of diagnostic tests for a car is the result of an engineering process. An interdisciplinary team of engineers from development, assembly, and rework determine the parameters that influence the schedules of these procedures. Car manufacturing is a dynamic and event-driven environment that can influence predetermined schedules of diagnostics procedures. Rescheduling becomes interesting in car architectures with powerful HPCs to optimize schedule variants in-vehicle and online under the condition of minimal schedule execution time. Rescheduling is motivated, e.g., by failed tests, unexpected longer test run times due to delays in software components, unexpected missing or unconnected parts, or in the worst case, broken IT communication. Rework operations on predecessor cars in the assembly line can also result in production delays.

Upgradable software functions, such as for autonomous driving, thus make an online scheduling concept interesting to flexibly adapt diagnostic test schedules in car production lines. The resulting research questions RQ1 and RQ2 in this paper are:

RQ1: What are the key requirements for online in-car scheduling of test procedures in car manufacturing?
RQ2: What are the benefits of the concept for the OEM?

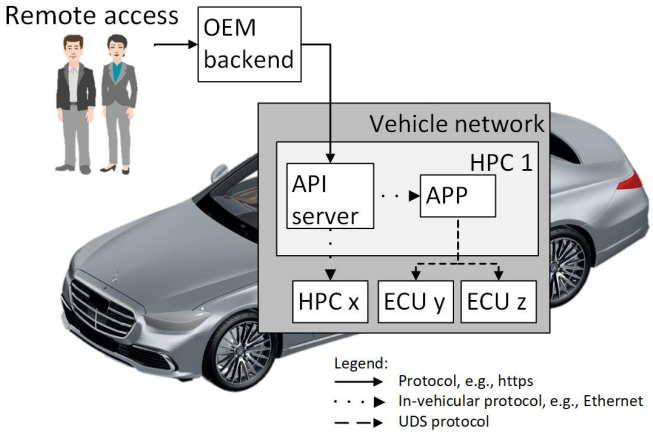


Fig. 1: Simplified scheme of e/e architecture in service-oriented cars

TABLE I: Relevant mathematical notation in this work

$I = \{1, \ldots, N\}$	Set of $N \in \mathbb{N}$ tests
$K = \{1, \ldots, S\}$	Set of $S \in \mathbb{N}$ machines
$J = \{1, \ldots, K\}$	Set of $K \in \mathbb{N}$ resources
$L = \{1, \ldots, Z\}$	Set of $Z \in \mathbb{N}$ status conditions of objects
$\{t_i\}_{i \in I}$	$t_i \in \mathbb{R}_{>0}$ units of time that test $i \in I$ takes
$\{pre_i\}_{i \in I}$	$pre_i \in I$ direct predecessor of test $i \in I$
$\{P_i\}_{i \in I}$	$P_i \subset I$ set of predecessors that must end before test $i \in I$ can start
$\{M_i\}_{i \in I}$	$M_i \subset I$ set of tests that may not run parallel to test $i \in I$
$\{r_{i,j}\}_{i \in I,\, j \in J}$	$r_{i,j} \in [0, 100]$ resources temporarily taken by test $i \in I$ on resource $j \in J$
$\{c_{i,l}\}_{i \in I,\, l \in L}$	$c_{i,l} \in \{turn_{\text{on}}, turn_{\text{off}}, req_{\text{on}}, req_{\text{off}}\}$ status of test $i \in I$ on condition $l \in L$
$\{s_i\}_{i \in I}$	$s_i \in \mathbb{R}$ start time of test $i \in I$
$\{e_i := s_i + t_i\}_{i \in I}$	$s_i \in \mathbb{R}$ end time of test $i \in I$
$\tilde{i}$	Test which has been executed and forces rescheduling
$e_{\max} = \max_{i \in I} \; e_i$	Makespan, the total length of a schedule
C	Set of constraints
$\Theta_C \in \mathbb{R}_{\geq 0}^{\lvert I \rvert}$	Feasible solution for a given set of constraints C

The paper's main contribution is a concept, applicable to cars with HPCs to schedule in-car tests online. Managing schedules online means managing diagnostic tests flexibly, based on dynamic manufacturing environments with varying test run times and results. We derive requirements for scheduling, the vehicle under test, and dynamic manufacturing environments that are essential for the concept. Based on the requirements, we present a concept that schedules tests online on an HPC in car manufacturing environments by optimizing a mixed-integer linear program. The concept is evaluated by an industrial OEM case study.

The remainder of the paper is structured as follows: Section II states the requirements for online car diagnostics test scheduling. Section III lists the related work. The concept for online scheduling of vehicular tests is presented in Section IV and evaluated using an OEM case study in Section V. The paper closes with a summary and outlook.

II. Requirements for online scheduling in dynamic car manufacturing

In the following section, we derive requirements that are essential for online scheduling of diagnostic tests in car manufacturing. The requirements are derived from OEM expert interviews. Table I summarizes the relevant notation concerning the requirements.

Requirement R_{sched}: The set of all tests is denoted by I, with $|I| \in \mathbb{N}$. A diagnostic test $i \in I$, which is executed during manufacturing, specifies a procedure that is conducted on the car to test or check functionalities in the e/e architecture. The test i is related to a planned test run time t_i specified by engineers. Within a set of tests I, a test i can have direct predecessors $\{pre_i\}_{i \in I}$ with $pre_i \in I$. $\{P_i\}_{i \in I}, P_i \in \mathbb{N}$ defines the set of tests that must run before test i. $\{M_i\}_{i \in I}, M_i \in \mathbb{N}$ defines the set of tests that must not run parallel to test i: Two tests i and j can be mutually exclusive, e.g., the heating and cooling function of the seating ECU. A diagnostic test i performed on the car network stresses car gateways when communicating with e/e components. Thus, a test i can limit gateway resources that are under load during communication. Therefore, we define $\{r_{i,j}\}_{i \in I \setminus \{i : r_{i,j} = 0\}, j \in J}$ with $r_{i,j} \in \mathbb{R}$ as the set of resources taken by test i on resource j, where $J = \{1, \ldots, K\}, K \in \mathbb{N}$ defines the set of K different resources. Finally, a test i can require conditions with respect to human guidance by workers, equipment, or states. Thus, let us define $\{c_{i,l}\}_{i \in I \setminus \{i : c_{i,j} = any\}, l \in L}$ with $c_{i,l} \in \{turn_{\text{on}}, turn_{\text{off}}, require_{\text{on}}, require_{\text{off}}\}$ as the set of condition states taken by test i on condition l, where $L = \{1, \ldots, Z\}, Z \in \mathbb{N}$ defines the set of Z different conditions. When performed as a test on a spot in the assembly line, a test i might have a timeout as well: the timeout can depend on the required test spot length, which is related, e.g., to the required test equipment and the manufacturing cycle time. We summarize the listed requirements in R_{sched} to realize time-optimized scheduling of diagnostic tests.

Requirement R_{veh}: The operation of the scheduling program in the car network depends on the car's HPC architecture. All relevant scheduling software should be adapted to the HPC architecture, which depends on the OEM guidelines, e.g., the AUTOSAR Adaptive Platform. Execution of diagnostic tests might be time critical and require low latency. Thus, we deal with soft and hard real-time assumptions. The computational scheduling process should take $<< 1s$ so that rescheduling does not take longer than the planned execution time of a test of, e.g., $1s$. CPU and RAM load on the HPC, stressed by the scheduling program, should be low.

Requirement R_{dyn}: Diagnostic test schedules should be created initially and only adjusted by rescheduling if there is a defined disturbance. The production hall influences the time-

TABLE II: Classification of related works (symbol 'x' means applicable, '(x)' partly applicable, '–' non-applicable or not specified)

Reference	R_{sched}	R_{veh}	R_{dyn}	**Approach**
Bohnenberger et al. [4]	(x)	-	x	Agents in online scheduling and production plant configuration
Gaid et al. [5]	(x)	(x)	(x)	Automotive network control systems
Sohn et al. [6]	-	(x)	-	Dynamic scheduling of car messages
Zhang et al. [7]	(x)	x	-	Automotive apps scenario of plug and play
Xie et al. [8]	(x)	x	-	Adaptive dynamic scheduling on automotive embedded systems
Dvořák and Hanzálek [9]	(x)	-	(x)	Hybrid scheduling on vehicular networks
Faizan and Pillai [10]	-	(x)	-	Dynamic task allocation on ECUs
Zhao et al. [11]	(x)	-	x	General dynamic flexible job-shops
Land et al. [12]	(x)	-	-	Test case scheduling in manufacturing
Pradeep Kumar and Pillai [13]	-	x	-	Dynamic scheduling for AUTOSAR apps
Wang et al. [14]	(x)	(x)	-	Scheduling on vehicular edge computing
Hu et al. [15]	(x)	x	-	Hybrid scheduling on vehicular network
This paper	x	x	x	Online scheduling on vehicular systems in manufacturing

optimized scheduling program since failed diagnostic tests can be retested, e.g., as long as there is time in the testing spot with respect to the cycle time. There might be ECU or IT network infrastructure breakdowns, that we treat as stochastic events in manufacturing. Diagnostic tasks might also include human interaction in sub-tasks. Variability in the flexible execution of the tests has to be controlled by checking the schedule status and adjusting it if necessary as soon as disturbances are present. Tests that are already in progress should not be disturbed, so the schedule program is non-preemptive. Disturbances, that require rescheduling, are varying test run times $\tilde{t}_i \neq t_i$ or failed tests with the status *not OK*.

The developed requirements R_{sched}, R_{veh}, and R_{dyn} are the answer to the research question RQ1. Based on the requirements, we classify related works in the next section.

III. Related work

We give an overview of scheduling and rescheduling to classify our problem theoretically. Then, we have a look at scheduling problems in manufacturing and vehicular networks. Table II classifies related work with respect to the derived requirements R_{sched}, R_{veh}, and R_{dyn} of Section II.

a) Background in scheduling: Optimization problems for scheduling have extensive literature, which is why we limit ourselves to online scheduling in manufacturing and the automotive domain. Basic literature on scheduling and online scheduling is discussed, e.g., by Błażewicz et al. [16].

Scheduling problems in manufacturing are either in static or dynamic environments [17]. In scheduling problems with static environments, the order of the different tasks can be specified at design time.

Dynamic scheduling involves robust proactive, completely reactive, and predictive-reactive scheduling [16]. The concept of robust proactive scheduling focuses on the creation of proactive schedules that do not have to be further adjusted during production. The concept of completely reactive scheduling describes scheduling in real time without primarily scheduling the processes, and therefore, it is the opposite strategy of robust proactive scheduling. The concept of predictive-reactive scheduling consists of two main steps. In the first step of predictive scheduling, an optimal schedule is created. If disturbances occur and the initial schedule becomes infeasible, reactive scheduling is processed in the second step to achieve an optimized schedule under the new environmental conditions. Reactive scheduling is also called rescheduling. If disturbances occur in the stochastic manufacturing environment or car, the initially created optimal schedule can become infeasible and needs to be adjusted.

There are three policies to decide when the rescheduling takes place: periodic, event-driven, and hybrid.

Moreover, a rescheduling strategy has to be chosen. Major strategies are regeneration, partial rescheduling, and right-shift scheduling [18]. In regeneration, a totally new schedule is computed for the remaining process steps. This leads to good performance but can require high computational effort and cause schedule nervousness. Partial rescheduling only considers the processes directly or indirectly affected by real-time disturbances. Since all other processes stay in their original order, schedule nervousness is reduced. Right-shift rescheduling postpones the remaining process steps by the duration of the machine breakdown or the duration of a rush order. It causes the least schedule nervousness, but can lead to longer overall processing times.

b) Online scheduling in manufacturing systems and the automotive domain: We identify related works based on the derived requirements R_{sched}, R_{veh}, and R_{dyn}. In Table II, the related works are classified.

Following the requirements of Section II, we integrate the problem into a manufacturing hall with disturbances. Taking the car configuration set and production planning into account, the schedules can be planned and executed flexibly on the automotive HPC. The environment is stochastic. Requirement R_{dyn} requires predictive-reactive scheduling. As described by requirements R_{veh} and R_{dyn}, the disturbances are event-driven, so we choose the event-driven policy. According to R_{veh}, the scheduling program should take $<< 1s$, such that we go on with the strategy of regeneration if we are able to apply a suitable solver realization. To sum up, we consider online scheduling with rescheduling and no repair strategy.

Bohnenberger et al. [4] demonstrate a general agent-based

approach to online scheduling and production plant configuration in the manufacturing environment.

Zhao et al. [11] describe a rescheduling method for flexible job shops with machine failures under the application of model-free reinforcement learning. The dynamic, flexible job-shop problem is more general, using dispatching rules as a rescheduling strategy, and is similar to the requirements of R_{sched}. The paper focuses on the scheduling method itself, so there are no assumptions made regarding R_{veh}. The dynamic environment is considered under machine failures with respect to R_{veh}.

Gaid et al. [5] present a general mixed logical dynamical framework to schedule a car suspension control system under network communication constraints.

Land et al. [12] apply dynamic programming to test case scheduling in automotive production systems. The goal is to maximize the overall utility of the test schedule in manufacturing within the limited test execution time. R_{sched} is partly fulfilled.

Sohn et al. [6] discuss a dynamic programming model for scheduling in-vehicle messages in information systems. The model prioritizes the presentation of messages over time to the driver.

Zhang et al. [7] discuss a schedule management framework in the plug-and-play scenario for cloud-based automotive software systems. The work addresses computation and storage capacity onboard the vehicle and at a server through cloud computing.

Xie et al. [8] introduce a fairness-based dynamic scheduling algorithm for automotive functions from a high-performance perspective on the AUTOSAR Adaptive Platform.

Dvořák and Hanzálek [9] present a heuristic scheduling algorithm to manage internal vehicle communication applied to the network communication protocol FlexRay for multiple vehicle variants.

Faizan and Pillai [10] describe a dynamic task allocation and scheduling approach for multi-core ECUs. The work promises better core assignment for safety-critical and real-time tasks.

Pradeep Kumar and Pillai [13] present a performance comparison of static and dynamic priority scheduling algorithms for automotive subsystems on AUTOSAR. The work addresses non-safety-critical applications. Therefore, R_{veh} is considered. Nevertheless, there is no transfer w.r.t. the requirements R_{sched} and R_{dyn}, as the presented scheduling concept is preemptive, which contradicts Section II.

Wang et al. [14] present an online offloading scheduling and resource allocation algorithm for vehicular edge computing systems. The authors present a vehicular edge computing architecture and use a game-theoretic online algorithm as the solver.

A hybrid scheduling framework for mixed real-time tasks on automotive systems is presented by Hu et al. [15]. A global scheduler manages hard real-time tasks and a local scheduler assigns soft real-time tasks.

To the best of our knowledge, no concept exists that addresses online scheduling in terms of the requirements R_{sched}, R_{veh}, and R_{dyn}.

IV. Concept for online test scheduling in car manufacturing

The preceding sections let us now present the concept for online test scheduling based on the requirements R_{sched}, R_{veh}, and R_{dyn}. The concept includes the underlying mathematical model and software architecture of how the online scheduling of diagnostic tests can be designed in variant-rich car manufacturing.

A. Mathematical model including rescheduling

Formalizing requirement R_{sched} from Section II, we follow the work by König et al. [19]. Let $\{s_i\}_{i\in I}$ with $s_i \in \mathbb{R}$ be the start time and $e_i := s_i + t_i$ be the end time of the test $i \in I$. Therefore, the time interval of test $i \in I$ is denoted as $\{T_i := [s_i, e_i]\}_{i\in I}$ with $T_i \in \mathbb{R}^2$. We define C as a fixed and finite set of constraints to be fulfilled by tests, as stated in Section II. A feasible solution of the start time $\{s_i\}_{i\in I}$ with respect to constraints in C is called $\Theta_C \in \mathbb{R}^{|I|}_{\geq 0}$. We minimize the schedule with respect to the makespan

$$e_{makespan} \coloneqq \max_{i\in I} \quad e_i. \tag{1}$$

An optimal solution for a set of constraints given an objective is called $\hat{\Theta}_C$.

Introducing online scheduling brings us to reactive scheduling. Reactive scheduling means rescheduling the schedule $\hat{\Theta}_C$ when a test $\tilde{i}$ of the running tests $\tilde{i} \cap Q_{\tilde{i}}$ is already finished, and there is a need for optimization, while the tests $Q_{\tilde{i}}$ are still in progress. If there is more than one test finished at the same time, we refer to the appropriate test set notation.

