Developing Context-Aware Applications in Automotive Environments

Julian Dörndorfer[a], Markus Schmidtner[b] and Holger Timinger[c]
Institute for Data and Process Science, University of Applied Sciences Landshut, Am Lurzenhof 1, Landshut, Germany

Keywords: Agile Manufacturing, Agile Processes, Context-Aware Applications, Context Modeling, Product Development Process, Complexity Management.

Abstract: IT companies like Google and Apple are entering the automotive market with disruptive technologies such as autonomous driving, enabled by numerous sensors evaluated by software. These sensors allow vehicles to recognize their context and passengers and react accordingly by adjusting features like interior temperature based on solar radiation and temperature. The domain-specific modeling language (DSML) SenSoMod incorporates these sensors early in software architecture design. Established automakers face challenges integrating more software components in the product development process (PEP). Optimizing the supply chain has led to a lack of common understanding of requirements and development. This multilayered process must now meet additional requirements for complex software development. SenSoMod can help by uniformly considering sensors across the supply chain, aiding in the implementation of context-aware software in vehicles. In this paper, experts from the automotive industry have assessed SenSoMod by its potential benefits and challenges in the automotive development process.

1 INTRODUCTION

At this point in time, technological changes that enable new applications are bringing the automotive industry to a crossroad and challenging it to fundamentally rethink its products (Winkelhake, 2021). The current combustion-based engines are on the decline and, in some cases, even prohibited for new cars by certain governments within the next few years (Stan, 2022). This, in turn, forces the automotive industry to develop new and clean energy engines (Biesinger et al., 2018; Biesinger et al., 2019). New players in the market, like Tesla, have specifically developed their models with these new engines in mind and now have a competitive advantage in this area (Kraft et al., 2021). Additionally, major IT companies like Google and Apple have plans to branch out into the automotive sector (Kirk, 2015). These fresh competitors plan to use new and disruptive IT technologies, like autonomous driving, to gain market share. In addition, today's cars have many sensors, and most functions are controlled by software. These sensors also make it possible to offer new features depending on the context of the vehicle or passengers. For example, while driving, the next gas or charging station can be displayed depending on the fuel/charging level, or the temperature in the vehicle can be adjusted depending on the measured body temperature. This all leads to context-dependent software development for cars, making the production of a car more and more complex (Vdovic et al., 2019). The IT technology leaders have the development experience to realize these new software systems in a comparably short time and put pressure on established manufacturers (Winkelhake, 2021). While the new competitors have the resources to close gaps in manufacturing with time, the established manufacturers face a much bigger hurdle. The established product development process (PDP), for a long time perfected and in the past one of the strengths of the manufacturers, is not very agile. Over the last two decades, the PDP has been outsourced to different partners along the supply chain and, therefore, involves a huge number of different development teams to reduce lead times and meet shorter life cycles (Morris et al., 2018). This distribution along the supply chain can lead to difficulties, such as a lack of product and domain knowledge (Broy, 2006; Prenner et al., 2021), challenges in requirements engineering (Prenner et al., 2021; Fabbrini et al., 2008), and isolated, uncoordinated development of components at suppliers and OEMs (Fabbrini et al., 2008;

[a] https://orcid.org/0000-0002-9882-6075
[b] https://orcid.org/0000-0002-1209-7968
[c] https://orcid.org/0000-0001-7992-0392

Dörndorfer, J., Schmidtner, M. and Timinger, H.
Developing Context-aware Applications in Automotive Environments.
DOI: 10.5220/0013738700004000

In *Proceedings of the 17th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2025) - Volume 2: KEOD and KMIS*, pages 438-449
ISBN: 978-989-758-769-6; ISSN: 2184-3228

Hohl et al., 2016). Hence, the development process in the automotive sector is becoming more complex and must be able to adapt to new technologies while still being a distributed process. To determine the status quo along the supply chain, different companies were interviewed about the challenges in automotive software development and whether information modeling has been used so far to meet the challenges. Additionally, to cope with the assumed challenges, we want to propose a modeling language called SenSoMod. It could enable developers to plan and document the context-specific requirements of their development in an easily understandable format and pass this knowledge along the supply chain. Having a better overview of the relevant context could enable both in-house and supply chain developers to work more efficiently.

1.1 The Flexibilization of Business Processes

Business processes, which are sequences of activities and events that create value for customers, are fundamental to developing context-aware applications for enterprises. These processes are standard in both theory and practice (Becker et al., 2011; Vom Brocke and Rosemann, 2010; Bichler et al., 2016). Therefore, the development of context-aware applications must consider the associated business processes and their contextual influences. Context and context-awareness definitions apply to both mobile applications and business processes. A context-aware application supports business processes by providing relevant information and services, achieved by evaluating and aggregating raw sensor data. To manage this complexity, a modeling language and tool should incorporate context. Rosemann et al. (Rosemann et al., 2011). found that business process languages need more flexibility to handle contextual influences, reducing time-to-market for products and services. The event-driven process chain (EPC) was extended to meet this demand for flexibility, though it primarily addresses general process flexibility rather than specific context modeling (Rosemann and Van der Aalst, 2007). Several approaches have been developed (Saidani and Nurcan, 2007; de La Vara et al., 2010; Heinrich and Schön, 2015; Dörndorfer and Seel, 2017; Conforti et al., 2013; Santra, 2024) to incorporate context into business processes during design time and runtime. However, these methods often fail to specify the origins and aggregation of raw data into context information. Software developers frequently need to write software for domains they do not fully understand, necessitating collaboration with domain experts. Information modeling is crucial for this collaboration, enabling a shared understanding of the domain through abstraction and graphical representation (Evans, 2011). Successful application requires mutual understanding of the model language and domain model by both developers and experts. Alalshuhai and Siewe extended UML diagrams (Alalshuhai and Siewe, 2015a; Al-alshuhai and Siewe, 2015b) to depict context, enhancing class and activity diagrams to be context-aware. Other approaches (Kramer et al., 2010; Hoyos et al., 2013) have developed domain-specific languages (DSLs) for handling context in mobile device development and context-aware systems, focusing on application transformation across platforms rather than data aggregation for context.

1.2 Context Terminology

Early attempts to define ”context” for mobile computing began in the 1990s. Schilit and Theimer's (Schilit and Theimer, 1994) definition focused on location, but Dey et al. (Dey et al., 2001) noted that context is broader, encompassing more than just location. Definitions by Franklin and Flachsbart (Franklin and Flachsbart, 2001), Brown (Brown, 1996), Hull et al. (Hull et al., 1997), Ward and Rodden et al. (Ward et al., 1997) used terms like situation and environment as synonyms for context, but these did not fully capture the concept. Abowed and Mynatt (Abowd and Mynatt, 2000) proposed defining context using the five W's (who, what, where, when, and why).

Dey (Dey et al., 2001) defined context as any information characterizing the situation of an entity, which can be a person, place, or object relevant to user-application interaction. This definition helps distinguish context information from raw data and is widely accepted in research. Sanchez et al. (Sanchez et al., 2006) differentiated raw data (unprocessed data from sources like sensors) from context information (processed data with added metadata). Dey and Abowd (Dey, 2000) defined context-awareness as a ”system's use of context to provide relevant information or services based on the user's task.” This definition, unlike earlier ones by Schilit and Theimer (Schilit and Theimer, 1994) and Ryan et al. (Ryan et al., 1998), clearly distinguishes context-aware systems. Henricksen's definition of a context model (Henricksen and Indulska, 2004), based on Dey et al.'s (Dey et al., 2001) research, identifies context subsets from sensors, applications, and users that can be exploited in task execution. This model evolves over time and is explicitly specified by the application developer.

2 CONTEXT AWARE MODELING WITH SenSoMod

In order to cope with the development of complex context-aware applications, a modeling language called SenSoMod was developed. Context is measurable through sensors (Henricksen and Indulska, 2004), which are the source for retrieving raw data. Sensors can be further distinguished as physical and virtual sensors. Physical sensors measure physical quantities such as acceleration or humidity. Virtual sensors, in contrast, determine virtual data such as database entries (driver information) or data from another application (smartphone app of the car). These two types of sensors, which obtain raw data that has not been previously processed or aggregated, are called atomic sensors. However, data can be processed and aggregated (for example, weather could be the combination of temperature, wind intensity, and humidity) and serve as input data for a context. This aggregation can be modeled with a computed sensor (cf. Table 1) within the DSML. A computed sensor has to be based on at least one other sensor, regardless of whether it is one of the two atomic or another computed sensor. The description of the outgoing objects of a sensor can be stated in the Out-area. The form of the description is as follows: first, the name of the outgoing object can be stated, followed by the type of object. For specific values or ranges (such as, arrays, enums, or lists), a bullet list can be used. The computed sensor has, in addition, a decision logic (DL) area beneath the Out-area, which is dedicated to stating the logic for the outgoing objects in context free grammar. For example, the computed sensor position returns 'roomA' if the service set identifier (SSID) of the Wi-Fi connection is 'wifiRoomA'. All sensor types can serve as as input for a context element and must be based on at least one sensor. The context element also has a DL- and Out-area to represent its outgoing objects and decision logic. The context description represents a context characteristic that triggers an event in the mobile context-aware application (for example, a reminder of an appointment). The outgoing objects of the context element used in the context description serve as input.

One of the main considerations in distinguishing modeling constructs is their shape, as it strongly affects object recognition (Moody, 2009). Therefore, objects of similar types, such as atomic sensors, have similar shapes, whereas objects of different types, such as atomic sensors and context, have different shapes (cf. Table 1). The model notation was developed based on the principles of Clinton and Jafar (Jeffery and Al-Gharaibeh, 2015) and the recommendations for proper visualization by Deelmann and Loos (Deelmann and Loos, 2004). For instance, the same font, line thickness, and similar forms of notation were used to maintain the required consistency of Deelmann and Loos. Metaphors were also utilized wherever possible, as advocated by Deelmann and Loos, Clinton and Jafar. For instance, the established database symbol was utilized for the virtual and physical atomic sensors to better differentiate between them. A more detailed description of the DSML can be found in (Dörndorfer et al., 2019).

3 RESEARCH DESIGN

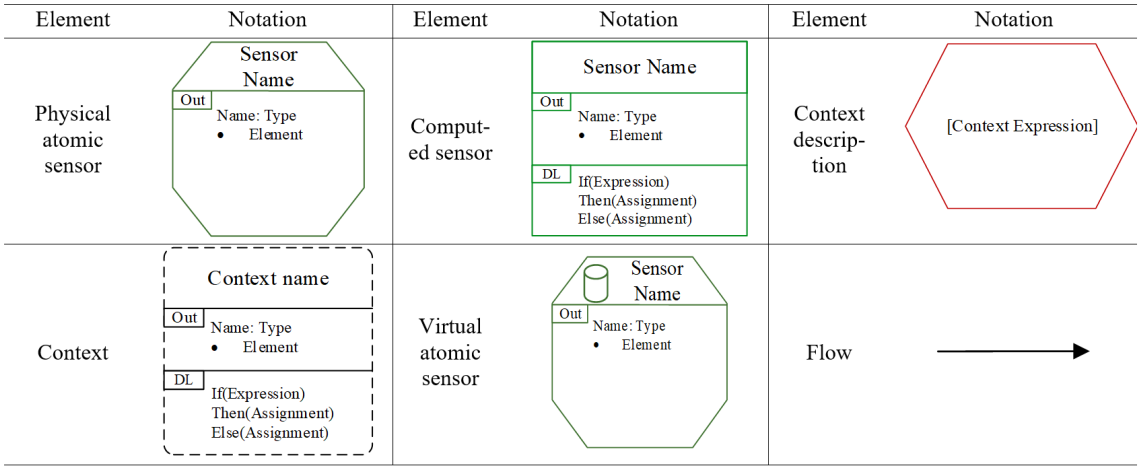
Based on the challenges identified in the introduction and related work sections, the following research objectives have been derived:

1. What challenges in developing (context aware) software do experts face in automotive development?
2. Can SenSoMod in particular be used to meet these challenges?