We state $S(I, C_I) = \hat{\Theta}_C$ as the optimal schedule, which is derived by the scheduling function S under the set of tests I with constraints C_I. A schedule S starts at time $s = 0$ and let $\tilde{i}$ be the first test, which is finished at time $t_{\tilde{i}}$. Thus, rescheduling refers to the test set $\tilde{I} = I\backslash\{\tilde{i}\}$ at time $t_{\tilde{i}}$ as long as $\tilde{I} \neq$. Online scheduling of a test set I with constraints C_I is successively achieved by

$$\min_{\{s_i\}_{i\in I\backslash\{\tilde{i}\}}} \quad \max_{i\in I\backslash\{\tilde{i}\}} e_i \tag{2}$$

subject to $(\forall i \in I\backslash\{\tilde{i}\})$

$$\begin{aligned}
& s_{pre_i} + t_{pre_i} = s_i && \\
& s_j + t_j \leq s_i && \forall j \in P_i \\
& s_j + t_j \leq s_i \quad \vee \quad s_i + t_i \leq s_j && \forall j \in M_i \\
& r_{i,j} + \sum_{k\in Q_i} r_{k,j} \leq 100 && \forall j \in J \\
& current_status \sim c_{i,l} && \forall l \in L
\end{aligned}$$

with $\quad Q_i := \{k : k \text{ parallel to } i\}$.

Rescheduling defines an adapted schedule with test $\tilde{i}$ exclusive. As requirement R_{dyn} requires non-preemptive rescheduling in automotive manufacturing, there exists a set of tests $I\backslash\{\tilde{i} \cup Q_{\tilde{i}}\}$ that follow under the condition of actual run time $t_{\tilde{i}}$ and the status.

The model ensures that the testing schedules are as cost-efficient as possible, whereby cost is related to minimal makespan.

B. Concept and software architecture design

Based on the mathematical model including the rescheduling of Section IV-A, we present the concept for online scheduling of diagnostic tests in car manufacturing. Moreover, we discuss a model design to be used for software architecture. Figure 2 illustrates the underlying concept of online test scheduling as a flow chart. Given a subset of diagnostic tests $\tilde{I} \subset I$, the first step of the concept includes solving the mathematical model presented in Section IV-A given the defined constraints C. The solution specifies the schedule of tests to be executed on the car's HPC. It selects the first k tests that can be executed in parallel. After a test i is finished, rescheduling is processed with the remaining test set $\tilde{I}$ for a possible new and optimal arrangement at that new time e_i. This means that the tests k already running are not part of the new sequence. These are initially ignored under the IO assumption.

There are two disturbance cases: either the time limit of a test i or the end of a test spot is reached, or a test has failed with the status *not OK*. If all tests are status *OK*, the concept is finished.

This concept executes the tests in a time-minimal way under consideration of constraints, which leads to a resilience against failures and an advantage compared to a one-time optimal schedule of tests. In order to guarantee an optimal schedule at any time, the scheduling process is repeated after each execution of one or more tests in parallel when they are finished.

The architectural software design for online test scheduling in fundamental modeling concepts (FMC) notation is illustrated in Figure 3. The design contains a component called the *Event Managing Component*, which monitors and manages the communication of the *Process Engine* and *Scheduling Component*. The *Event Managing Component* receives a call from a *remote client*, i.e., a remote control system. Based on the call's information w.r.t. the car and its equipment, the *Event Managing Component* calls the *Scheduling Component* to calculate the optimal schedule for a car under scheduling constraints C. The constraints are stored for the car onboard and relate to the car's configuration. The process models that correspond to the tests in the schedule are located in the *Process Model Storage*. The schedule of diagnostic tests is processed by a *Process Engine*, which executes the corresponding test models in correspondence with the *Diagnostic Framework*, e.g., supporting an SOVD interface [3]. As the setup is just a schema for generalization, there might be more HPCs, gateways, and ECUs connected via bus systems. If a set of tests has been executed, the *Event Managing Component* initiates rescheduling based on the tests $\tilde{I}$.

V. Evaluation by an OEM case study

In this section, we evaluate the concept presented in Section IV. We look at implementation details and the baseline model, show data-based results, discuss them, and conclude

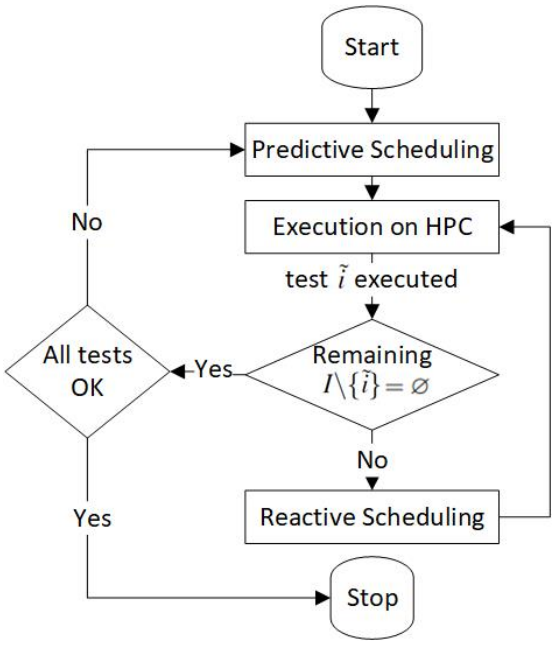


Fig. 2: Concept steps to perform online scheduling of diagnostic tests

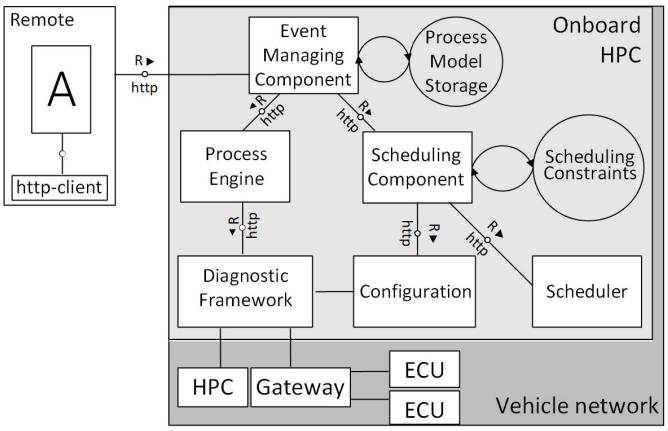


Fig. 3: Architectural diagram in FMC notation to design online scheduling on automotive HPCs

with an assessment of the requirements R_{sched}, R_{veh}, and R_{dyn}.

A. Implementation details

As automotive computing hardware, a *Smart Diagnostic Interface DE-4* device from DSA GmbH was used with an HPC emulator regarding the architecture and performance of an ARM64 Quadcore CPU with 18000 DMIPS Hypervisor. The components from Figure 3 are containerized as Docker containers on the HPC. The components are accessed via REST APIs with HTTP client to be appropriate for the diagnostics ASAM standard SOVD [3]. By not using HTTPS clients, we neglect an appropriate car security concept.

Algorithm 1 Pseudo code for the *Event Managing Component* in online scheduling

Input: Tests I with list of constraints C and k the number of tests to be executed in parallel w.r.t. a schedule
Output: Executed tests with status information (*OK* or *not OK*)

```
while |I| > 0 do
    POST request to Scheduler with tests I and parse response schedule s from Θ̂_C
    SchedulerResponse ← s
    executableTests ← SchedulerResponse.ExecutableTests()
    // Choose k tests from s that can be executed in parallel
    POST request to Process Engine with k tests
    repeat
        GET request to Process Engine and parse response e
    until at least one of the tests is completed or aborted
    EngineResponse ← e    // Information on active tests with execution status
    solvedTests ← EngineResponse.solvedTests
    failedTests ← EngineResponse.failedTests
    I ← cleanupTests(solvedTests, failedTests)    // remove executed tests from test set
    ManagerResponse ← ManagerResponse + solvedTests + failedTests
end while
return ManagerResponse
```

The car under diagnosis is a German OEM vehicle. The chosen setup is significant to validate the concept as a prototype in an automotive production environment.

As a baseline model, we consider offline scheduling, which is realized using the CP-SAT solver in Google OR-Tools. The offline schedule is calculated once and is not adjusted online. The adapted solver w.r.t [19] is a prototype in the programming language Python and is used as a core function in *Scheduler* for online scheduling as well. For $N < 30$ in the case study, the schedule calculation time is $\sim 100ms$. This is sufficient to be used for optimization during the run time on the HPC. The *Event Managing Component* is realized as a Python prototype based on Algorithm 1. The *Process Engine* is implemented in C++ following standards in the AUTOSAR Adaptive guideline.

B. Data collection

In the industrial OEM case study, we refer to a test spot at a German car assembly line. We consider a test bench where drive system variants are installed on cars, calibrated and tested. Each drive system is, in turn, realized by variants of ECUs depending on the customers' orders. For the case study, we consider the four variants of test distributions with the highest number of cars. The tests are distributed on four variants with $N_1 = 19, N_2 = 21, N_3 = 20, N_4 = 19$. Engineers and production planners specify test run times $t_i, i \in \{N_1, N_2, N_3, N_4\}$ and model dependencies as listed in Section IV-A. The range of test run times is between $1 - 60s$. These tests are candidates which would probably highly benefit from online scheduling, as the run times $\{t_i\}_{i \in I}$ are averaged estimates.

C. Evaluation criteria

To evaluate the concept, we apply qualitative and quantitative metrics. As the HPC is an embedded car system, resource usage in terms of processor CPU and RAM of the components are of interest. The turnaround time $|min(e_{makespan})|$ expresses how long an optimized test schedule runs, including rescheduling. It follows that the mean turnaround time mtt for online scheduling compared to offline scheduling for a set of tests var over a specific number of repetitions p is given by

$$mtt_{var}(p) := \sum_{j=1}^{p} \frac{|min(e_{makespan,j})| - min(e_{makespan,j})}{p}$$

Since the OEM's engineering process of planning and scheduling is an interdisciplinary interaction of engineers from development, assembly, and rework, it is also important to include qualitative criteria. It is, therefore, of interest to what extent online scheduling influences the work of the interdisciplinary team.

D. Results and discussion on case study

During the case study, CPU utilization on the HPC averaged 5% for the *Event Managing* and *Scheduling Components*. The *Process Engine Container*, based on the C++ engine, fluctuates between $10 - 80\%$ utilization, with the *Diagnostics Framework* accounting for the largest share. RAM utilization is low for the *Event Managing* and *Scheduling Components* at 2%, and the *Diagnostic Framework* uses $10 - 12\%$. Rescheduling takes between $40 - 100ms$ in our scenario, depending on the number of tests N. The results show that the application design is suitable for a prototype.

Offline and online scheduling with actual testing times during production for a case study variant with $N_1 = 19$ tests is compared in Figure 4. In Figure 4a, 19 tests are scheduled offline. Test $i = 3$ is highlighted in blue with an estimated run time $t_3 = 60s$. It is a function test of the air conditioner during manufacturing. In Figure 4b, the schedule is adapted during online scheduling. Test 3 is actually executed in $\tilde{t}_3 = 30s$. The online scheduling approach allows us to perform rescheduling after completion of test t_3, which leads to an optimized schedule that is $25s$ faster compared to the offline calculated schedule. Predictive-reactive scheduling results in a positive time optimization.

In order to evaluate the flaw of completely reactive scheduling, a study was conducted to calculate the mean time saved by online compared to offline scheduling. For this purpose, we took the four variants and calculated the average difference in the turnaround time between online and offline scheduling in $p = 10$ cycles.

$$mtt_{1,2,3,4}(10) \approx 4$$

The average time calculated by online scheduling is $4s$ shorter than in offline scheduling. The result can be interpreted as meaning online scheduling is slightly better w.r.t. the total

turnaround time than offline scheduling in the defined setup. The reason lies in the rule set of the variant which is tightly coupled by experienced engineers and allows almost no optimization by rescheduling. This is the flaw of online scheduling, which arises through completely reactive scheduling.

If the planners' time estimates match the actual run times, there is no time advantage from online scheduling because the test schedule is identical to offline scheduling. In the limiting case, no run times have to be planned in the long term, since the set of tests to be executed is calculated online based on dependency rules. Estimations $\{t_i\}_{i \in I}$ are no longer required.

On the one hand, predictive-reactive scheduling can be used to react flexibly to changes in the run time. The overall test time of a vehicle can be minimized if there are options to adapt the initial schedule. Since the planners no longer have to estimate test times in collaboration with engineers for the e/e components, they are relieved in their work. On the other hand, employees will also benefit from this approach. Managing variability in testing schedules is automated by software algorithms. The evaluation of online scheduling showed the necessity of choosing predictive-reactive scheduling in the concept, and thus, provides the answer to research question RQ2.

E. Assessment of requirements R_{sched}, R_{veh}, and R_{dyn}

We access the results of the case study in terms of the requirements derived in Section II.

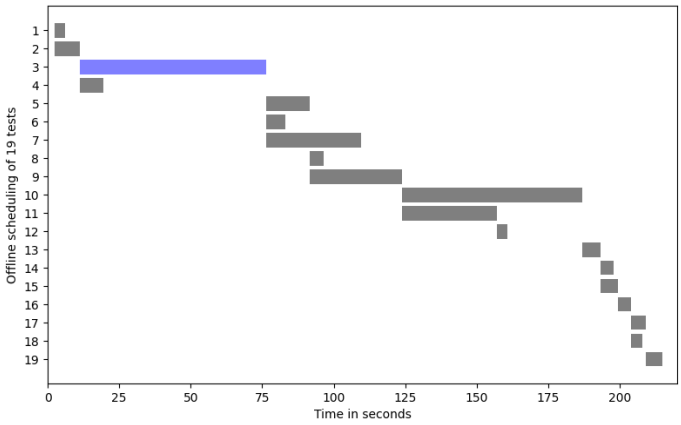
Requirement R_{sched} is fulfilled by the model design in Section IV-A. Requirement R_{veh} is fulfilled by the architectural design illustrated in Figure 3. As the implementations of the *Event Managing* and *Scheduling Components* are Python prototypes, their industrialization, e.g., in C++, should be considered to better utilize the vehicle resources. Requirement R_{dyn} is partly fulfilled: in the case study design, online scheduling was taken into account in a dynamic environment where tests take different times $\{t_i\}_{i \in I}$. A complete defect management concept in the context of failed tests has not been considered. The interaction of the manufacturing workers with the vehicle was neglected. A delay in manufacturing work caused by workers results in a delay of a test i.

a) Scalability: The concept's scalability depends on the vehicle resources RAM, CPU, and the computation time of the rescheduling process w.r.t. $\tilde{I}$. In our setup, the *Scheduler* in Figure 3 computes a new schedule for $N < 30$ tests in $< 100ms$. If the scheduling problems to be optimized depend on more tests $N >> 30$ and related constraints C, the resource utilization should be checked. If the computation time is no longer appropriate to the turnaround times of the individual tests, the rescheduling strategy should be checked. Otherwise, the benefit of online scheduling may be reduced.

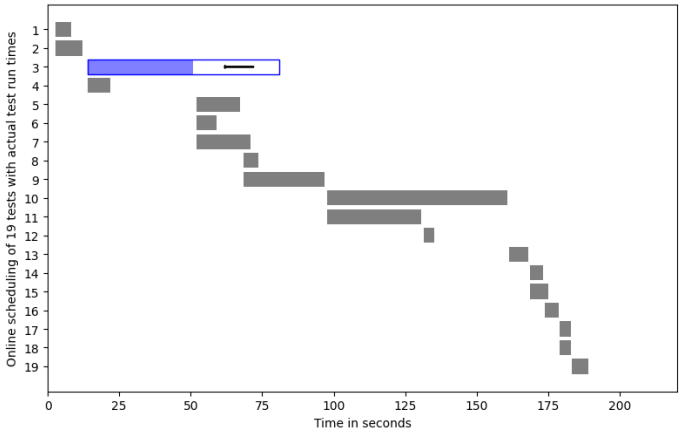
VI. Summary and Outlook

The number of automotive software components in e/e architecture is increasing, and so is the importance of testing components and diagnostics. The factory IT infrastructure and the car itself can be used to calculate the schedule of diagnostic tests and calibrations online. In this work, we have developed a concept for online scheduling of tests on cars in production lines. The concept is designed for a service-oriented car architecture. It has been successfully evaluated on the basis of an OEM case study in a German car production plant. The advantages of online scheduling come into play when the planned and actual run times of tests deviate, and rescheduling carries out a time-optimized schedule. The concept relieves employees in their work through automation solutions and maximizes the use of time resources in automotive manufacturing.

In further research, a collaborative approach for cars will be explored to improve scheduling globally.



(a) Offline schedule of 19 tests



(b) Online optimized schedule of 19 tests

Fig. 4: Illustration of the benefit of online compared to offline scheduling: online scheduling reduces total time by rescheduling after test $i = 3$ has been executed

ACKNOWLEDGEMENT

The publication is based on the research project SofDCar (19S21002), that is funded by the German Federal Ministry for Economic Affairs and Climate Action.

References

[1] C. Fehling, M. Frank, and O. Kopp, "Digital sustainability and digital diversification: The two key challenges for automotive software development," in *18th IEEE International Conference on Software Architecture Companion, ICSA Companion 2021, Stuttgart, Germany, March 22-26, 2021*. IEEE, 2021, pp. 162–166. [Online]. Available: https://doi.org/10.1109/ICSA-C52384.2021.00039

[2] D. Reinhardt, U. Dannebaum, M. Scheffer, and M. Traub, "High performance processor architecture for automotive large scaled integrated systems within the european processor initiative research project," S. T. P. 2019-01-0118, Ed. SAE Technical Paper, 04 2019.

[3] ASAM e. V., "Service-oriented vehicle diagnostics," 2022. [Online]. Available: https://www.asam.net/standards/detail/sovd/

[4] T. Bohnenberger, K. Fischer, and C. Gerber, "Agents in manufacturing: online scheduling and production plant configuration," in *Proceedings. First and Third International Symposium on Agent Systems Applications, and Mobile Agents*, 1999, pp. 66–73.

[5] M. Gaid, A. Cela, and Y. Hamam, "Optimal integrated control and scheduling of networked control systems with communication constraints: application to a car suspension system," *IEEE Transactions on Control Systems Technology*, vol. 14, no. 4, pp. 776–787, 2006.

[6] H. Sohn, J. D. Lee, D. L. Bricker, and J. D. Hoffman, "A dynamic programming algorithm for scheduling in-vehicle messages," *IEEE Transactions on Intelligent Transportation Systems*, vol. 9, no. 2, pp. 226–234, 2008.

[7] L. Zhang, D. Roy, P. Mundhenk, and S. Chakraborty, "Schedule management framework for cloud-based future automotive software systems," in *2016 IEEE 22nd International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA)*, 2016, pp. 12–21.

[8] G. Xie, G. Zeng, Z. Li, R. Li, and K. Li, "Adaptive dynamic scheduling on multifunctional mixed-criticality automotive cyber-physical systems," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 8, pp. 6676–6692, 2017.

[9] J. Dvořák and Z. Hanzálek, "Multi-variant scheduling of critical time-triggered communication in incremental development process: Application to flexray," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 1, pp. 155–169, 2019.

[10] M. Faizan and A. S. Pillai, "Dynamic task allocation and scheduling for multicore electronics control unit (ecu)," in *2019 3rd International Conference on Electronics, Communication and Aerospace Technology (ICECA)*, 2019, pp. 821–826.

[11] M. Zhao, X. Li, L. Gao, L. Wang, and M. Xiao, "An improved q-learning based rescheduling method for flexible job-shops with machine failures," in *2019 IEEE 15th International Conference on Automation Science and Engineering (CASE)*, 2019, pp. 331–337.

[12] K. Land, B. Vogel-Heuser, and S. Cha, "Applying dynamic programming to test case scheduling for automated production systems," in *Systems Modelling and Management*, Ö. Babur, J. Denil, and B. Vogel-Heuser, Eds. Cham: Springer International Publishing, 2020, pp. 3–20.

[13] V. Pradeep Kumar and A. S. Pillai, "Dynamic scheduling algorithm for automotive safety critical systems," in *2020 Fourth International Conference on Computing Methodologies and Communication (ICCMC)*, 2020, pp. 815–820.

[14] Z. Wang, S. Zheng, Q. Ge, and K. Li, "Online offloading scheduling and resource allocation algorithms for vehicular edge computing system," *IEEE Access*, vol. 8, pp. 52 428–52 442, 2020.

[15] B. Hu, Y. Shi, Z. Cao, and M. Zhou, "A hybrid scheduling framework for mixed real-time tasks in an automotive system with vehicular network," *IEEE Transactions on Cloud Computing*, pp. 1–14, 2022.

[16] J. Błażewicz, K. Ecker, E. Pesch, G. Schmidt, M. Sterna, and J. Weglarz, *Handbook on Scheduling: From Theory to Practice*, ser. International handbooks on information systems. Springer, 2019. [Online]. Available: https://books.google.de/books?id=A9dNygEACAAJ

[17] D. Ouelhadj and S. Petrovic, "A survey of dynamic scheduling in manufacturing systems," *J. Scheduling*, vol. 12, pp. 417–431, 08 2009.

[18] J. W. Herrmann, *Rescheduling Strategies, Policies, and Methods*. Boston, MA: Springer US, 2006, pp. 135–148. [Online]. Available: https://doi.org/10.1007/0-387-33117-4_6

[19] S. König, M. Reihn, F. G. Abujamra, A. Novy, and B. Vogel-Heuser, "Flexible scheduling of diagnostic tests in automotive manufacturing," *Flexible Services and Manufacturing Journal*, Jan 2022. [Online]. Available: https://doi.org/10.1007/s10696-021-09438-3

All links were last followed on January 31, 2023.

Paper5

BPMN4CARS: A Car-Tailored Workflow Engine

Simone König, Birgit Vogel-Heuser *IEEE Fellow*, Jan Wilch
Institute of Automation and Information Systems
Technical University of Munich
Garching, Germany

Tobias Unger
Opitz Consulting GmbH
Bad Homburg, Germany

Michael Hahn, Stjepan Soldo, Oliver Kopp
Mercedes-Benz AG
Sindelfingen, Germany

2023 IEEE 21st International Conference on Industrial Informatics (INDIN) | 978-1-6654-9313-0/23/$31.00 ©2023 IEEE | DOI: 10.1109/INDIN51400.2023.10218082

***Abstract*—The importance of high-performance computers in car networks increases to realize software assistance functions. These computers offer new opportunities for use cases in car testing and diagnostics towards vehicle software and services. Therefore, car testing and diagnostics are no longer limited to the testing and functional checking of electric and electronic components. Proven concepts from computer science, such as business process management, can be integrated into the car network to provide a uniform basis for automotive use cases. In this paper, (i) the Business Process Model and Notation (BPMN) standard is used to model diagnostic processes in future car production lines and (ii) a car-tailored BPMN workflow engine is introduced to execute the processes in a high-performance vehicular computer. The research on executable BPMN models in car networks bridges the gap between interdisciplinary business processes and execution in automotive IT systems.**

***Index Terms*—Software-defined car, Service-oriented vehicle diagnostics, Control and testing, Car-tailored workflow engine, Executing BPMN**

I. MOTIVATION FOR EXECUTABLE BPMN PROCESSES IN CAR NETWORKS

The digitization of the car and its interfaces to the environment means proven web technologies from computer science can now be applied in the cars. For instance, the standardized future in-car diagnostics interface for *Service-Oriented Vehicle Diagnostics (SOVD)* [1] is following a service-oriented approach providing diagnostic services of a car in form of a RESTful HTTP API. As a consequence, the technique of service orchestration using workflows [2] can be used to implement diagnostic processes in a modern way, as these processes are specified as sequences of service invocations.

Testing and diagnostic processes are cross-company business processes to test, inspect, and commission electrical and electronic components in cars during assembly line production. The processes are reoccurring for car lines and can be specified in terms of sequences. Subsequently, these sequences are executed in the car network using the modern *SOVD API*.

The Business Process Model and Notation (BPMN) standard [3] for modeling business processes is well-known in business informatics. BPMN supports the user in the graphical modeling of business processes, whereby, e.g., activities, events and gateways are represented by icons. User interaction is possible by user tasks. The BPMN data flow is defined in relation to the sequence flow such that parameterization of processes is visually supported. Besides offering a graphical notation, BPMN specifies process execution semantics. It describes how the theoretical concept of a token traverses the sequence flow by passing through the BPMN elements in a specified business process model.

Commercial tools for the BPMN modeler and engine exist, such that the maintainability of the tools is in general not limited to traditional automotive suppliers. Camunda Services GmbH [4] have already produced a Java-based solution called Camunda Workflow Engine. The concept of BPMN can reduce IT system media breaks for interdisciplinary teams with their software-based products, as the specified business process is executed directly. König et al. [5] have already dealt with the modeling of real testing, diagnosis, and commissioning processes using the BPMN standard. This paper follows the approach and will focus on the in-car execution of BPMN processes, that is called BPMN4CARS, in this work.

The overall business processes setup in BPMN4CARS is illustrated in Figure 1. First, an engineer uses a *BPMN modeler* to model a test process for an electrical component w.r.t. a car line. The resulting model is specified as *bpmn.xml* and is stored in a global *BPMN model repository*. The repository ensures that all relevant engineers can adapt the model during the interdisciplinary work. For execution, the process model is deployed from the *OEM backend* to the car and is then executed by the in-car workflow engine. In the setting of car

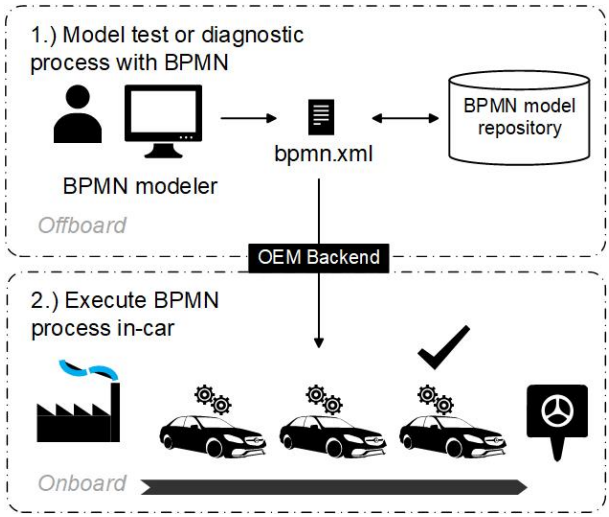


Fig. 1: Business processes setup in BPMN4CARS

production, there is typically one process instance of each process model running on the car's in-car workflow engine.

Modern cars are equipped with automotive high-performance computers (HPCs), which contain complex assistance functions, and in principle, allow the integration of a BPMN workflow engine. However, there is no systematic approach to evaluate existing BPMN workflow engine implementations. The authors claim that existing workflow engines have a too high CPU usage and memory footprint to be used as an in-car workflow engine in a car production line.

The main contribution of this paper is the concept BPMN4CARS for the execution of BPMN-modeled processes in the car network using an automotive workflow engine. A car customized workflow engine is introduced for car testing and diagnostic business processes in future production lines.

The structure of the paper is as follows: In Sect. II, requirements for an in-car workflow management system, including a BPMN-based workflow engine, are presented. Related work is discussed in Sect. III. In Sect. IV, the concept BPMN4CARS is presented. In Sect. V, two workflow engines are benchmarked for evaluation. Sect. VI concludes the paper.

II. REQUIREMENTS FOR A BPMN-BASED IN-CAR WORKFLOW ENGINE

As a basis for the requirements for an in-car workflow engine, the structure of an assembly line, interviews within a German automotive original equipment manufacturer (OEM) assembling cars, the specification of the future diagnostics interface *SOVD* [1], and the specification of the AUTOSAR Adaptive Platform [6] are taken into account. Based on that, the following seven requirements can be derived:

1) OEM internal expert interviews summarize that test cases for car testing and commissioning during production should be implemented with as little programming know-how as possible. Engineers might be already familiar with the OTX modeling standard [7] that exists under the SOVD predecessor ODX diagnostic standard, so a user-friendly graphical modeling notation is a requirement.
2) From the standard *SOVD* [1], it can be deduced that web services based on RESTful HTTP APIs play an increasingly important role in car networks. During car production, the interaction of the worker with the car needs to be supported. Thus, the concept must include artifacts as service tasks to call car internal APIs as well as user tasks for worker-car interaction.
3) A concept should support seamless integration of specification and execution of diagnostic processes from the OEM backend to the car.
4) Typical BPMN application scenarios have thousands of process instances running on a single BPMN engine [8]. The engine is scaled as needed to cover the workload. In those scenarios, the latency requirement on the time between the caller and the callee is not important. Moreover, the APIs are available generally. In contrast, in car production settings, the production-related APIs are opened to the production environment only and there exist strict latency requirements in millisecond range, e.g., in window scaling, or in communication with test benches. Some latency requirements are mandatory for safety or certification-relevant topics. As a consequence, the workflow engine must be placed in an individual car to execute exactly those diagnostic processes required for this concrete car within the required latency boundaries. Thus, not a single workflow engine executes many instances of the same process model, but multiple workflow engines in different cars execute single instances of the same model.
5) Car production is an assembly line process, where the KPI *Overall Equipment Effectiveness (OEE)* is maximized. To increase OEE through increased performance, testing and commissioning scopes should take as little time as possible. This results in a short start phase for the workflow engine.
6) Since modern vehicles contain powerful computers such as HPCs, there is an option to use the automotive network for the execution of the test cases. The memory and CPU consumption should have a low footprint during the runtime of the workflow engine.
7) The software components should match with automotive standards as the AUTOSAR Adaptive guideline [6], where C++ is recommended as a programming language.

The seven requirements build the basis for the an overall in-car workflow execution concept. Following the work of König et al. [5], this work opts for BPMN as modeling standard to fulfill the first three requirements presented above.

III. RELATED WORK

The *Business Process Model and Notation (BPMN)* [3] standard is known and applied throughout the industry. The paper wants to adapt concepts that cover the execution semantics of business processes, as this technology is reoccurring under different business and IT contexts.

BPMN workflow engines have been benchmarked by Ferme et al. [9]. The setting was that there are hundreds of instances of the same process models. In the setting of car production, however, there is one process instance per process model in an individual car. The efficient orchestration of multiple process models within a car is achieved by a scheduling mechanism, e.g., as introduced by König et al. [10]. Thus, the benchmarking setting is completely different and a call for an automotive benchmark is justified.

Lenhard et al. [11] present general lessons learned about workflow-engine benchmarking. The main findings are issues of model portability and interaction with the API of the benchmarked workflow management systems. The measurements conducted in the context of this work are not hindered by such issues as the applied test workflows use common elements and only two deployment alternatives exist.

Traganos et al. [12] present a BPMN-based Printing Process Management System where the modeling and orchestration of printing business processes are enabled. The process engine is

integrated into the *global level* of a cyber-physical system. The authors do not specify further details on the process engine.

Caracas [13] discusses the application of BPMN towards orchestration of wireless sensor networks (WSN). There, the BPMN workflow is mapped to WSN-specific code, which is distributed among the nodes in the network. That approach is not applicable in the car production setting.

Chis and Ghiran [14] extend BPMN to be able to cope with services using the Constrained Application Protocol (CoAP). The extension was implemented in a desktop tool offering modeling and execution. That execution environment cannot be used in car production settings.

Vanderfeesten et al. [15] developed an information system to control the execution of high-tech manufacturing processes using BPMN. The proposed HORSE system architecture aims to bridge the Internet of Things-oriented level of collaborative robotics and the BPM-oriented level of manufacturing processes. The system architecture includes local and global execution engines from Camunda Services GmbH. However, the authors do not cover the in-car specific requirements for a BPMN engine presented in Sect. II.

Camunda Services GmbH published a study where process automation, including the Camunda Workflow Engine, is already applied in industrial fields [16].

To best of our knowledge, there is no work dealing with the in-car execution of test processes using the BPMN standard or regarding concepts for workflow engines fulfilling the in-car specific requirements presented in Sect. II.

IV. BPMN4Cars: A Concept to Execute BPMN Processes with a Car-Tailored Workflow Engine

The main contribution of this work is presented in terms of the concept BPMN4Cars for the in-car execution of BPMN-modeled processes based on the requirements discussed in Sect. II. The design is driven by the generic approach to integrate a BPMN workflow engine within the car, which is open for in-car applications.

The workflow engine is placed in individual cars and it executes the process models required for a concrete car. This circumstance leads to a specialized in-car BPMN workflow system architecture, which is sketched in Figure 2 and detailed in Figure 3.