Every research project should be guided by an appropriate research method. To ensure a robust research process, a suitable research paradigm must be chosen. Hevner et al. (Hevner et al., 2004) proposes the approach of Design Science, which can be summarized as the creation of useful IT artifacts. The SenSoMod artifact was therefore created according to this paradigm. Subsequently, SenSoMod was evaluated for its usability in the development of mobile context-aware software. The evaluation, which has already been published (Dörndorfer and Seel, 2022), demonstrates that the language is easy to learn and also statistically demonstrates that DSML is more effective in modeling context for applications than standard modeling languages such as Unified Modeling Language (UML) and Business Process Model Notation (BPMN). It was also faster to model compared to standard languages. Thus, the use of DSML in the development of context-aware applications has significant advantages.

However, in order to address a similar set of context based challenges identified in the automotive development process, a series of expert interviews were conducted. The objectives of these interviews were to gather insights from experts in various sectors of the supply chain about the current state of context-aware software development and modeling. Additionally, SenSoMod was presented to them as a possible modeling tool and they were asked about its potential benefits and challenges in use. Table 2 shows that

Table 1: Notation of the domain-specific modeling language presented in (Dörndorfer et al., 2019).



data was collected from fourteen experts who are part of the product development process (PDP) in different German companies within the automotive sector. The interviews were conducted via Zoom or Microsoft Teams. To gain insight into different perspectives, interview partners were selected from both OEMs and tiers along the supply chain. The interviewees were chosen based on their positions in the company, with a focus on those in project management or development roles. Project managers were selected because they have an overview of the software architecture and can provide a different perspective on the challenges faced and whether SenSoMod can be of use to them. The interviewees were selected using the basic rules of theoretical sampling as outlined by Glaser and Strauss (Glaser and Strauss, 1998), based on their knowledge, work experience, and position. Table 2 provides an overview of the background of the expert interviews.

At the beginning of the interview, the interviewees were informed that SenSoMod is a research project being conducted as part of a dissertation. Hereafter, personal career data relevant to the survey was collected, such as the length of employment in the automotive industry or current position. Afterwards, the actual state of developing context-aware applications in their current position was queried. SenSoMod was introduced, and a hypothetical use case of a context-aware application in the automotive industry with SenSoMod was presented. Then, participants were asked questions regarding if and how SenSoMod could help in their current professional position, as well as their general impression of SenSoMod. Finally, each participant was asked standard questions that had to be answered on a Likert scale (Likert, 1932) of 1 (agree) to 5 (disagree). After transcription,

Table 2: Overview of the interviews including positions (CEO: Chief Executive Officer, PM: Project Manager, PTM Project Team Member, SA: Senior Advisor, SE: Systems Engineer, SWE: Software Engineer)

No.	Organization	Position	Experience	Duration
A	Tier	PM	7 years	38 min
B	Tier	PTM and SW	5 years	50 min
C	OEM	PTM	8 years	43 min
D	Tier	CEO	22 years	43 min
E	OEM	SWE	7 years	55 min
F	OEM	PM	6 years	36 min
G	OEM	PM	15 years	49 min
H	Tier	SA	17 years	34 min
I	Tier	PM	3 years	46 min
J	Tier	PM	3 years	47 min
K	OEM	PM	2 years	28 min
L	OEM	SWE	4 years	37 min
M	Tier	SE	7 years	48 min
N	OEM	PM	8 years	38 min

the interviews were analyzed using a three-step coding cycle following the Grounded Theory by Glaser and Strauss (Glaser and Strauss, 1998; Glaser and Strauss, 1967), which is a well-known methodology in many research studies (Chun Tie et al., 2019). In Grounded Theory, data pieces are encoded (labeled) to identify emerging concepts. Then the relations of the concepts to each other have to be explored by grouping them into categories and examining their connections. Finally, these categories are integrated around a central idea to build a theory grounded in the data. Initially, open coding was conducted, and one of the authors was given responsibility for this process, but coding was done jointly to allow for new and richer codes each time. New codes of subsequent interviews were constantly compared with codes of previous interviews, as suggested by the comparative

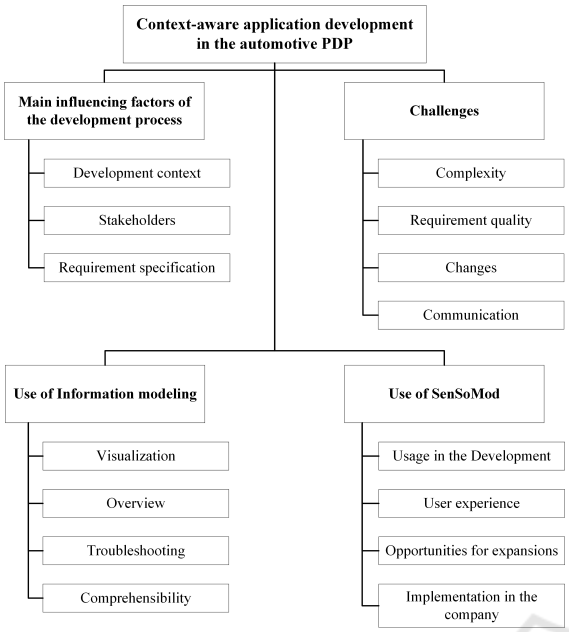


Figure 1: Coding taxonomy for context-aware application development in the PDP.

method of Glaser and Strauss (Glaser and Strauss, 1967) and Corbin and Strauss (Corbin and Strauss, 2015). The next step was to consolidate the codes into a common code scheme, a process that the authors performed together and adjusted until consensus was reached.

4 FINDINGS

4.1 Main Influencing Factors of the Development Process

The consolidation of codes was subsumed into a taxonomy.. Fig. 1 shows the coding taxonomy. The development context of the application, the stakeholders, and the requirement specification were identified as the main influencing factors of the development process.

The context of the development, such as how the component or software will be used in the vehicle, is not always clear. This applies not only to standard products from suppliers but also to the development of new components. Sometimes the suppliers themselves need to inquire in detail in their own interest. The following excerpts from the interviewees' statements provide an illustration of this.

Interviewee G – "Well, just like an external supplier, we receive the requirements [...] and do not necessarily know the use case behind them. That is actually the case."

Interviewee E – "There are some requirements that are initially ambiguous, and only later in the field do we get a better sense of what is needed. This can be quite challenging, as we may have a rough idea of the use case, but it still needs to be filled with logical requirements."