The software system architecture for integrating the BPMN4Cars concept into the car network is shown in Figure 2. The *BPMN Workflow Engine* is C++ based, is located on an in-car *HPC* and executes the car-specific BPMN test processes. More details on the BPMN test processes are discussed as part of the introduction of the detailed architecture shown in Figure 3. In general, the workflow engine calls respective endpoints for web service calls to communicate either with the *User Task Management* or with *In-car Applications*. The former provides the required functionality for the interaction with workers at the assembly line. The latter is a placeholder for further in-car components and the services they expose through respective interfaces. Such components can be *HPCs* or vehicle *gateways*, for example, which form

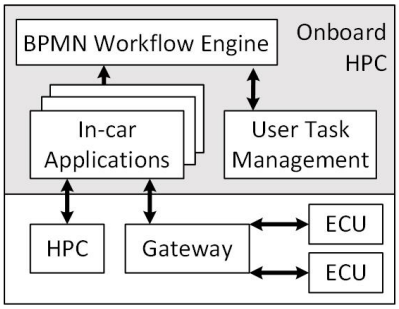


Fig. 2: System architecture to realize BPMN4Cars within the car network

interfaces to individual ECUs. Web service calls are realized within *BPMN service tasks*. A BPMN service task is a task that uses some sort of service, which could be a web service or an automated application [3].

The engine internal component architecture is shown in Figure 3. The architecture follows the three tier system structure, as e.g., outlined for workflow management systems by Leymann and Roller [17]. The *Presentation Layer* includes the front end application with a REST API, which is accessed by users, e.g., engineers, or other onboard as well as offboard components to send requests to the business logic layer. For example, deploying a new process model to the workflow engine or instantiating a deployed process model to start the execution of a new process instance.

The *Business Logic Layer* is responsible for the various types of business operations of the BPMN workflow engine. It mainly consists of eight components, which will be introduced in the following in more detail. The *BPMN Models* component provides the underlying BPMN metamodel to represent BPMN process models and their contained elements. Important elements are BPMN *service tasks* specified to invoke web services or *call activities* to call other process models deployed to the engine as subprocesses. The *Deployment Controller* manages the deployment of BPMN process models, e.g., by engineers, to the workflow engine. This comprises the validation of the provided files as well as forwarding the data to the respective components responsible for persisting and managing the internal representation of the underlying process models using the *BPMN Models* component. Therefore, the *Process Manager* creates a new process model representation and makes it accessible within the engine. In general, the *Process Manager* component is responsible for the organization and management of process models and their lifecycle state. Based on an engine-internal process model representation, new process instance can be created and managed via the *Instance Manager* component. This comprises the management of the lifecycle state of the instances of process models as well as their associated runtime data. Possible process instance lifecycle states are pending, running, waiting, aborted and completed. Both, the *Process Manager* and *Instance Manager* components are also responsible for persisting the engine-internal representations and their associated data to

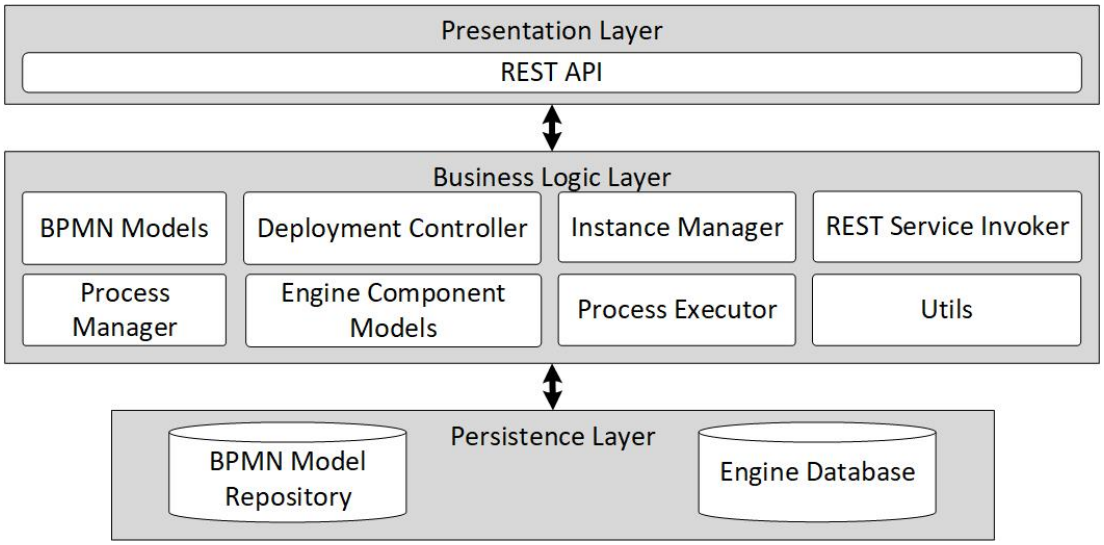


Fig. 3: Component architecture for the prototype BPMN Workflow Engine, which is introduced in Figure 2

the engine database. The *Process Executor* component is conducting the created process instances by navigating through the underlying BPMN process model based on the engine-internal representation provided by the *Process Manager* component. The execution logic of each of the respective *BPMN Models* components used to represent the elements of a process model are provided by the *Engine Component Models*. They define the engine-internal implementations of BPMN model elements, such as service tasks, user tasks, sequence flows, data inputs, etc., in form of respective C++ classes. The *Process Executor* orchestrates this functionality to execute the control and data flow specified within the process model underlying to a currently executed process instance. The *REST Service Invoker* component is responsible for calling engine-external applications through their REST APIs. An example for such an in-car application providing a REST API, is the SOVD API [1] providing access to all diagnostic services of the car. The *Utils* component includes utility functions such as an universally unique identifiers (UUID) generator for providing unique process instance identifiers, or a JSON query (jq) transformer to filter and process JSON data as part of specified data associations, e.g., in the context of a REST API call within a BPMN service task.

With the *Persistence Layer*, the logic to read, write or delete data is isolated from the business logic. It includes the car-internal *BPMN Model Repository* and the *Engine Database*. The Model Repository stores the deployed process models and their related metadata. The Engine Database stores all engine-internal data. This comprises, for example, data about executed process instances, their status, or related execution events (audit trails).

V. Evaluation of the in-car Workflow Engine

This section covers the evaluation of the presented concept BPMN4Cars in terms of the introduced car-tailored workflow engine. On the one hand, the authors want to fulfill the business requirements discussed in Sect. II, on the other hand, they are interested in re-using existing and commercial software as much as possible within the production domain. Therefore, a comparison of two workflow engines seems appropriate. For this purpose, a benchmark of the car-tailored workflow engine presented in Sect. IV against the BPMN workflow engine from Camunda Services GmbH [4] is of interest.

A. Benchmarking against Camunda BPMN Workflow Engine

For the benchmarking the Camunda Workflow Engine is used, which is a software product of the company Camunda Services GmbH and is used across industries for process automation using BPMN workflows. Since this engine is built on Java, the authors assume that its application domain is mostly in IT enterprise systems and less in the embedded systems domain. Therefore, the conducted benchmark compares the Camunda Workflow Engine against the C++ BPMN Workflow Engine presented within this work.

The benchmark criteria applied for the comparison of the two workflow engines follows the requirements $5)$ and $6)$ introduced in Sect. II, which are:

- start phase duration,
- memory consumption, and
- CPU consumption as footprint.

B. Evaluation Setup

The evaluation setup is compared w.r.t. the footprint of both workflow engines on Intel® Core™ i7-10750H CPU @ 2.60GHz, 12 cores, 32 GB RAM with operating system Ubuntu 20.04 LTS, kernel: 5.10.16.3-microsoft-standard-WSL2. The Camunda setup includes Camunda 7.16.0, Quarkus 2.1.12.final, and Java OpenJDK runtime Temurin-11.0.14+9 (build 11.0.14+9). The evaluation is conducted on a desktop device, however, it is comparable to future automotive HPC device scenarios. The overall time duration for the evaluation setup is $10s$. Each engine gets the same workflow deployed: That

workflow first receives a message and then sends the message to another http endpoint. This minimal process was chosen to focus on the minimal time needed that an engine handles a request after startup. For measurement, one call was sent to the workflow engine and then the CPU and memory consumption was measured. The CPU and memory usage was measured using the Linux top tool.

C. Observations

The measured footprints of both engines are illustrated in Figure 4. The CPU usage of the car-tailored workflow engine, shown in Figure 4a, is uniformly distributed, even during the start phase. The CPU usage being 0 is an indicator that the logic is very lean and does not need any real database interactions, etc. The memory consumption is about 17.5 MB throughout the measurement phase. As Figure 4b shows, the CPU usage of the Camunda BPMN Workflow Engine is not uniformly distributed during the start phase. It is even more than 100% load, which indicates that multiple cores have been used during startup. The memory consumption is steadily increasing during the start phase up to 600 MB throughout the whole measurement phase.

The car-tailored workflow engine shows a lower and in general more steady footprint in terms of memory consumption and CPU usage than the commercial Camunda workflow engine. The footprint does not increase significantly during the start phase either. This behavior favors the car-tailored engine in terms of requirement 6). The duration of the start phase of the Camunda engine is $4s$, whereas the car-tailored workflow engine starts after $100ms$. A short start phase is essential for flexible production lines (requirement 5)) as production planning does not provide waiting or reboot times of several seconds. Moreover, the design and implementation of the car-tailored workflow engine ensure that requirements 6) and 7) from Sect. II are fulfilled. The conceptual design of BPMN4CARS fulfills requirement 4).

To sum up, the prototypical implementation of a car-tailored workflow engine for car diagnostics in future production lines is mandatory based on the presented benchmark results. The underlying concept BPMN4CARS fulfills all seven requirements presented in Sect. II.

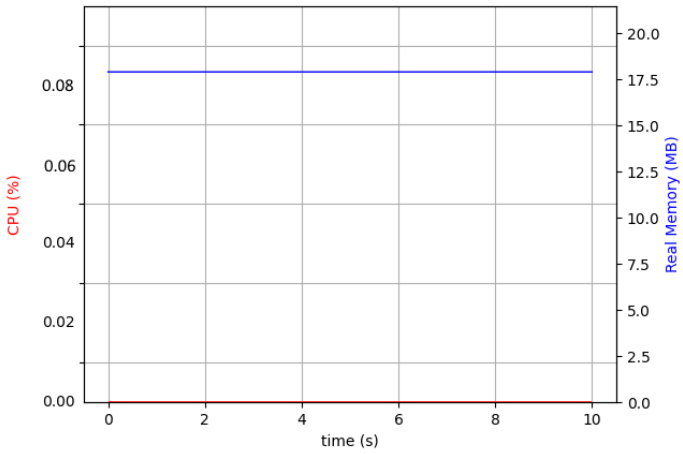
D. Limitations

The presented car-tailored workflow engine is a software prototype. The prototype does not yet support all BPMN elements and auditing w.r.t. Leymann and Roller [17]. Since the focus of the evaluation is on the feasibility of the concept BPMN4CARS, the limitation is acceptable. We focused on the CPU and memory consumption. A holistic comparison has to cover the supported BPMN elements and more [18].

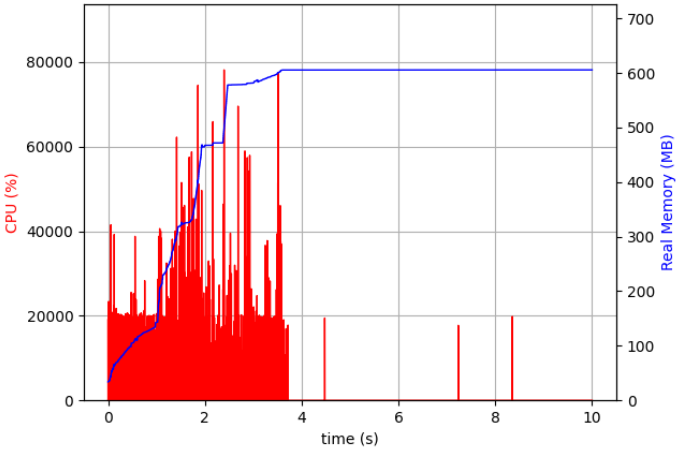
VI. CONCLUSIONS AND FUTURE WORK

Automated cars adapt to the customer and the environment through software assistance functions, which are implemented in high-performance computers within cars. To ensure that the risen complexity in the car does not overload the automotive OEMs in their management of internal business processes, the use of executable process models can facilitate the engineers' work. By transferring the concept of the cross-industry standard BPMN to the car, the interaction between engineers and the car is simplified, as specification documents are modeled as BPMN workflows, which can then be directly executed by a workflow engine inside the car.

In this paper, the concept BPMN4CARS is presented, which includes the in-car execution of BPMN-based process models in the context of car testing and diagnostics. The successful evaluation included the design of a car-tailored workflow engine to execute the BPMN process models in an in-car HPC. Compared to a popular commercial BPMN workflow engine, the use of a specialized automotive workflow engine



a Footprint of the Car-tailored BPMN Workflow Engine



b Footprint of Camunda BPMN Workflow Engine within Quarkus Framework

Fig. 4: Comparison of the footprints for the car-tailored and the Camunda BPMN workflow engine in terms of CPU and memory usage

built on C++ is mandatory.

In future work, aspects on variant management of testing and diagnostics processes for car lines using the introduced car-tailored workflow engine might be investigated. Moreover, the potential incorporation of the presented BPMN4CARS concept into new and upcoming automotive standards, such as ASAM *Service-Oriented Vehicle Diagnostics (SOVD)* [1], should be evaluated.

ACKNOWLEDGEMENT

This publication is based on the research project SofDCar (19S21002), which is funded by the German Federal Ministry for Economic Affairs and Climate Action.

REFERENCES

[1] ASAM e. V., "Service-Oriented Vehicle Diagnostics," 2022. [Online]. Available: https://www.asam.net/standards/detail/sovd/

[2] C. Pautasso, "BPMN for REST," in *Third International Business Process Modeling Notation Workshop (BPMN 2011)*, 2011.

[3] OMG, *Business Process Model and Notation (BPMN), Version 2.0*, Object Management Group Std., Rev. 2.0, January 2011. [Online]. Available: http://www.omg.org/spec/BPMN/2.0

[4] Camunda Services GmbH. [Online]. Available: https://camunda.com/platform-7/workflow-engine/

[5] S. König, B. Vogel-Heuser, E. Fieg, M. Hahn, and O. Kopp, "Modelling production workflows in automotive manufacturing," in *2021 IEEE 23rd Conference on Business Informatics (CBI)*, vol. 02, 2021, pp. 39–46.

[6] AUTOSAR GbR, "AUTOSAR Adaptive Platform," 2022. [Online]. Available: https://www.autosar.org/standards/adaptive-platform/

[7] ISO, *Road vehicles — Open Test sequence eXchange format (OTX) – ISO 13209*, International Organization for Standardization Std., 2011. [Online]. Available: https://www.iso.org/standard/53507.html

[8] B. Rücker and J. Freund, *Real-Life BPMN (4th edition)*. Independently Published, 2019.

[9] V. Ferme, M. Skouradaki, A. Ivanchikj, C. Pautasso, and F. Leymann, "Performance comparison between BPMN 2.0 workflow management systems versions," in *Enterprise, Business-Process and Information Systems Modeling*. Springer International Publishing, 2017.

[10] S. König, M. Reihn, F. G. Abujamra, A. Novy, and B. Vogel-Heuser, "Flexible scheduling of diagnostic tests in automotive manufacturing," *Flexible Services and Manufacturing Journal*, Jan 2022. [Online]. Available: https://doi.org/10.1007/s10696-021-09438-3

[11] J. Lenhard, V. Ferme, S. Harrer, M. Geiger, and C. Pautasso, "Lessons learned from evaluating workflow management systems," in *Service-Oriented Computing – ICSOC 2017 Workshops*. Springer International Publishing, 2018.

[12] K. Traganos, I. Vanderfeesten, P. Grefen, J. Erasmus, T. Gerrits, and W. Verhofstad, "End-to-end production process orchestration for smart printing factories: An application in industry," in *2020 IEEE 24th International Enterprise Distributed Object Computing Conference (EDOC)*, 2020, pp. 155–164.

[13] A. Caracas, "From business process models to pervasive applications: Synchronization and optimization," in *2012 IEEE International Conference on Pervasive Computing and Communications Workshops*. IEEE, 2012.

[14] A. Chis and A.-M. Ghiran, *BPMN Extension for Multi-Protocol DataOrchestration*. Cham: Springer International Publishing, 2022, pp. 639–656.

[15] I. Vanderfeesten, J. Erasmus, K. Traganos, P. Bouklis, A. Garbi, G. Boultadakis, R. Dijkman, and P. Grefen, *Developing Process Execution Support for High-Tech Manufacturing Processes*. Cham: Springer International Publishing, 2019, pp. 113–142. [Online]. Available: https://doi.org/10.1007/978-3-030-17666-2_6

[16] Camunda Services GmbH, "Camunda Compared to Alternatives," 2022. [Online]. Available: https://page.camunda.com/wp-camunda-compared-to-alternatives

[17] F. Leymann and D. Roller, *Production Workflow – Concepts and Techniques*. Prentice Hall PTR, 2000.

[18] S. Harrer *et al.*, "A pattern language for workflow engine conformance and performance benchmarking," in *EuroPLoP*. ACM, 2017.

All links were last followed on May 23, 2023.

Paper6

Material from: S. König, B. Vogel-Heuser, A. Hradecky *et al.,* “Executing automotive test workflows with BPMN and web service orchestration in software-defined car manufacturing”, *Production Engineering Research and Development,* 2024, Springer Nature.

Production Engineering
https://doi.org/10.1007/s11740-024-01302-1

ORIGINAL PAPER

Executing automotive test workflows with BPMN and web service orchestration in software-defined car manufacturing

Simone König[1] · Birgit Vogel-Heuser[2,3,4] · Adam Hradecky[1] · Sophie Horn[1] · Michael Hahn[1]

Received: 5 May 2024 / Accepted: 9 July 2024

Abstract
Automotive original equipment manufacturers (OEMs) are facing the challenge of keeping their cars innovative for the customer by integrating new advanced software functions, e.g., to provide higher levels of automated driving. Nonetheless, internal business processes should stay efficient to support the transformation despite the increasing complexity caused by new information and car technologies. For engineering and production processes where interdisciplinary teams of engineers, software developers, and workers interact along the car life cycle, the concept of executable process models offers an opportunity for work streamlining and process automation by web service orchestration. This paper will shed light on using executable process models under the cross-industry Business Process Model and Notation (BPMN) standard, along with the example of automotive test and diagnostic processes in smart manufacturing. We introduce a lean approach that includes the enterprise IT architecture for process modeling and the product car for process execution. Finally, we present our evaluation of the approach through a focus group with OEM experts.

Keywords Executable BPMN · Software-defined car manufacturing · Lean test workflow management

1 Introduction to automotive trends

To be attractive to customers, cars include a rising number of software-assisted functions, which are essential for all levels of vehicle autonomy. The implementation of these functions implies that the number of software components in the car is also increasing and thus, original equipment manufacturers (OEMs) develop new underlying software and electrical & electronic (E/E) architectures. Supportive of this, automotive development partnerships such as AUTOSAR GbR and ASAM e.V. publish new standards for automotive software development or diagnostics for software-defined cars to provide a modern and flexible set of tools, concepts, and technologies.

E.g., the AUTOSAR Adaptive Platform supports a runtime environment for separated hardware and software in electrical control units (ECUs) as well as for central car clusters including high-performance computers (HPCs) [1]. ASAM e.V. proposes software-oriented vehicle diagnostics (SOVD) for diagnosing and communicating with cars over a RESTful HTTP API [2].

The technological change within the car influences the OEMs' internal engineering and production processes as well. On the one hand, processes are adapted to be flexible to respond to agile changes in the production process arising from new car software updates. On the other hand, internal OEM processes should be streamlined to provide an additional budget for the digital transformation. In this context, the term software-defined Manufacturing aims to decouple physical production hardware from the control software,

✉ Simone König
simone.koenig@tum.de

Birgit Vogel-Heuser
vogel-heuser@tum.de

Adam Hradecky
adam.hradecky@mercedes-benz.com

Sophie Horn
sophie.horn@mercedes-benz.com

Michael Hahn
michael.a.hahn@mercedes-benz.com

[1] Mercedes-Benz AG, Sindelfingen, Germany

[2] Institute of Automation and Information Systems, Technical University of Munich, Garching, Germany

[3] MDSI, Munich, Germany

[4] MIRMI, Munich, Germany

Published online: 20 July 2024

which requires a service-oriented, modular architecture with virtualization and abstraction layers [3].

Business process management can serve with proven techniques from non-automotive industries to overcome these challenges: the Business Process Model and Notation (BPMN) 2.0 is an established cross-industry standard to graph business processes [4]. The standard includes execution semantics, which allows the definition of directly executable process models that are processed with a so-called process engine. In Fig. 1, an interdisciplinary car testing lifecycle during manufacturing work is illustrated with the BPMN standard. Four roles on specification and scheduling, testing, vehicle connection, and shopfloor and car engineering interact in a time-directed workflow to perform interdisciplinary work w.r.t. to a car. The question arises of bridging the business-IT gap between such a business process model and the execution of car-related tasks. This approach can form the basis for flexible and intelligent manufacturing [5]. Executing process models in the planning level of the automation pyramid is typically achieved by enterprise architecture engines. There is the particular challenge of incorporating the car network, including HPCs, for runtime critical applications, i.e., the field level with process engines. Aside from the technical components, user acceptance of the technology is also essential.

The underlying research question of this work is as follows:

- How can executable business process models for automotive testing and diagnostic processes improve software-defined manufacturing?

In doing so, we also investigate the five principles for conceptual modeling, i.e., user scenario, modeling language, modeling tool, people, and utility of the model [6].

The main contribution of this paper is a BPMN-based process automation approach to bridge the business-IT gap between engineering and production in automotive manufacturing systems. Based on automotive expert knowledge, we identify criteria for software artifacts to execute test process models in interdisciplinary work in and around a software-defined car. We will give an architectural blueprint w.r.t. seamless execution of internal engineering processes for car testing and diagnostics in flexible manufacturing with BPMN. The approach is evaluated by an automotive focus group and is therefore intended as an inspiration for end-to-end ecosystems to achieve quick adaption times.

The remainder of this paper is structured as follows: in Sect. 2, the industrial scope is stated and technical requirements are derived. Related work is reviewed and evaluated w.r.t. the requirements in the executable BPMN environment (see Sect. 3). The solution approach is presented in Sect. 4 and the respective artifact characteristics are described in detail. The approach is evaluated with a focus group discussion in Sect. 5. The conclusion of this work is given in Sect. 6.

2 Automotive scope and requirements

We have a closer look at the automotive scope based on industrial case studies from the car manufacturing domain and derive requirements for the software artifact creation.

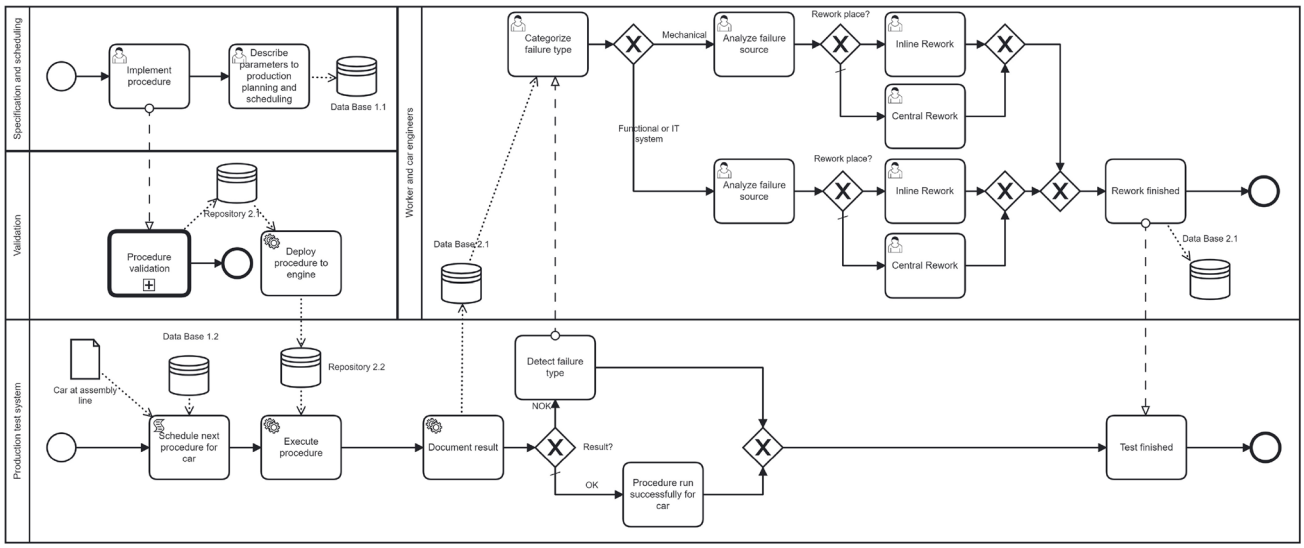


Fig. 1 Interdisciplinary workflow during car manufacturing represented in BPMN notation

The following case studies give a detailed view of the workflow from Fig. 1.

2.1 Case studies in automotive business process management

E/E car components are tested and commissioned in car production and, if necessary, flashed and coded with software. For this purpose, the underlying diagnostic tests are specified and programmed into a sequence of tasks (see the general setup of Fig. 2). This test specification is an interdisciplinary manual process for engineers from development, production, and after-sales that results in a textual specification document. This specification is, in general, valid for a car's lifetime, i.e., > 10 years, including repair workshops. A diagnostic test is a test to identify the cause of a fault, such as a test to see if the door lights illuminate when the door is opened.

The process is implemented in source code with variants for several car series. In the car development phase, these implementations are executed individually. The interaction of production staff is required for manual execution and rework. For a selected assembly line section in volume production, the relevant tests are planned and scheduled by computer scientists for the assembly line so that executing the tests on the car requires a minimum run time under predefined boundary conditions. For any given car, the set of relevant tests is then performed on an assembly line section with worker-car interaction. There are standardized interfaces for vehicle diagnostics, such as the SOVD API [2].

In the event of an error during execution, the manufacturing rework process is initiated. Therefore, the worker makes a computer-aided decision as to whether the failed test is due to a mechanical cause, such as a defective component, or whether there is a functional error, such as the incorrect coding of an ECU. A decision is made if the rework can be performed inline on the assembly line or if the car must be sent for central rework to better understand and reproduce the error. In the daily shop floor meeting, workshop employees and car engineers study shopfloor reports, e.g., on a car line with errors and test times from the previous production day.

An exemplary test is the control of the door exit light. There, an ECU session is established via diagnostics, the door exit light is checked, and the session is closed again. The case studies are shown in Fig. 2 with screenshots of the initially mentioned function test of the door exit light.

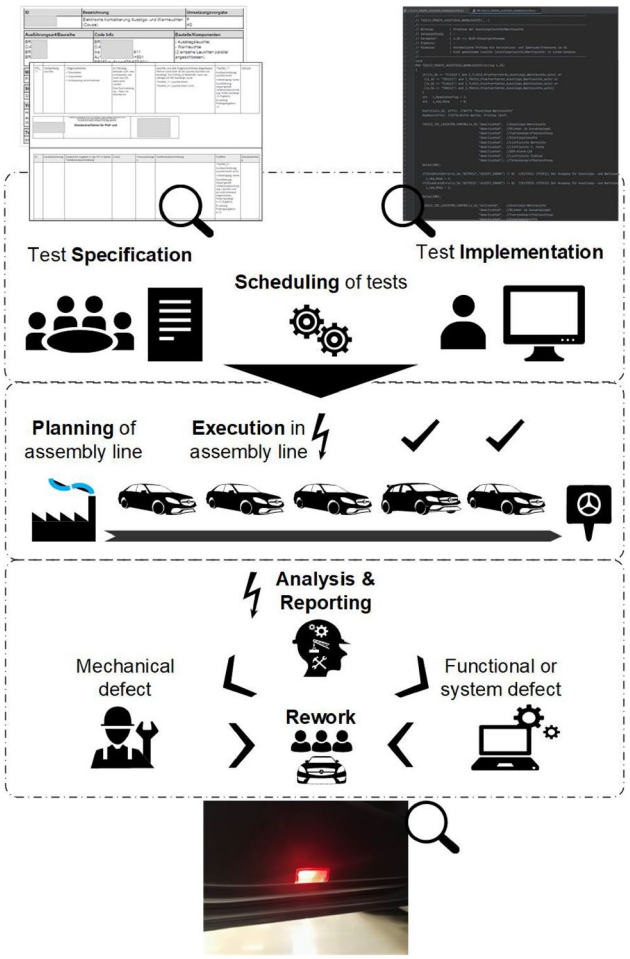


Fig. 2 Relevant automotive processes in this paper with screenshots of test specification, implementation and execution

2.2 Requirements

Group discussions in testing projects with automotive experts highlighted seven relevant requirements for developing the concept of executable process models in software-defined manufacturing. The requirements are also identified from further work by König et al. [7] and the technological and data related challenges listed in the work of Bickelhaupt et al. [8].

1. Executing business processes is no new approach. We demand the re-use of existing cross-industry methods, i.e., even in industries outside the automotive domain ($R_{\text{re-use}}$).
2. The developed approach should be integrated into large enterprise IT scopes with OEMs and their suppliers (R_{IT}). Testing specific applications as planning and scheduling should be supported, which means that algorithmic solutions must be integrated, too.
3. To reduce complexity in internal digital processes and respective IT systems, the test specification document should also serve as an executable artifact. The approach

should support the execution of test specifications such that the specification and implementation are no longer separated ($R_{seamless}$).
4. As the diagnostic web service calls in the car are supposed to be routed over SOVD for software-defined cars [2], the parameterization of the API should be directly integrated into the process specification (R_{API}). The approach should support API Management for automotive standards as SOVD and benefit from web technologies.
5. Programming knowledge should no longer be required for process specification and execution. The approach should not require programming knowledge for defining schedule descriptions and visual representations are preferred ($R_{\text{no-program}}$).
6. The concept should assist assembly line workers during rework ($R_{analysis}$). Therefore, a simple analysis of failed tests to find root causes should be supported.
7. As the considered work in the business process is along interdisciplinary teams along the product life cycle, new employees are involved over time. Thus, new employees should be able to learn the approach quickly, i.e., standards already known from other industries, universities, or online tutorials are preferable (R_{low}).

3 Related work

Based on the automotive case studies and the requirements presented, we assess our work into the related work w.r.t. business process management in software-defined manufacturing.

The nomenclatures "Software-defined Car" and "Software-defined Manufacturing" are predominantly driven by research projects, which are publicly-funded by the German Federal Ministry for Economic Affairs and Climate Action. For instance, there is a research project named SofDCar for Software-Defined Cars, as well as SDM4FZI for Software-Defined Manufacturing. The projects aim to improve vehicular and automation systems by making the functionalities of vehicular and manufacturing hardware definable through software. Car diagnosis can be implemented over-the-air for autonomous driving functions, whereas production systems are intended to be more reconfigurable. In the production environment, particularly Oechsle et al. deal with the independence of vendor-specific applications [9], however, the focus is on a general real-time capable architecture in manufacturing. The proposed deployment and orchestration workflow is not specific to a modeling language and lacks a concrete implementation.

In the next step, relevant modeling languages are discussed. On the one hand, in the context of the automotive industry and orchestrating test workflows, there exists the ISO standard OTX [10]. OTX is the standard to describe diagnostics and testing sequences for the automotive industry based on the UDS protocol [11]. As OTX is focused on the automotive onboard diagnostics setting with UDS, it does not allow including human tasks, and is less feature-rich than BPMN. This means especially, there is no task element that supports web services.

On the other hand, UML [12] and SysML [13] are languages to model the design and structure of software systems with various diagram types. They are not intended to model business processes and execute these with process engines.

Event-driven process chain (EPC) is a modeling language to graphically illustrate business processes with flowcharts [14]. It is not standardized or used worldwide and moreover, not indented to be executed through a process engine.

To summarize, in the context of manufacturing, Garcıa-Domınguez et al. provide a comparison with further modeling notations [15].

As we are interested in a domain-independent modeling language with standardized basis exchange format and execution semantics, we focus on BPMN, as already touched by König et al. [16]. In the following, we have a look at existing work on executing BPMN process models in manufacturing lines and on flexible orchestration of services.

Kocher et al. discuss modeling and executing production processes with capabilities and skills using ontologies and BPMN [17]. Modeling and executing IoT-enhanced business processes through BPMN and microservices with the use of ontologies for low-level real-world data is discussed by Valderez et al. [18]. We do not focus on ontologies and there is no focus on scheduling or hardware-related process execution.

Erasmus et al. discuss the application of business fragments and process models with BPMN for the specification of manufacturing operations [19]. The presented HORSE system discusses the execution of models as well, but there is no explicit focus on the manufacturing of cars.