Stakeholders are a major influencing factor in the development process. They sometimes provide requirements with varying levels of detail, and strict requirements are also often imposed by regulatory bodies. Another important factor is the method of collecting requirements, which is typically determined by the client and can be either agile or traditional. The interviews revealed that the level of detail in requirements can vary greatly, ranging from very detailed to more general. The documentation of requirements typically consists of a mixture of text and diagrams, although sometimes it is purely text-based. Some teams use special tools for eliciting requirements, while others use simpler tools like Excel:

Interviewee B – "So we tend to set requirements at a higher level, and when it really gets down to the details, we actually expect the supplier to come up with proposals."

Interviewee D – "In some cases yes, in other cases there is no modeling language. In that case [...] it is then represented purely linguistically via a specification document."

4.2 Challenges

Employees in the automotive industry have identified several ongoing challenges, including requirements quality, complexity, changes, and communication. The quality of requirements is a significant challenge, according to the respondents. Interviewee B stated, "I experience poorly specified requirements on a daily basis." Often, requirements are described as a use case from which hardware and software implications must be derived without precise specification, as noted by Interviewee N: "Of course, there is a certain transfer because customers tend to describe their requirements in their use cases, and I have to translate them into the technical requirements for the sensor." Other challenges are the complexity of context-aware applications and the frequent requests for changes from customers. For example, Interviewee K stated, "You always have different deadlines that must be clearly defined and well-coordinated, and there are always difficulties. Especially when changes are necessary at short notice, and this has an influence on the spinning or the hardware." But constantly

changing suppliers can also be challenging. Interviewee B commented as follows: ”Since we work with service providers, there is this mentality of one company does it today, another one will do it tomorrow. This change, this fluctuation, obviously makes it even more difficult if you always have to start from a new baseline.” One point often mentioned is the challenge of communication. It was possible to distinguish communication problems in the three areas of technical, organizational, and requirements. An increased challenge can arise from new employees or suppliers, as Interviewee B stated: ”With new colleagues, it is often the case that at least at the first attempt, you talk a bit past each other.” But technical coordination is also perceived as challenging, as in the case of Interviewee A: ”There is a bit of divergence of opinions or the way in which something is implemented. Is it implemented in terms of hardware technology or in terms of functionality with the software?”

Organizationally, there are repeatedly communication problems both within a company and with suppliers: Interviewee K – “Yes, there are definitely hurdles, because the colleagues who take care of the software or the control units lead their own SE teams, and the coordination between the SE teams is of course always somewhat difficult.” Interviewee H – “Communication is our number one problem, both in-house and externally, so a good tool could of course prevent misunderstandings or communication problems.” There are also challenges in the communication of requirements, as the following interviewees indicated: Interviewee A - ”So the regular problem is that there are misunderstandings or misconceptions. Mostly, the concepts are designed relatively freely, there are then several (three, four) preliminary appointments (with the customer), where you harmonize technically and tighten the whole thing.” Interviewee M - ”Who is responsible for what and where does the interface start and where does it end? And that’s often quite fluid and are usually the major points of discussion.”

4.3 Use of Information Modeling

During the interview, the participants were asked about their views on modeling as a tool for eliciting specifications and requirements, as well as the benefits of using models in their daily operations. The advantages identified included the visualization of abstract concepts, domains or logic, a better overview of the domain, support in troubleshooting, and better comprehensibility (cf. Fig. 1). Modeling helps through visualization. Interviewee E stated: ”There are diagrams in between (in the requirements documents), which are supposed to illustrate something. [...] they simply illustrate what was tried to be described in the last 23 sentences before. The diagrams are there to put all the mentioned terms together somehow.” Interviewee C - ”The process is not always quite clear. What comes in, what comes out, what should happen in between? Yes, the flowcharts helped both on a small scale [...], but also on a large scale.” For the improved overview, the interviewees expressed the following: Interviewee E – ”You first model a simple application with one sensor and realize afterwards: Okay, actually I need five other sensors in the main configuration, then I need another temperature sensor in it [...], then it would just be good that I don’t have to look at the code every time.” Interviewee C – ”(Diagrams are useful) to recognize: What is there? What is ”everything” anyway, which software components are affected and where can it get stuck? Which interfaces are in between? The software developers we may have contracted externally also got that (the models).” Asked whether the diagrams are useful in troubleshooting, Interviewee C said: ”Well, I had the impression yes. Especially in the development of the mobile online services, where the customers collaborate with us on the services, it was always important that we also considered every smallest corner case.” Regarding comprehensibility, the interviewees stated the following: Interviewee M – ”One advantage of having a particular thing modeled, you can obviously quickly go into discussions with a customer or with other departments and quickly show something that everybody understands.”

4.4 Use of SenSoMod

A hypothetical use case, visualized in Fig. 2, was presented to the participants after the previous questions in order to give them an impression of the new modeling language. The goal of the hypothetical use case was to illustrate how SenSoMod works. The experts were able to ask comprehension questions at any time during the presentation. The hypothetical use case of the context-aware algorithm to adjust the lane-keeping assistant was chosen as a practical and appropriate example and should have a certain complexity to show the relevance and challenges of the topic.

This trend is evident as more car manufacturers replace physical buttons with software buttons on large screens. Tesla, which has almost no buttons at all, is a pioneer and a radical example of this trend (Tesla, 2020). Even established automakers like VW are embracing this shift, as seen with the ID.3 (Topgear - BBC Studios Distribution Limited, 2020). As inter-