Traganos et al. study end-to-end production process orchestration for smart printing factories [20]. The presented architecture focuses on a cyber-physical system with BPMN for modeling and executing purposes.

The general development of process execution support for high-tech manufacturing processes is discussed by Vanderfeesten et al. [21].

Roller and Engesser discuss the application of BPMN for automotive plant simulation [22].

Dubani et al. present a BPM-based framework for modeling and executing automotive business processes [23]. The framework combines BPMN and BPEL, which is no wide-applied industry standard anymore.

Hasic et al. study the execution of IoT Processes in BPMN 2.0 and identify remaining challenges [24].

Neubauer et al. [25] present subject-oriented business process management (S-BPM) with respect to high and low-level control over a common communication model.

A novel framework to automatically generate executable web services from BPMN models is presented by Zafar et al. [26]. The focus lies on Java web services.

Schäffer et al. present a process-driven approach within the engineering domain by combining BPMN with process engines [27]. The focus is on the product emergence process (PEP) that includes the subprocesses of product development, production planning, and production.

Stiehl et al. illustrate process-driven application with BPMN are illustrated and exploit the full functionality of BPMN 2.0 to model and execute cross-organizational differentiating processes [28].

In terms of flexible orchestration, a decision notation needs to be introduced. Funk creates a low-code business process execution platform with Python, BPMN, and Decision Model and Notation (DMN, OMG [29]) [30], which does not map to a flexible scheduling setting. Peinl and Perak study BPMN and DMN for easy customizing of manufacturing execution systems [31]. Nevertheless, the application of decision tables alone is not sufficient for the overall concept, as it is already highlighted by König et al. in terms of production workflows in manufacturing lines [7]. So we can use an optimization solver as discussed by König et al. [32].

To the best of our knowledge, there is no related work that studies the end-to-end application of executable business process models with BPMN in the context of car testing in software-defined manufacturing.

4 Approach on executable BPMN in software-defined car manufacturing

Based on the requirements and the identified research gap, we formulate an approach for executing automotive tests modeled with BPMN in software-defined manufacturing in the following. The approach includes an architectural proposal and refers to appropriate software artifacts to realize the system.

4.1 High and low-level test management with BPMN

The artifact overview is presented in Fig. 3. Following the automation diabolo model of Vogel-Heuser et al. [33] and the work of Neubauer et al. [25], the automotive architecture model for testing in software-defined manufacturing is divided into three layers. High-level control along automotive business processes is depicted in the upper rhombus

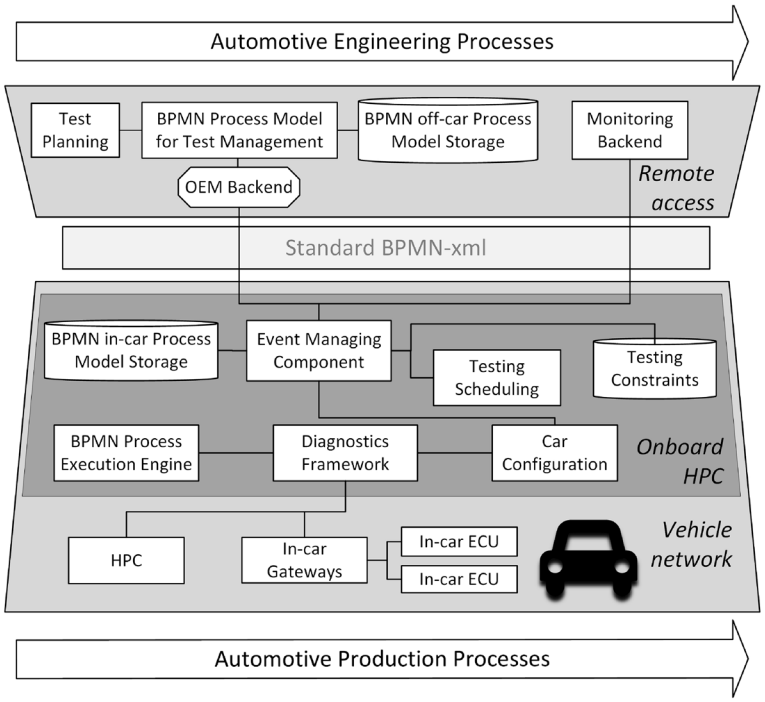


Fig. 3 Artefact overview w.r.t. the automation diabolo for improved test management (originally from Vogel-Heuser et al. [33])

and refers to an off-car architecture in the enterprise IT architecture (more details in Sect. 4.2). Low-level control along automotive production processes occurs in the lower gray rhombus in-car and refers to the in-vehicle architecture (details follow in Sect. 4.3). The familiar interface is the standard BPMN XML exchange format, illustrated as a box in the middle. The artifact overview builds the basis for an automotive workflow management system.

The respective transformation of the processes w.r.t. users and production strategy are displayed in matrix form in Fig. 4.

	Specification & Implementation	Scheduling	Execution of process models	Rework & Analysis documentation
Goal				
User	Engineer	Computer scientist	Software	Worker and business expert
Before	Text-based & source code	Manually	Runtime & worker-guided	Worker-guided
Now	BPMN	Orchestration of BPMN models	Process engine & worker-guided	Worker-guided & BPMN workflows

Fig. 4 Transformation matrix for consistent interdisciplinary process management

Engineers specify and implement a test for diagnostics or functionality checking by using the notation BPMN. The notation element *Service Task* can be used to interact with the car diagnostics API for web service calls. Worker interaction during execution is represented with a *User Task*. A decision based on *Data Objects* can be illustrated with *Exclusive Gateways*. Subprocesses can be represented as fragments in *Call Activities*.

Scheduling of tests is achieved by orchestrating BPMN models w.r.t. an automated optimization solution, e.g., as presented by König et al. [32].

Executing the diagnostic test with an automotive process engine takes a particular role. In this step, the concept leaves the horizontal level of engineering processes and carries out executable BPMN in diagnostic procedures in-car (see Fig. 3).

Rework and analysis documentation can be highlighted in the BPMN process, e.g., by a heatmap. In the following, we discuss details on high-level control in the upper rhombus of Fig. 3.

The conceptual model gives a detailed understanding of the highlighted components and relates to the central class

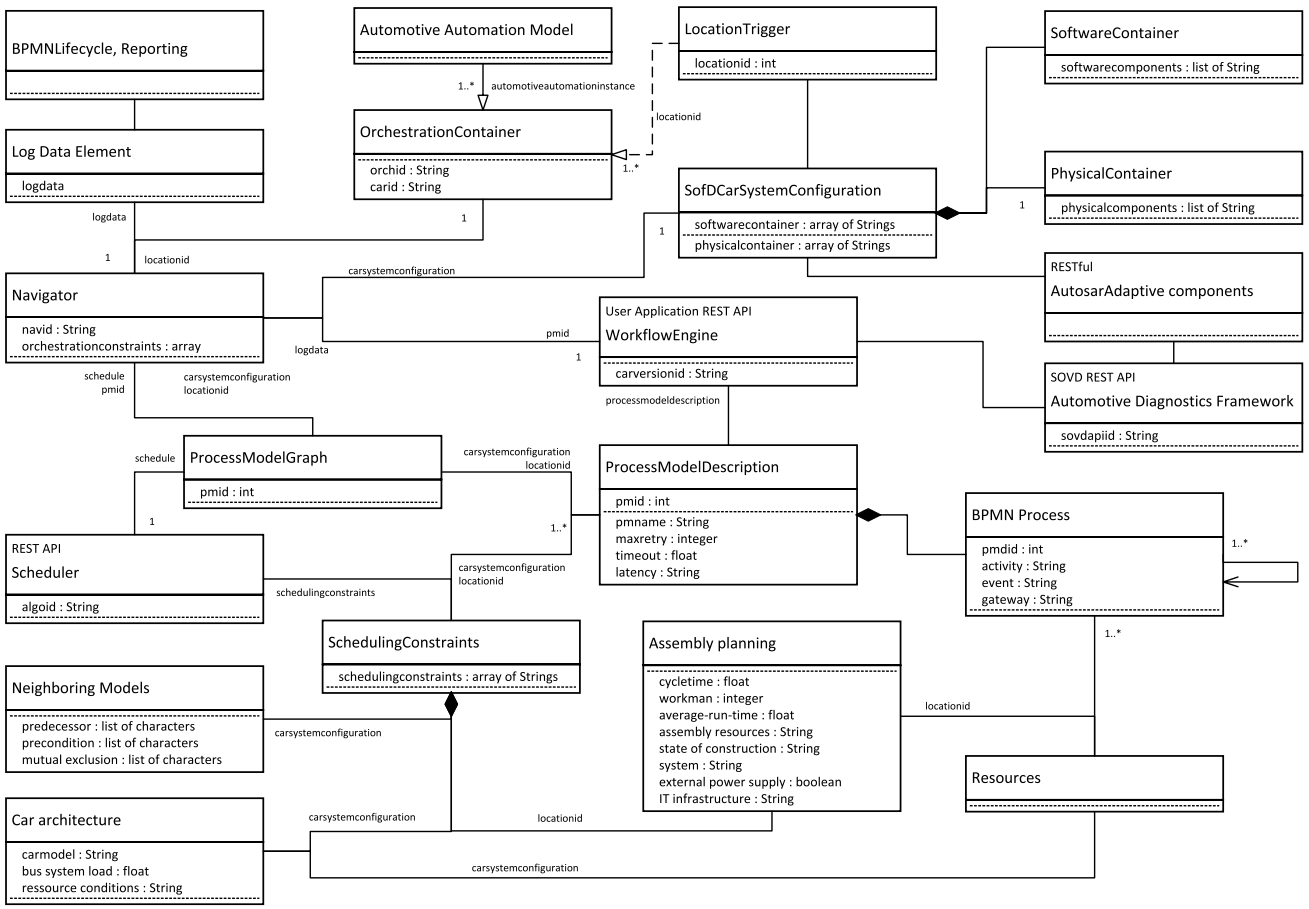


Fig. 5 Conceptual model of the overall approach

of the automotive automation model associated with a web service *OrchstrationContainer* (see Fig. 5).

The software system orchestration container is built upon the concept of a process model (PM) graph [34]. It builds the formal basis for BPMN process models. A *ProcessModelGraph* instance instantiates a software system for a specific *CarSystemConfiguration* at a test location w.r.t. to the *Assembly planning*. The car system configuration depends on the car's software and hardware (see Fig. 5).

4.2 Details on high level control in enterprise IT systems

The underlying use case is on modeling tests for production diagnostics. An evaluated guideline exists on how to model these workflows with BPMN published by König et al. [35]. The DMN standard is integrated to represent decision via a decision table. A test management process model is deployed via the OEM backend onto the vehicle onboard HPC and stored in the BPMN off-car Process Model Storage as standard BPMN serialized in XML format (see Fig. 3).

Moreover, constraints for test scheduling must be planned as proposed by König et al. [32].

After reviewing various commercial modeling tools, the authors decided to develop their modeling tool prototype because the modeling of the data flow with data objects as data inputs and outputs, as well as directed associations could not take place explicitly with just one tool. In addition, the integration of the interface to the vehicle was to be solved more flexibly between the modeling tool and engine. Thus, the modeling prototype should trigger the explicit modeling of data flows, the deployment, and the execution of process models in the car and support the integration of BPMN templates developed explicitly for the automotive sector. A screenshot of this software prototype is shown in Fig. 8.

Since the use of *Service tasks* is of particular interest in the application area of car diagnostics, e.g., to invoke diagnostics-related API methods of a car's SOVD API, the modeling tool allows to specify service calls against any HTTP API generically. Therefore, it enables employees to create and enrich required API requests based on modeled data objects and extract data from API responses through defined transformation expressions.

On the other hand, the user can also integrate existing SOVD OpenAPI templates. For this purpose, a particular BPMN template is offered for the automotive industry. Thus, the API methods specified within an OpenAPI document can be directly converted into a user-friendly BPMN template containing all the required information for calling the respective API method. These templates can then be offered within the palette of the modeling tool to enable employees to add them to their process models via drag and drop. In the Appendix, we propose a BPMN *Service Task* template. It is an excerpt from an evaluation study. It utilized a transformation for the API call. Data field assignments are also defined in the BPMN 2.0 standard but are not realized in the present modeling tool.

Monitoring and reporting of BPMN processes and their executions are available in customized tools. Since we do not have any specific requirements regarding their application in the context of test management, we do not focus on these aspects in detail.

4.3 Details on low-level control in vehicle networks

The BPMN process models have been deployed from remote access over the OEM backend to the vehicle network into the storage of the BPMN in-car process model. Now, the event managing component operates an instance of process models into a test schedule dependent on the respective test constraints for the car configuration. This belongs to the *OrchestrationContainer* of Fig. 5.

The BPMN process execution engine is integrated into the car network and communicates with the diagnostics framework during software-defined manufacturing. In contrast to existing industry applications, e.g., as studied by Rücker and Freund [36], there runs only one process instance at a time within one process engine in a car. In car production, APIs must meet strict millisecond latency requirements for tasks like window scaling or test bench communication, often for safety or certification reasons. Consequently, individual workflow engines are deployed in each car to execute diagnostic processes specific to that vehicle within the necessary latency limits. A low footprint in terms of memory and CPU consumption, as well as fast execution of processes. As figured out in an automotive runtime comparison framework [16], the process engine is written in C++. This choice is underlined by the work of AUTOSAR GbR [1]. This paper does not take a deeper look into the process engine as details regarding a car-tailored process engine are presented in König et al. [16].

To enable flexible orchestration of test processes and their constraints to each other in the car, the method of adaptive rescheduling is used. This allows for a new schedule to be recalculated even during runtime if a more optimal solution should be available due to time shifts of past tests. Results of a respective quantitative experiment are presented in König et al. [37].

The illustrated business-car-related system architecture of Fig. 3 supports the interdisciplinary work during the engineering phase of software-defined cars by multi-disciplinary teams composed of engineers, IT architects, programmers, process experts, and data analysts.

5 Evaluation and discussion

The approach presented in Sect. 4 is evaluated in two steps: a BPMN modeling tool and in-car execution engine can prove the end-to-end integration for tests. In the following, we call this an integration test. The overall approach to executable BPMN is discussed in a focus group with OEM experts.

5.1 OEM evaluation setting

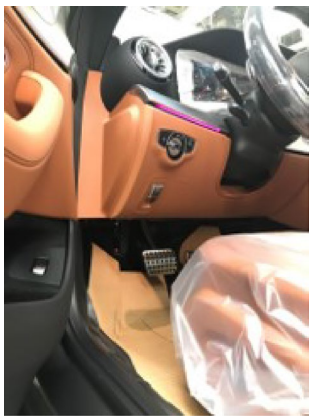
The system used for experimental evaluation is part of an automotive OEM-internal development setting for production technology. An employee specifies and develops tests on a personal notebook and further executes the tests via a car tester over the onboard diagnostics interface, the prototypical evaluation setting was intended to be similar.

The automotive BPMN modeling tool runs on a personal notebook. The BPMN Process Models are deployed onto a car diagnostics tester that includes an assembler of an automotive HPC.

The car tester dongle is a HPC emulator. In Fig. 6a, the device is illustrated itself, whereas, in Fig. 6b, the device is plugged into the OBD socket. The starting test setting is illustrated in Fig. 6c. After the successful integration and setup of all components, the scenario was also successfully applied and tested in an actual assembly line production context.

The BPMN Process Execution Engine is outlined in the work of König et al. [16], and the Orchestration Service is validated by König et al. [37]; both are containerized as Docker containers on the HPC emulator. The in-car components are accessed via REST APIs with HTTP clients. So they can be applied for the ASAM standard SOVD.