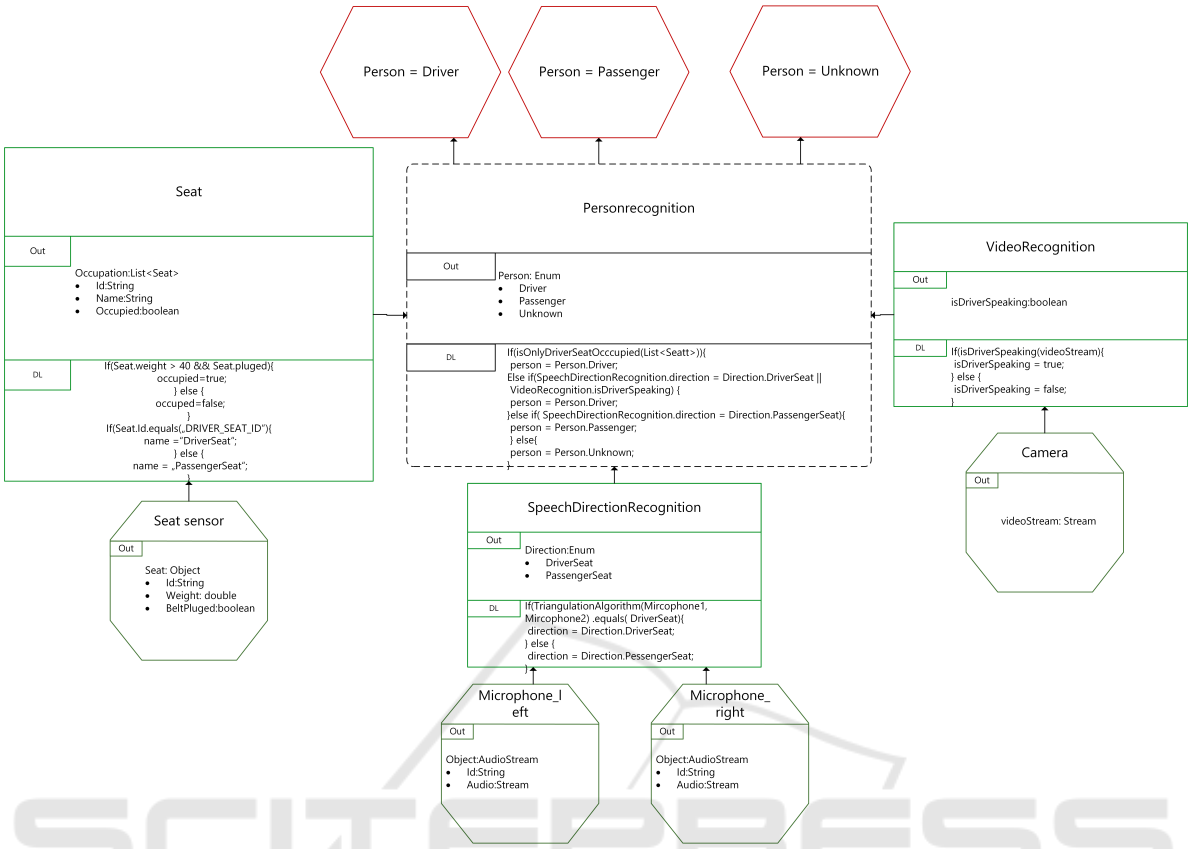


Figure 2: The context model to decide whether or not the user is the driver or a passenger, modeled with SenSoMod.

faces become more minimal, understanding the user's context becomes essential to ensure intuitive and supportive interaction. One important aspect of this is the distinction between authorized and unauthorized inputs, depending on who is making them. It can be assumed that certain inputs to the vehicle may only be made by the driver or the passenger via the context-aware application. For example, adjustments to the lane-keeping assist may only be conducted by the driver, as they have a direct impact on the driving behavior of the car. Thus, the context that needs to be evaluated is who made the input - the driver or the passenger?

A more detailed specification of the parameters for this gateway would overload the diagram and impair its clarity. Nevertheless, for context-aware application development, it is crucial to determine which data and sensors the decision is based on. Therefore, the context (driver or passenger) is modeled with DSML, as shown in Fig. 2. At first glance, the evaluation of whether the driver or passenger gives a command seems simple, but several sensors and data aggregation steps are necessary to make this decision. The evaluation is based on four "atomic sensors", which only provide data but do not perform any processing. These can be seen as green octagons in the figure at the bottom. For example, the seat sensor returns a data object "SeatData" with the Id, weight, and whether the seat belt is fastened. Above the seat sensor is the computed sensor "Seat". In total, there are three computed sensors (rectangles with solid lines) that evaluate the data from the respective atomic sensors.

The evaluation logic is specified in the DL area. For example, the Computed Sensor "Seat" determines for each seat whether someone is sitting on it or not, if the weight is more than 40 kg and the belt is fastened. If this is the case, the Boolean "occupied" is set to true. As a return value, the Computed Sensor returns a list of all evaluated seats. This is specified in the Out-area.

In the center of the figure is the Context Element Person recognition (rectangle with dashed line). This element represents the context that is necessary for a decision in a (business) process. In addition to the data from the Computed Sensor Seat, data from the Computed Sensors Speech Direction Recognition and Video Recognition serve as input for the context eval-

uation. There, the data aggregation works according to the same pattern as with the Computed Sensor Seat that was previously presented. The decision logic of how the context is determined is also described in the DL section of the context element.
The first step is to determine whether there are more people than the driver in the car. This is done by evaluating the seats. If only the driver's seat is occupied, only the driver can have given the command. If there are more people in the car, either the direction of the speaker can be determined or the camera that is directed at the driver can be evaluated. If the request comes from the direction of the driver or the recordings show that the driver spoke, then the request was made by the driver. However, if the request was made from the direction of a passenger, the request also came from the passenger. If it cannot be clearly determined who made the request, the return value "person" is specified as unknown. The return values of the context are specified in the Out-area. The central decision values resulting from the context aggregation are specified in the Context Description. After the demonstration, the interviewees raised several points, including the advantages and challenges of using the SenSoMod language in development, the user experience, opportunities for expansion, and implementation in the company. One of the advantages of using SenSoMod during development is that it allows for more room for innovation. It can also save time, improve cost estimation, and enhance quality. Additionally, interviewees noted that SenSoMod provides benefits in requirements elicitation and planning, as well as giving a good overview of the context and helping to find solutions. Improved communication is another advantage mentioned by the interviewees, as it ensures clarity and helps to prevent misunderstandings in interdisciplinary cooperation.

Despite the many advantages of SenSoMod, the interviewees also mentioned several challenges that could arise from its use. Technical hurdles were noted, such as the restriction of the solution space. As interviewee A stated, " it's usually the case that it's designed freely, because the more you specify, the more you restrict the companies. I think the way you do something like that, everybody has their own ideas."

A few respondents also identified organizational challenges. Interviewee N stated, "Such a language has a certain scope. You can only model what the language allows. Now, if I have or need anything that goes beyond that, then it becomes difficult." In some areas of development, there are also legal requirements that must be met, and the language would also have to comply with or guarantee these. Interviewee M said, "The tools have to be qualified so that no errors can occur when generating code, and I think the process is very elaborate." The user experience of SenSoMod was consistently positive.

Three main things were mentioned regarding expansion options. Firstly, there should be an option for visual demarcation, such as showing component boundaries. Secondly, a simulation with sensor data parameters would be desirable. This would enable testing of which context values are computed for specific sensor values. Thirdly, it was desired that other programming languages besides Java be generated from the model. The participants gave a differentiated assessment of whether SenSoMod can be used in the company. An added value of the language is definitely acknowledged, consisting mainly of an improved visualization of context computation and a better understanding of algorithm logic: Interviewee D - "There are people who need this represented in block diagrams like here (in your modeling language)." Interviewee K - "I don't know how it is elsewhere, but at our company (OEM), everyone has their component, and you communicate as a component through a clear interface. Within (the development of) a component, this (modeling) is definitely applicable. So, I could model my application the same way." Interviewee B - "The modeling would be super helpful in function development on the one hand, and good for explaining how my algorithm should work on the other." Obstacles for using the language in the company were identified, such as some people preferring textual descriptions, as pointed out by Interviewee D: "That depends on the people. There are people who need the whole thing presented in prose text." Increased workload due to implementation was also mentioned by Interviewee D: "As soon as there is a high workload, it becomes difficult to implement that, to get people to discipline to execute it that way." The language was found to be rather unsuitable for hardware developers as they prefer texts. The experts' answers give a good impression of the challenges in developing context-aware applications in the PDP and whether modeling can help to overcome them. Additionally, the modeling with SenSoMod was evaluated positively overall. This is also indicated by the evaluation of the standard questions that were asked to each expert at the end of the interview (cf. Table 3. According to this, 78.6% (42,9% agree, 35,7% slightly agree) of the experts interviewed found that modeling with SenSoMod can support collaboration with domain experts.

Table 3: Results from the evaluation of the standard questions.

	agree (1)	slightly agree	neither	slightly disagree	disagree	not applicable (5)
Did you find SenSoMod understandable in the scenario?	78,6%	21,4%	0,0%	0,0%	0,0%	0,0%
Were you able to distinguish well between the individual elements of SenSoMod?	28,6%	57,1%	7,1%	7,1%	0,0%	0,0%
Do you think that modeling with SenSoMod alone can facilitate the implementation of context-aware applications?	21,4%	35,7%	28,6%	0,0%	7,1%	7,1%
Do you think SenSoMod facilitates collaboration with the domain expert by graphically displaying context aggregation, making it easier to explain how the application adapts to the context?	42,9%	35,7%	14,3%	7,1%	0,0%	0,0%
Do you think that SenSoMod could in general be integrated or used in your product development process?	7,1%	50,0%	21,4%	7,1%	7,1%	7,1%

5 DISCUSSION

The purpose of this work was to investigate whether the use of a DSML such as SenSoMod can contribute to overcoming the challenges in the field of automotive PDP. To this end, SenSoMod was presented to the participants and they were asked for their assessment. Vital parts of context-aware software, such as sensors, are determined at a very early stage, making it necessary to provide reliable information about the data sources required for recognizing specific contexts. For instance, a rain sensor must be included to if the software should automatically start the windshield wiper when it rains.

With SenSoMod, it is possible to model the sources (sensors) of context aggregation, and the logic required for such aggregation can be specified and discussed with business departments early on. If the departments are not satisfied with the model, their feedback can be incorporated, allowing for an iterative process, which has proven useful in software development. Furthermore, SenSoMod allows for the generation of source code from the model in a later implementation phase. This can speed up the implementation process and lead to faster product development.

Furthermore, SenSoMod allows for a more precise specification of the required sensors. Since it is possible to model the necessary context recognition of the software in the design phase of the PDP, the sensor requirements can be derived from it. For example, if it is necessary to detect if someone is sitting in the car, this can be done in different ways, depending on the application's purpose. It can be determined by a seat sensor that detects a certain weight or by a camera in the vehicle compartment. If the purpose is to detect whether the driver is wearing a seat belt or not, it may be sufficient to install a seat sensor and a seat belt sensor. If the goal is also to detect drowsiness, a camera that provides images of the driver's face is needed. If the goal is to detect who is driving, a high-resolution biometric camera is needed. The sensor equipment variation can cause different costs or installation complexity. Thus, SenSoMod can be used to decide how elaborate the sensors have to be to implement certain context-aware software capabilities. This leads to a more precise specification of the sensors, which results in more accurate cost and effort estimation.

In addition, the model can create a common understanding of the required components among the OEM departments involved in the car's development. The model can also help overcome the lack of information flow problem between the OEM and suppliers, which is particularly prevalent in the early phases of the PDP. The model can ensure that a shared understanding between the different parties is established on why certain sensors are needed and must be considered to achieve context-aware software. This increases transparency between the departments involved and between the companies.

Based on the results of the interviews, a taxonomy was created and presented in section 4. It was confirmed that the PDP is a very rigid process. Requirements for the software are usually specified, which the supplier then has to implement, but often the con-

text is not known. For simple programs and components, this is not considered critical. However, study participants confirmed that the complexity of applications and the interlocking of individual components is increasing, making it more important to know the context of the application or component. Specification quality and communication were seen as further important topics. The quality of the specification depends on who you are working with. If you already know the customer or supplier and how they work, the requirements can be documented accurately and well. Communication is also important but is often neglected towards customers and suppliers, as well as within the company. Participants stated that diagrams are generally useful for visualization, comprehension, and problem identification, and give a good overview of software development, so that even technical experts without programming knowledge can understand algorithms. However, requirements are often described textually rather than with modeling. Thus, the current status quo has its challenges, such as frequent changes, communication, or increasing complexity, and modeling can remedy these by visualizing algorithms, increasing comprehensibility, and contributing to problem-solving.

When asked about the benefits of SenSoMod, participants generally gave a positive, albeit differentiated, response. This can also be seen in the standard questions that the participants were asked at the end of each interview. They had to answer these questions on a five-point Likert scale (Likert, 1932). The evaluation of the questions can be seen in Fig. 3. After all, 57.1% stated that SenSoMod could help them to implement context-aware applications in current developments. One participant differentiated and said that SenSoMod would have advantages especially for new projects. This is also in line with the statements of the participants in the open questions, who saw improved innovations, better requirements, and improved communications as advantages of the language. This was also confirmed in the standard questions, where 78.6% of participants said SenSoMod can facilitate collaboration with technical experts.