(a) Dongle with integrated HPC emulator



(b) Dongle plugged in OBD car connector

(c) Test bench

Fig. 6 The OEM evaluation set up

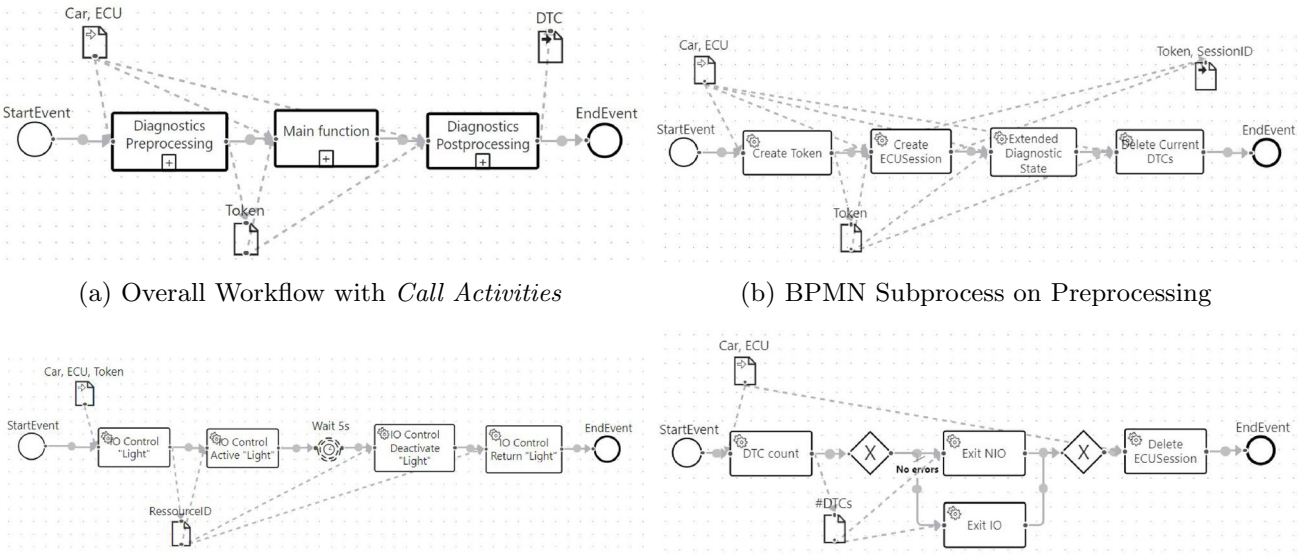
5.2 Integration test for BPMN modeling tool and car BPMN engine

Using the OEM evaluation setting, we perform a so-called integration test to show the implementation and execution of a BPMN process model in the car with the provided software tool components.

To do this, the function test for the car door light was chosen as it checks basic diagnostic routines and includes different hierarchy levels within the overall workflow.

In Fig. 7a, the test is illustrated as a workflow of *Call Activities* for three subprocesses, which are further described in the Fig. 7b–d. The subprocesses receive data inputs *Car* and *ECU*, which are parameters for the API calls so that the test can be parameterized and executed for several cars and door light ECU variants. Within the first subprocess in Fig. 7b, a diagnostic security token is generated via a *Service Task* API call. The token is saved as a data object for further *Service Tasks* within the subprocess and provided as data output to the following subprocesses in Fig. 7c and 7d. Further automotive-specific steps are performed. In Fig. 7c, the light is activated within the time duration of 5*s*. If the door exit light works, the test produces the visual output illustrated in Fig. 2. In the subprocess on postprocessing in Fig. 7d, the ECU error memory is checked, and finally, the process is terminated.

The realization for two car variants showed that the integration of BPMN in high and low-level test management is fulfilled.



(a) Overall Workflow with *Call Activities*

(b) BPMN Subprocess on Preprocessing

(c) BPMN Subprocess on the Main Routine

(d) BPMN Subprocess on Postprocessing

Fig. 7 BPMN models of the door exit light functionality for automotive modeling tool and engine integration test—screenshots are taken from the modeling tool that includes pixels in the canvas

5.3 Focus group validation

A focus group validation has been conducted to evaluate the approach to executable BPMN with the involved user group.

Focus group members The focus group consisted of eight employees of an OEM, including one employee with an expert background in production planning, three employees involved in specification and implementation of testing and diagnostics, two employees engaged in test schedule programming, one employee working on failure analysis during shopfloor, and one employee being a production test system developer. One of the eight experts had experience less than a year in his current job. Two of the eight experts were women. The moderator of the focus group was one of the authors of this paper.

The focus group participants were already involved in the product development of the automotive BPMN modeling tool during two workshops of totally 3.5*h* to create a user-friendly artifact. The focus group tasks were communicated online to the participants, who had three weeks to complete the tasks and provide a questionnaire of the approach in the tool *Mural*. Afterwards, a hybrid 1.5*h* workshop with discussion has been conducted.

The engineering processes of test sequence specification and programming, schedule generation, execution in the vehicle, and the idea of an end-to-end ecosystem were evaluated concerning the parameters: *Applicability of the BPMN-based approach*, *Expressiveness of BPMN models for the automotive testing setting*, and *Feasibility of the modeling and executing setup*. The rating could be made in five levels: *very good*, *good*, *neutral*, *poor*, and *very poor*. In addition, three free-text columns were added to the questionnaire: *Name advantages of the solution*, *Name challenges of the solution and, if applicable, suggestions for addressing them (w.r.t. tooling, concept...)*, and *Comments*. The full questionnaire is shown in the Appendix.

Modeling tasks The first modeling task was to understand a software module test to control the fan in the car seat (see Fig. 8). Three functional changes were then to be made in functional parameters. In addition, a subprocess invocation had to be modeled via a *Call Activity*.

In Fig. 9, further process models are illustrated. The employees were presented with one example each for the integration of BPMN *script tasks* (see Fig. 9a) and a representation of *error* (see Fig. 9b) handling in the context of test bench communication.

In Fig. 9a, it is shown how a user can integrate an external source code script via the BPMN element *Script Task*. The authors have included this process model in the evaluation because there were concerns in preliminary discussions with focus group participants that, e.g., measurement evaluations could be mapped within BPMN.

In Fig. 9b, an example of error handling events in a process model is shown, which executes a test routine as a

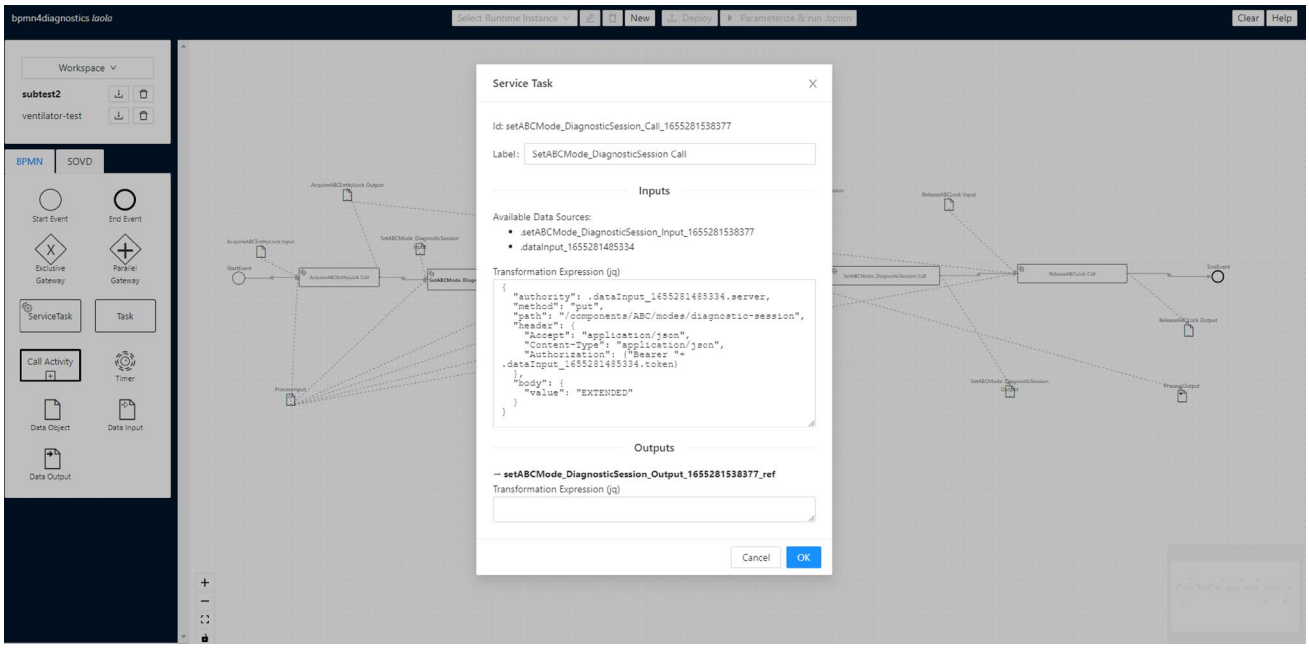


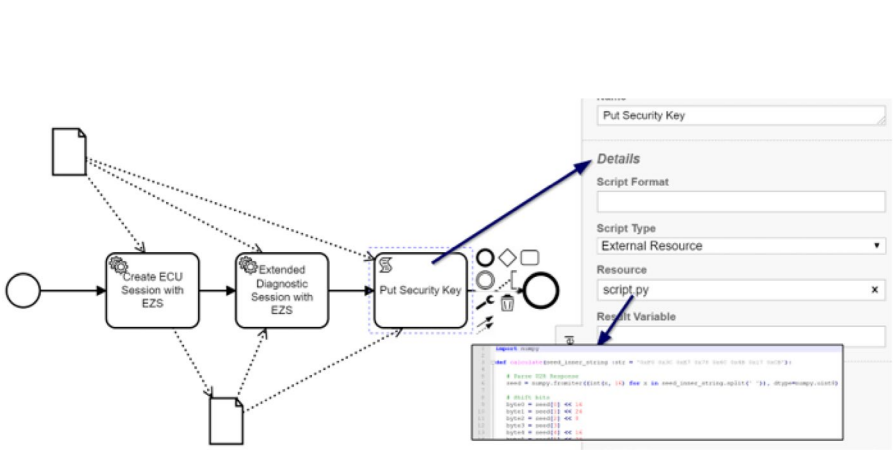
Fig. 8 Illustration of BPMN data handling and processing during a focus group evaluation task

subprocess via a *Call Activity* and contains two error cases: When the door is opened, a test stand goes into idle mode. Otherwise, the test is terminated.

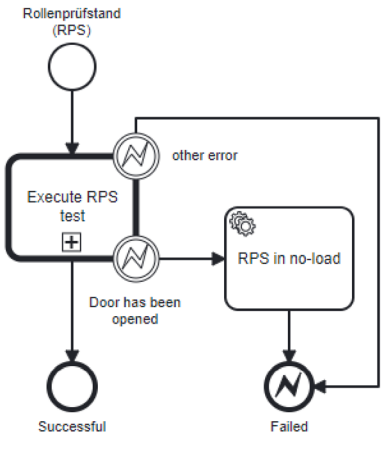
Results and discussion In the subsequent discussion meeting during the on-site workshop, the results were discussed with the participants. The aggregated results of the questionnaire are presented in Fig. 10. At first glance, most results range from neutral to good.

The evaluation result regarding the *applicability* of the concept from the perspective of the focus group participants is shown in Fig. 10a. One participant found the applicability to be very good in terms of test specification and programming, six participants found it good, and one participant was undecided in the scores very good and good (0.5 points each).

The advantages of the solution are that no programming is required and that functions can be reused in the company through templates. Moreover, the modeling approach makes it possible to track an error and its propagation through the process model graphically. A graphical overview is actually always advantageous, the participants said. In summary, simple test sequences can be mapped well. Challenges in the current tooling were cited as the naming being incomprehensible, which the authors justified by the fact that the



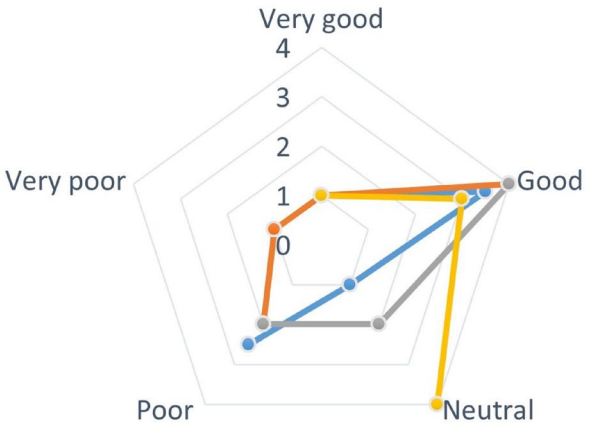
(a) BPMN Model with integrated *Script Task*



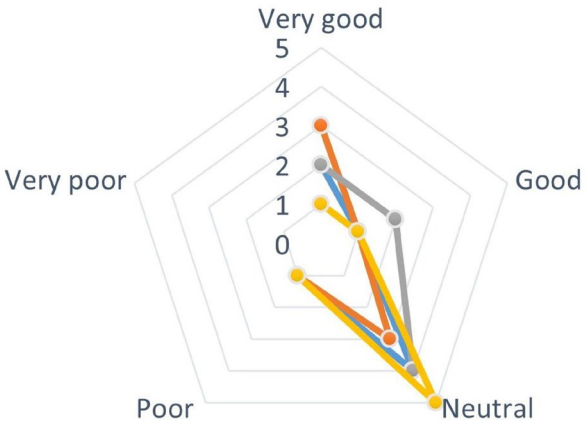
(b) Exemplary Failure Handling

Fig. 9 Interpretation tasks during focus group evaluation

(a) Scores on Applicability of Executable BPMN



(b) Scores on Expressiveness of BPMN Models



(c) Scores on Feasibility of Automation

Fig. 10 Cumulative ratings among 8 experts as part of the focus group—line color blue for area of test sequence, orange for test schedule, grey for in-car execution and yellow for extrapolation to OEM-wide testing workflows

modeling tool is a software prototype. One employee noted that in the case of complex examples, he considered source code programming to be clearer. Another coworker saw in addition the construct of the *Script Tasks* as suitable, if complex contents can be merged in external scripts.

For schedule programming, the concept is rated as very good by two participants, good by five participants and one participant rated it as poor.

The advantage of the solution is that no programming is required and the graphical representation provides a good overview of the processes. Nevertheless, in case of flexible manufacturing, an automated concept, as proposed in the paper of König et al. [37], is preferable. No concrete challenges were named. Critiques were that there is much initial effort since existing specifications and program code would have to be modeled from scratch in BPMN 2.0. The current prototype lacks visualization for error detection. In this context, the current prototype of the modeling tool was discussed, which should be improved in terms of user experience, e.g., labels in the modeling tool often need to be clarified for the users. The participants can think of assistance functions that flow creation is partly automated.

In the in-vehicle execution, two participants rate the approach as very good, three as good and three as neutral.

It was noted that only those defects can be visualized graphically that are also related to the modeled processes. A higher level of detail would still have to be analyzed. According to the employees, the concept is a milestone away from the current process. Great potential is seen in the execution of safeguards, not only on the car, but also, e.g., in software-in-the-loop, hardware-in-the-loop, or functional mock-ups.

Regarding an end-to-end ecosystem, two participants rate the concept as good, five are undecided and one participant rates it as poor.

It is seen as a necessary foundation that all relevant stakeholders, including suppliers, need the same modeling tool and execution engine, e.g., as further development of the ASAM standards.

In the next part, the evaluation regarding the *expressiveness* of BPMN models were evaluated. The results are presented in Fig. 10b.

One participant rates the expressiveness of the models for test specification and programming as very good, three participants evaluate it as good, one participant is neutral and two participants evaluate it as poor. One participant finds good and poor portions, but does not want to rate neutral, resulting in 0.5 points each for good and poor.

The possibility of creating functionalities quickly by modeling is seen as an advantage. The parameterization of REST calls in *Service Tasks* must be made easier for users. The authors have countered that the modeling tool is a prototype that is intended to map the basic functionalities and

does not represent a finished end product. One challenge is the confusion caused by long strings in the parameterization. Basically, it would be necessary to work with sub-routines in order to maintain an overview. Here, more visual concepts are needed on how to understand a complex process without clicking back and forth too much.

For the test schedule generation, one participant sees the significance as very good, four participants as good, two participants as poor and one participant as very poor.

Basically the same benefits and challenges were mentioned as for test creation.

Regarding the in-car execution, the participants rate the informative value with 50% as good, 25% neutral and 25% as poor.

It was noted that for engineers, testing on the car involves working with a laptop. Consequently, a smaller monitor is available as when working on a desk, where more complex models can be clearly displayed in the modeling tool.

For end-to-end ecosystem design along car development, production and after sales, one participant rates very good, three participants rate good, and four participants are neutral.

In the context of interdisciplinary collaboration, the concept is considered to be quickly understood. If modifications are expected in the program, expert knowledge is required.

The third and last part of the evaluation concerns the *feasibility* with the goal of automating executable process models with BPMN (see Fig. 10c). Two participants rate the feasibility in the environment of test specification and programming as very good, one participant as good, four participants as neutral and one participant as poor.