Structural reasons, such as long adaptation cycles in the automotive industry and legal requirements when it comes to code generation, were named as the main challenges in the use of SenSoMod. But disadvantages in the modeling were also seen. The solution scope should not be overly prescribed so that the supplier can realize their own ideas. All participants ultimately understood the language and its concept during the interviews. Also, most participants (85.7%) were able to distinguish the elements of the language well.

6 LIMITATIONS

While the study was designed to include experts from project management as well as software and hardware engineering between OEMs and suppliers, it must be noted that all experts from the OEM side were from only three different OEMs, while the experts from the supplier side came from different companies but were all familiar with the OEMs. Additionally, all experts work in the German automotive industry, so there may be a local bias in the findings.

7 CONCLUSION AND FURTHER WORK

This paper has shown that context information is becoming increasingly important in designing features for new cars. As the current automotive PDP is split between multiple supplier tiers and very different companies, efficient communication is necessary. In order to contribute, the DSML SenSoMod was presented and subsequently evaluated by experts. While there are still challenges in implementing a new modeling language in the process, such as obtaining organizational permission, most of the interviewed experts agreed that it would benefit their work from a technical and communication perspective. The interviews also resulted in new and interesting perspectives on additional functions that SenSoMod could provide in the future. The simulation aspect was particularly interesting, so that planned sensor systems and their logic could be tested in a theoretical environment. Possible next research steps could focus on the implementation of these new features or on how the organizational challenges could be handled.

REFERENCES

Abowd, G. D. and Mynatt, E. D. (2000). Charting past, present, and future research in ubiquitous computing. *ACM Transactions on Computer-Human Interaction*, 7(1):29–58.

Al-alshuhai, A. and Siewe, F. (2015a). An extension of class diagram to model the structure of context-aware systems. In *The Sixth International Joint Conference on Advances in Engineering and Technology (AET)*.

Al-alshuhai, A. and Siewe, F. (2015b). An extension of uml activity diagram to model the behaviour of context-aware systems. In *Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing (CIT/IUCC/DASC/PICOM)*, pages 431–437.

Becker, J., Kugeler, M., and Rosemann, M., editors (2011). *Process management: A guide for the design of business processes*. Springer, Berlin, 2. ed. edition.

Bichler, M., Frank, U., Avison, D., Malaurent, J., Fettke, P., Hovorka, D., Krämer, J., Schnurr, D., Müller, B., Suhl, L., and Thalheim, B. (2016). Erratum to: Theories in business and information systems engineering. *Business & Information Systems Engineering*.

Biesinger, F., Kras, B., and Weyrich, M. (2019). A survey on the necessity for a digital twin of production in the automotive industry. In *2019 23rd International Conference on Mechatronics Technology (ICMT)*. IEEE.

Biesinger, F., Meike, D., Krass, B., and Weyrich, M. (2018). A case study for a digital twin of body-in-white production systems general concept for automated updating of planning projects in the digital factory. In *2018 IEEE 23rd International Conference on Emerging Technologies and Factory Automation (ETFA)*. IEEE.

Brown, P. J. (1996). The stick-e document: A framework for creating context-aware applications. *Electronic Publishing*, 9:1–14.

Broy, M. (05282006). Challenges in automotive software engineering. In Osterweil, L. J., Rombach, D., and Soffa, M. L., editors, *Proceedings of the 28th international conference on Software engineering*, pages 33–42, New York, NY, USA. ACM.

Chun Tie, Y., Birks, M., and Francis, K. (2019). Grounded theory research: A design framework for novice researchers. *SAGE open medicine*, 7:2050312118822927.

Conforti, R., La Rosa, M., Fortino, G., ter Hofstede, A. H., Recker, J., and Adams, M. (2013). Real-time risk monitoring in business processes: A sensor-based approach. *Journal of Systems and Software*, 86(11):2939–2965.

Corbin, J. M. and Strauss, A. L. (2015). *Basics of qualitative research: Techniques and procedures for developing grounded theory*. SAGE, Los Angeles and London and New Delhi and Singapore and Washington DC and Boston, fourth edition edition.

de La Vara, J. L., Ali, R., Dalpiaz, F., Sánchez, J., and Giorgini, P. (2010). Business processes contextualisation via context analysis. In Parsons, J., Saeki, M., Shoval, P., Woo, C., and Wand, Y., editors, *Conceptual modeling - ER 2010*, volume 6412 of *Lecture notes in computer science*, pages 471–476. Springer, Berlin.

Deelmann, T. and Loos, P. (2004). Grundsätze ordnungsmäßiger modellvisualisierung. In Rumpe, B., editor, *Modellierung 2004*, GI-Edition Proceedings, pages 289–290. Ges. für Informatik, Bonn.

Dey, A., Abowd, G., and Salber, D. (2001). A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications. *Human-Computer Interaction*, 16(2):97–166.

Dey, A. K. (2000). *Providing Architectural Support for Building Context-aware Applications*. PhD thesis, Atlanta, GA, USA.

Dörndorfer, J., Hopfensperger, F., and Seel, C. (2019). The sensomod-modeler – a model-driven architecture approach for mobile context-aware business applications. In Cappiello, C. and Carmona, M. R., editors, *CAiSE Forum Proceedings – Information Systems Engineering in Responsible Information Systems*, Lecture Notes in Business Information Processing. SPRINGER NATURE.

Dörndorfer, J. and Seel, C. (2017). A meta model based extension of bpmn 2.0 for mobile context sensitive business processes and applications. In Leimeister, J. M. and Brenner, W., editors, *Proceedings der 13. Internationalen Tagung Wirtschaftsinformatik (WI)*, pages 301–315, St. Gallen.

Dörndorfer, J. and Seel, C. (2022). Context modeling for the adaption of mobile business processes – an empirical usability evaluation. *Information Systems Frontiers*, 24(1):195–210.

Evans, E. (2011). *Domain-driven design: Tackling complexity in the heart of software*. Addison-Wesley, Upper Saddle River, NJ.

Fabbrini, F., Fusani, M., Lami, G., and Sivera, E. (28.07.2008 - 01.08.2008). Software engineering in the european automotive industry: Achievements and challenges. In *2008 32nd Annual IEEE International Computer Software and Applications Conference*, pages 1039–1044. IEEE.

Franklin, D. and Flachsbart, J. (2001). All gadget and no representation makes jack a dull environment.

Glaser, B. G. and Strauss, A. L. (1967). *The discovery of grounded theory: Strategies for qualitative research*. Aldine, Chicago.

Glaser, B. G. and Strauss, A. L. (1998). *Grounded theory: Strategien qualitativer Forschung*. Hans Huber Programmbereich Pflege. Huber, Bern.

Heinrich, B. and Schön, D. (2015). Automated planning of context-aware process models.

Henricksen, K. and Indulska, J. (2004). A software engineering framework for context-aware pervasive computing. In *Second IEEE Annual Conference on Pervasive Computing and Communications, 2004. Proceedings of the*, pages 77–86.

Hevner, A. R., March, S. T., Park, J., and Ram, S. (2004). Design science in information systems research. *MIS Q*, 28(1):75–105.

Hohl, P., Münch, J., Schneider, K., and Stupperich, M. (2016). Forces that prevent agile adoption in the automotive domain. In Abrahamsson, P., Jedlitschka, A., Nguyen Duc, A., Felderer, M., Amasaki, S., and Mikkonen, T., editors, *Product-Focused Software Process Improvement*, volume 10027 of *Lecture notes in computer science*, pages 468–476. Springer International Publishing, Cham.

Hoyos, J. R., García-Molina, J., and Botía, J. A. (2013). A domain-specific language for context modeling in context-aware systems. *Journal of Systems and Software*, 86(11):2890–2905.

Hull, R., Neaves, P., and Bedford-Roberts, J. (13-14 Oct. 1997). Towards situated computing. In *Digest of*

Papers. First International Symposium on Wearable Computers, pages 146–153. IEEE Comput. Soc.

Jeffery, C. and Al-Gharaibeh, J. (2015). *Writing virtual environments for software visualization*. Springer, New York, NY.

Kirk, R. (2015). Cars of the future: the internet of things in the automotive industry. *Network Security*, 2015(9):16–18.

Kraft, T., Alagesan, S., and Shah, J. (2021). The new war of the currents: The race to win the electric vehicle market. *SSRN Electronic Journal*.

Kramer, D., Clark, T., and Oussena, S. (2010). Mobdsl: A domain specific language for multiple mobile platform deployment. In *2010 IEEE International Conference on Networked Embedded Systems for Enterprise Applications*, pages 1–7. IEEE.

Likert, R. (1932). *A Technique for the Measurement of Attitudes: New York, Columbia Univ., Diss., 1932*. Archives of psychology, New York, NY.

Moody, D. (2009). The "physics" of notations: Toward a scientific basis for constructing visual notations in software engineering. *IEEE Transactions on Software Engineering*, 35(6):756–779.

Morris, D., Madzudzo, G., and Perez, A. G. (2018). Cybersecurity and the auto industry: the growing challenges presented by connected cars. *International Journal of Automotive Technology and Management*, 18(2):105.

Prenner, N., Klunder, J., Nolting, M., Sniehotta, O., and Schneider, K. (17.05.2021 - 19.05.2021). Challenges in the development of mobile online services in the automotive industry - a case study. In *2021 IEEE/ACM Joint 15th International Conference on Software and System Processes (ICSSP) and 16th ACM/IEEE International Conference on Global Software Engineering (ICGSE)*, pages 22–32. IEEE.

Rosemann, M., Recker, J., and Flender, C. (2011). Designing context-aware business processes. In Siau, K., Chiang, R., and Hardgrave, B. C., editors, *Systems analysis and design*, Advances in management information systems, pages 51–73. M.E. Sharpe, Armonk, NY u.a.

Rosemann, M. and Van der Aalst, W. (2007). A configurable reference modelling language. *Information Systems*, 32(1):1–23.

Ryan, N., Pascoe, J., and Morse, D. (1998). Enhanced reality fieldwork: the context-aware archaeological assistant. In Gaffney, V., van Leusen, M., Exxon, S., Ryan, N. S., Pascoe, J., and Morse, D. R., editors, *Computer Applications in Archaeology*, British Archaeological Reports, pages 269–274, Oxford. Tempus Reparatum.

Saidani, O. and Nurcan, S. (2007). Towards context aware business process modelling. In *Workshop on Business Process Modelling, Development, and Support*, page 1, Norway.

Sanchez, L., Lanza, J., Olsen, R., Bauer, M., and Girod-Genet, M. (2006). A generic context management framework for personal networking environments. In *3rd Annual International Conference on Mobile and Ubiquitous Systems - workshops, 2006*, pages 1–8, Piscataway, NJ. IEEE.

Santra, D. (2024). Design of a context-aware healthcare business process model offering an extension strategy in bpmn. In Mondal, S., Piuri, V., and Tavares, J. M. R. S., editors, *Intelligent Electrical Systems and Industrial Automation*, Studies in Autonomic, Data-driven and Industrial Computing, pages 263–277. Springer Nature Singapore, Singapore.

Schilit, B. N. and Theimer, M. M. (1994). Disseminating active map information to mobile hosts. *IEEE Network*, 8(5):22–32.

Stan, C. (2022). No more cars with internal combustion engines, but what about airplanes? In Stan, C., editor, *Energy versus Carbon Dioxide*, pages 51–56. Springer Berlin Heidelberg, Berlin, Heidelberg.

Tesla (2020). Model 3: Owner's manual. https://www.tesla.com/ownersmanual/model3/en_us/, accessed on 03.08.2025.

Topgear - BBC Studios Distribution Limited (2020). The top gear car review: Tesla model 3.

Vdovic, H., Babic, J., and Podobnik, V. (2019). Automotive software in connected and autonomous electric vehicles: A review. *IEEE Access*, 7:166365–166379.

Vom Brocke, J. and Rosemann, M., editors (2010). *Handbook on Business Process Management 2: Strategic Alignment, Governance, People and Culture*. International handbooks on information systems. Springer-Verlag Berlin Heidelberg, Berlin, Heidelberg.

Ward, A., Jones, A., and Hopper, A. (1997). A new location technique for the active office. *IEEE Personal Communications*, 4(5):42–47.

Winkelhake, U. (2021). *Digital transformation of the automotive industry, pp. 4-7 & 85–88*. Springer.