In principle, the feasibility is good to neutral. In the case of special cases with many parameters and sub-menus, the challenges lie in maintaining an overview.

Regarding test schedule generation, three participants give the rating very good, one participant good, three participants neutral and one participant poor.

To achieve a well-structured flow, the test sequences must be well modularized. If all BPMN components are allowed at the flow level, automatic optimization can be difficult to implement. Reference is made by stakeholders to the concept of König et al. [32].

Regarding the feasibility for automation in the vehicle, two participants give the rating very good, two participants give the rating good and four participants are neutral.

Regarding the feasibility of executable BPMN in the ecosystem, one participant gives the rating very good, one participant good, five participants are neutral in their rating and one participant gives the rating poor.

It was noted that all stakeholders from development, production and after sales need the same processes and tools. If all parties involved the new standard, the greatest added value is created.

In the further discussion, it was noted that templates for standard processes with nested structure with 10–15 elements are practical. There should be the possibility to group BPMN elements by color. Staff would also like to be able to store comments. The authors noted that this is possible in the BPMN standard, but is not part of the modeling prototype. A debugging concept with breakpoints is desired, as is a simulation of test cases. A connection to repositories like GitHub is desired to observe development statuses. A concept for user training is demanded.

5.4 Assessment of the requirements

We presented and discussed the findings of the integration test and the focus group evaluation w.r.t. the approach on executable BPMN with flexible in-car orchestration in software-defined manufacturing in Sect. 5.3. The assessment of the requirements from Sect. 2.2 gives more details on the slightly positive evaluation.

Requirement $R_{\text{re-use}}$ is fulfilled as the integration test's result shows and it is a plus for the focus group participants that the notation is not only automotive-specific. In a next step, templates for execution should be defined. The integration test showed that the general approach set up works and thus, requirement R_{IT} is fulfilled. It would be a plus, if car development and after sales domains applied the approach as well. Via the integration test and the results of the focus group,the requirement $R_{seamless}$ is fulfilled in the automotive manufacturing domain. The seamless integration was the most significant point for the focus group participants. Requirement R_{API} is partly fulfilled as there is a need for better user experience as the focus group evaluation showed. There was a lack in user experience and given templates. As a result, the participants had problems to quickly understand the appraoch. Moreover, requirement $R_{\text{no-program}}$ is fulfilled. This was a main requirement for the employees as production processes get more complicated because of the integration of more and more software artifact into the business processes. Especially workers in automotive shopfloor highlight the benefit of visual analysis, so requirement $R_{analysis}$ is fulfilled. This helps the workers to easily understand what the current problems are, e.g., during shopfloor meetings. Engineers confirm that the ramp-up time for new employees can be shortened by the cross-industry approach

on executable BPMN. Nevertheless, quantitative metrics on time reduction are missing and should be studied. So, R_{low} is partly fulfilled.

Regarding the research question, executable business process models with BPMN build a solid basis for automotive testing and diagnostic processes in software-defined manufacturing.

Transferability The approach is applicable for run time critical application scenarios. This means the non-automotive core components of the approach cannot only be integrated into vehicle networks, but also, e.g., on programmable logic controllers to execute testing processes seamlessly. The transfer to areas outside the automotive test and diagnostic processes was not examined in this paper.

6 Conclusion and outlook

Lean process automation for software-defined car manufacturing is the basis for a digital ecosystem. The approach of this paper implies the application of business process management with execution semantic approaches to bridge operational and information technology for automotive testing processes for software-defined cars in manufacturing. As one result continuous engineering processes are enabled without vendor-specific software frameworks. The cross-industry BPMN standard plays a crucial role, as business processes can be modeled graphically and executed in enterprise IT systems and cars with an in-car process engine.

In this work, the contribution of executing automotive test workflows with BPMN and web service orchestration in interdisciplinary software-defined car manufacturing is highlighted through an evaluated approach at an OEM. The presented system allows for seamless modeling and execution of manufacturing processes in the enterprise architecture and the car. It includes flexible orchestration of web services to manage car variants and interact with modern automotive diagnosis interfaces.

The approach is validated with respective software artifacts and a software-defined car-integrated case study. A focus group evaluation with automotive experts highlights the benefits of seamless internal processes and shows ideas for further research. User experience design for the modeling tool is essential when employees in interdisciplinary teams collaborate in complex systems. With respect to the software-defined car manufacturing lines, a production-specific web service interface should be developed to further integrate information technologies into car architectures.

Moreover, the concept should be evaluated across all diagnosis domains, i.e., car development, production and after sales. A study across OEMs, e.g., as part of a future research project, could show more benefits on standardization of the software-enabling technology that was presented in this paper.

Appendix

Listing 1: XML Representation of a BPMN Service Task Element with SOVD Endpoints

```
<bpmn:serviceTask id="
      setABCMode_DiagnosticSession_Call" name="
      SetABCMode_DiagnosticSession Call"
      implementation="https://github.com/cars/engine#
      rest">
  <bpmn:ioSpecification>
    <bpmn:dataInput id="
          setABCMode_DiagnosticSession_Call_data_input
          " />
    <bpmn:dataOutput id="
          setABCMode_DiagnosticSession_Call_data_output
          " />
    <bpmn:inputSet>
      <bpmn:dataInputRefs>
          setABCMode_DiagnosticSession_Call_data_input
          </bpmn:dataInputRefs>
    </bpmn:inputSet>
    <bpmn:outputSet>
      <bpmn:dataOutputRefs>
          setABCMode_DiagnosticSession_Call_data_output
          </bpmn:dataOutputRefs>
    </bpmn:outputSet>
  </bpmn:ioSpecification>
  <bpmn:dataInputAssociation id="
      setABCMode_DiagnosticSession_Call_dia">
    <bpmn:sourceRef>
          setABCMode_DiagnosticSession_Input_ref</
          bpmn:sourceRef>
    <bpmn:sourceRef>dataInput</bpmn:sourceRef>
    <bpmn:targetRef>
          setABCMode_DiagnosticSession_Call_data_input
          </bpmn:targetRef>
    <bpmn:transformation language="https://stedolan.
          github.io/jq">{
        "authority": .dataInput.server,
        "method": "put",
        "path": "/components/ABC/modes/diagnostic-
            session",
        "header": {
          "Accept": "application/json",
          "Content-Type": "application/json",
          "Authorization": ("Bearer "+ .dataInput.
              token)
        },
        "body": {
          "value": "EXTENDED"
        }
    }</bpmn:transformation>
  </bpmn:dataInputAssociation>
  <bpmn:dataOutputAssociation/>
</bpmn:serviceTask>
<bpmn:dataObject id="
      setABCMode_DiagnosticSession_Input"/>
<bpmn:dataObjectReference id="
      setABCMode_DiagnosticSession_Input_ref" name="
      SetABCMode_DiagnosticSession Input"
      dataObjectRef="
      setABCMode_DiagnosticSession_Input" />
<bpmn:dataObject id="
      setABCMode_DiagnosticSession_Output" />
<bpmn:dataObjectReference id="
      setABCMode_DiagnosticSession_Output_ref" name="
      SetABCMode_DiagnosticSession Output"
      dataObjectRef="
      setABCMode_DiagnosticSession_Output" />
```

See Table 1.

Table 1 Questionnaire form as basis for the focus group discussion

No.	Question	Rating					Feedback	
		Very good	Good	Neutral	Poor	Very Poor	Advantages	Challenges (tooling, concept...)
1	How do you rate the applicability based on the test sequence creation?							
2	How do you rate the applicability based on the test schedule creation?							
3	How do you rate the applicability based on in-car execution of BPMN process models?							
4	How do you rate the applicability on the basis of a consistent OEM test and diagnostics world in software-driven cars, e.g., also integration tests with suppliers?							
5	How do you rate the expressiveness based on the test sequence creation?							
6	How do you rate the expressiveness based on the test schedule creation?							
7	How do you rate the expressiveness based on in-car execution of BPMN process models?							
8	How do you rate the expressiveness on the basis of a consistent OEM test and diagnostics world in software-driven cars, e.g., also integration tests with suppliers							
9	How do you rate the feasibility of automation on the basis of test sequence creation?							
10	How do you rate the feasibility of automation on the basis of test schedule creation?							
11	How do you rate the feasibility based on in-car execution of BPMN process models?							
12	How do you rate the feasibility on the basis of a consistent OEM test and diagnostics world in software-driven cars, e.g., also integration tests with suppliers?							

Acknowledgements This publication is based on the research project SofDCar (19S21002), which is funded by the German Federal Ministry for Economic Affairs and Climate Action.

Author Contributions All authors contributed to the study conception and design. Material preparation, data collection and analysis were performed by Simone König. The first draft of the manuscript was written by Simone König and Michael Hahn. All authors read and approved the manuscript.

Funding Open Access funding enabled and organized by Projekt DEAL.

Data Availability Not applicable.

Code availability Not applicable.

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

Ethics approval The research activity envisaged in this work has been conducted applying fundamental ethical principles. We confirm that all the authors involved in the writing of this article are aware of this work and approve all its contents. In addition, the paper has not been published or submitted to any other journal.

References

1. AUTOSAR GbR. AUTOSAR Adaptive Platform. https://www.autosar.org/standards/adaptive-platform
2. ASAM e V. Service-Oriented Vehicle Diagnostics. https://www.asam.net/standards/detail/sovd/

3. Behrendt S, Martin M, Puchta A, Ströbel R, Fisel J, May MC et al (2023) Software-defined manufacturing for the entire life cycle at different levels of production. In: Kiefl N, Wulle F, Ackermann C, Holder D (eds) Advances in automotive production technology—towards software-defined manufacturing and resilient supply chains. Springer International Publishing, Cham, pp 25–34
4. OMG. Business Process Model and Notation (BPMN), Version 2.0. http://www.omg.org/spec/BPMN/2.0
5. Wieland M, Nicklas D, Leymann F (2008) Managing technical processes using smart workflows. In: Towards a service-based internet. Springer, Berlin, pp 287–298
6. Sinz EJ (2021) Five questions to be clarified before starting to model conceptually. In: Enterprise modelling and information systems architectures (EMISAJ)—international journal of conceptual modeling, vol 16, no 3. https://doi.org/10.18417/emisa.16.3
7. König S, Vogel-Heuser B, Fieg E, Hahn M, Kopp O (2021) Modelling production workflows in automotive manufacturing. In: 2021 IEEE 23rd conference on business informatics (CBI), vol 02, pp 39–46
8. Bickelhaupt S, Hahn M, Nuding N, Morozov A, Weyrich M (2023) Challenges and opportunities of future vehicle diagnostics in software-defined vehicles. In: SAE Technical Paper 2023-01-0847
9. Oechsle S, Walker M, Fischer M, Frick F, Lechler A, Verl A (2023) Real-time capable architecture for software-defined manufacturing. In: Kiefl N, Wulle F, Ackermann C, Holder D (eds) Advances in automotive production technology—towards software-defined manufacturing and resilient supply chains. Springer International Publishing, Cham, pp 3–13
10. ISO (2011) Road vehicles—Open Test sequence eXchange format (OTX). ISO 13209 ed. Vernier. International Organization for Standardization, Geneva. https://www.iso.org/standard/53507.html
11. International Organization for Standardization (ISO). ISO 14229-1:2020 road vehicles—unified diagnostic services (UDS)—part 1: application layer. https://www.iso.org/standard/72439.html
12. OMG. OMG Unified Modeling Language (OMG UML), Superstructure, Version 2.4.1. http://www.omg.org/spec/UML/2.4.1
13. OMG. OMG Systems Modeling Language (OMG SysML), Version 1.3. http://www.omg.org/spec/SysML/1.3/
14. Riehle DM, Jannaber S, Karhof A, Delfmann P, Thomas O, Becker J (2016) Towards an EPC standardization—a literature review on exchange formats for EPC models. In: Multikonferenz Wirtschaftsinformatik, MKWI 2016
15. García-Domínguez A, Marcos-Bárcena M, Medina I (2012) A comparison of BPMN 2.0 with other notations for manufacturing processes. Key Eng Mater 04(502):593–600. https://doi.org/10.1063/1.4707613
16. König S, Vogel-Heuser B, Wilch J, Unger T, Hahn M, Soldo S et al (2023) BPMN4CARS: a car-tailored workflow engine. In: 2023 IEEE 21st international conference on industrial informatics (INDIN), pp 1–6
17. Kocher A, Silva LMVD, Fay A (2022) Modeling and executing production processes with capabilities and skills using ontologies and BPMN. In: 2022 IEEE 27th international conference on emerging technologies and factory automation (ETFA). IEEE. https://doi.org/10.1109/ETFA52439.2022.9921564
18. Valderas P, Torres V, Serral E (2022) Modelling and executing IoT-enhanced business processes through BPMN and microservices. J Syst Softw 184:111139. https://doi.org/10.1016/j.jss.2021.111139
19. Erasmus J, Vanderfeesten I, Traganos K, Grefen P (2020) Using business process models for the specification of manufacturing operations. Comput Ind 123:103297. https://doi.org/10.1016/j.compind.2020.103297
20. Traganos K, Vanderfeesten I, Grefen P, Erasmus J, Gerrits T, Verhofstad W (2020) End-to-end production process orchestration for smart printing factories: an application in industry. In: 2020 IEEE 24th international enterprise distributed object computing conference (EDOC), pp 155–164
21. Vanderfeesten I, Erasmus J, Traganos K, Bouklis P, Garbi A, Boultadakis G et al (2019) Developing process execution support for high-tech manufacturing processes. Springer International Publishing, Cham, pp 113–142
22. Roller D, Engesser E (2014) BPMN process design for complex product development and production. In: Plödereder E, Grunske L, Schneider E, Ull D (eds) Informatik 2014. Gesellschaft für Informatik e.V, Bonn, pp 1979–1984
23. Dubani Z, Soh B, Novel Seeling C (2010) A novel design framework for business process modelling in automotive industry. In: 5th IEEE international symposium on electronic design. test & applications, pp 250–255
24. Hasić F, Asensio ES (2019) Executing IoT processes in BPMN 2.0: current support and remaining challenges. In: 2019 13th international conference on research challenges in information science (RCIS), pp 1–6
25. Neubauer M, Stary C, Kannengiesser U, Heininger R, Totter A, Bonaldi D (2017) S-BPM's industrial capabilities. Springer, Cham, pp 27–67
26. Zafar I, Azam F, Anwar MW, Maqbool B, Butt WH, Nazir A (2019) A novel framework to automatically generate executable web services from BPMN models. IEEE Access 7:93653–93677. https://doi.org/10.1109/ACCESS.2019.2927785
27. Schäffer E, Stiehl V, Schwab PK, Mayr A, Lierhammer J, Franke J (2021) Process-driven approach within the engineering domain by combining business process model and notation (BPMN) with process engines. Proc CIRP 96:207–212. https://doi.org/10.1016/j.procir.2021.01.076
28. Stiehl V, Raw R, Smith P (2014) Process-driven applications with BPMN. Springer International Publishing, Cham
29. OMG. Decision Model And Notation (DMN), Version 1.3
30. Funk D (2023) Creating a low-code business process execution platform with Python, BPMN, and DMN. IEEE Softw 40(1):9–17. https://doi.org/10.1109/MS.2022.3212033
31. Peinl R, Perak O (2019) BPMN and DMN for easy customizing of manufacturing execution systems. In: Di Francescomarino C, Dijkman R, Zdun U (eds) Business process management workshops. Springer International Publishing, Cham, pp 441–452
32. König S, Reihn M, Abujamra FG, Novy A, Vogel-Heuser B (2021) Flexible scheduling of diagnostic tests in automotive manufacturing. Springer Flexible Services and Manufacturing Journal
33. Vogel-Heuser B, Kegel G, Bender K, Wucherer K (2009) Global information architecture for industrial automation. Automatisierungstechnische Praxis (atp) 01(51):108–115. https://doi.org/10.17560/atp.v51i01-02.1948
34. Leymann F, Roller D (1999) Production workflow: concepts and techniques. Prentice Hall PTR, Hoboken
35. König S, Vogel-Heuser B, Fieg E, Hahn M, Kopp O (2021) Modelling production workflows in automotive manufacturing. In: 2021 IEEE 23rd conference on business informatics (CBI), vol 02, pp 39–46
36. Rücker B, Freund J (2019) Real-life BPMN, 4th edn. Independently Published
37. König S, Vogel-Heuser B, Karg F, Land K, Carbone E, Hradecky A et al (2023) Online test scheduling in car production lines. In: 2023 IEEE intelligent vehicles symposium (IV), pp 1–8