SYSTEMIC MECHATRONIC FUNCTION DEVELOPMENT

Pulm, Udo (1);
Stetter, Ralf (2)

1: Hamburg University of Applied Sciences (HAW Hamburg);
2: Ravensburg-Weingarten University of Applied Sciences (RWU)

ABSTRACT

Since decades, researchers are developing methods, tools, and processes for the development of mechatronic systems. In the last decade, this field has widened, including, amongst others, aspects such as cyber-physical systems (CPS) or the internet of things (IoT). Still, little concrete guidance can be identified for engineers who have to decide whether to realize a certain product function mechanically, electronically, through software functions, or with combinations thereof. The goal of the paper is to contribute to the development of guidelines for design engineers in which domain (or in which combination of domains and other areas) to allocate the solution for an engineering problem.

Keywords: Mechatronics, Function development, Systems Engineering (SE), Fault-tolerant design, Design methods

Contact:
Pulm, Udo
Hamburg University of Applied Sciences (HAW Hamburg)
Department of Mechanical Engineering
Germany
udo.pulm@haw-hamburg.de

Cite this article: Pulm, U., Stetter, R. (2021) 'Systemic Mechatronic Function Development', in *Proceedings of the International Conference on Engineering Design (ICED21),* Gothenburg, Sweden, 16-20 August 2021. DOI:10.1017/pds.2021.554

1 INTRODUCTION

Today, a still increasing number of product functions can be realized and enhanced by using electronic and software solutions. Additionally, new functions also require connectivity and cloud-based approaches. Despite these developments, many solutions or solution elements must remain mechanical as they concern concrete physics and the purpose of the product in a physical world. In the design process, engineers need to decide whether certain functions should be realized mechanically, electronically, with software, cloud-based, or even with other approaches (in this paper, the term "function" covers a wide scope including features and functionalities).

The paper reviews the state of the art concerning domain spanning function development processes in academia and industry, develops a model of the different levels of mechatronic function development, analyses industrial case studies, and formulates initial guidelines for engineers faced with the challenge of selecting the solution domain for engineering problems. The resulting guidelines are intended to support the development of multi-domain solution components and the systematic multi-criterion evaluation of these components and combinations thereof – and by that the collaboration and communication between the domains themselves.

Due to increasing product complexity, fault-tolerance is increasingly important and becomes a key aspect of the considerations. The analysis of the case studies showed that frequently non-technical aspects such as legal and social issues need to be taken into consideration for identifying the optimum solution. The contribution of the paper from a scientific point of view are a systemic view of the development of mechatronic and cyber-physical products including concrete initial guidelines for design engineers. It aims at both early phases, where the functions have to be assigned to the domains, as well as later stages, where problems can be solved within the diverse domains.

The next section serves as a basis for the further parts of this paper and will analyse the current state of the art of system development processes in industrial companies. Section 3 focuses on function development processes, i.e. which sub-processes are currently used in these processes and which methods and tools are applied for assisting these sub-processes. Case studies will be described and analysed in Section 4, and initial guidelines are derived in Section 5.

2 STATE OF THE ART

In line with the trend of digitalisation, more and more functions are realized using electronics and software. In the automotive sector for instance, electronic systems are expected to account for half of the product costs in 2030 (Restrepo et al. 2020). Other new functions such as automotive concierge systems (trunk delivery, on-demand fuelling and charging, etc.) are based on electronic systems and software, but do also require connectivity and cloud-based solution, but still some solutions or solution elements will remain mechanical as they concern concrete physics such as the tire-to-road contact. In the early phases of the system design, engineers – as well as managers and designers – have to decide how to realize a function, i.e. in which domain and department to place it.

Numerous research activities have led to powerful methods and tools assisting the development of mechatronic and cyber-physical systems (e.g. Rajkumar 2012, Zheng et al. 2017, Barbieri et al. 2014). Currently, a strong interest can be observed on integrated engineering frameworks (compare e.g. Walter et al. 2019, Stetter 2020a, Zech et al. 2019). However, up to now, little assistance is available for deciding whether a certain function should be realized mechanically, electronically, or with software, and further domains, such as connectivity, cloud-based solutions, or influenceable social and legal aspects, are hardly considered.

Due to increasing legal and quality requirements, nearly all industrial companies have meanwhile developed detailed process charts and descriptions of their design process. The variety of technical systems is enormous and so is the variety in these process charts concerning both the persons, departments and external entities involved as well as the included process steps and their (serial and parallel) sequence. Still, the underlying logic can be described based on the well-known V-model (compare VDI 2206 (2004)). A version emphasizing certain aspects is shown in Figure 1.

https://doi.org/10.1017/pds.2021.554 Published online by Cambridge University Press

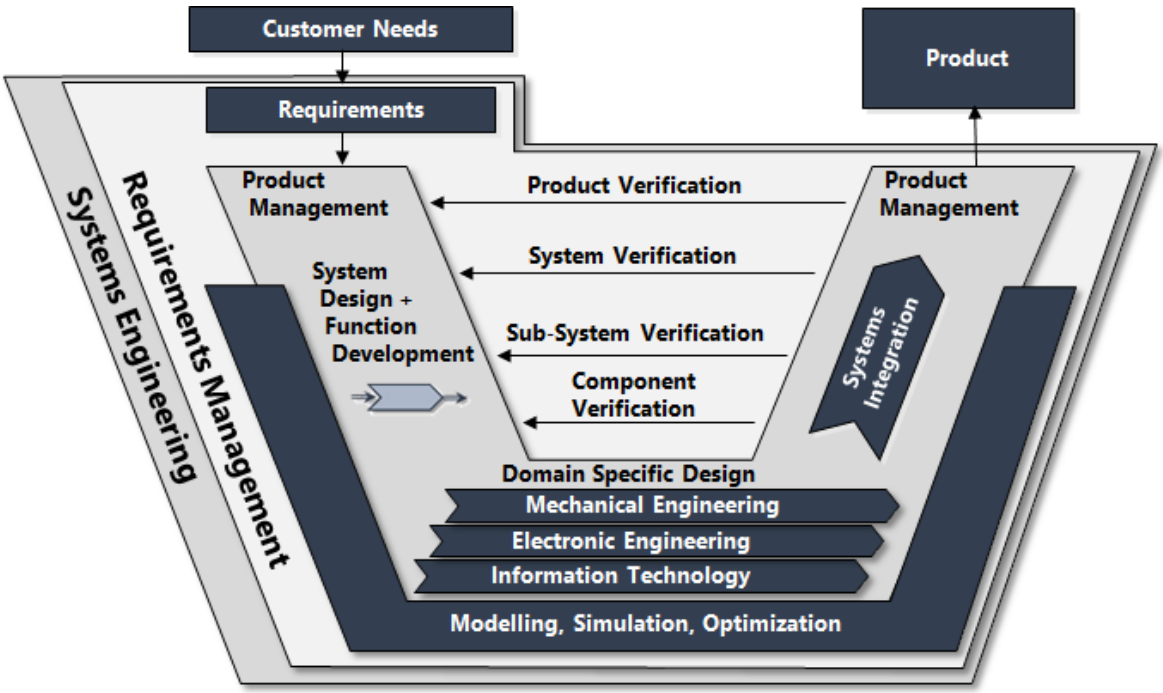

Figure 1. Essential elements of design processes of mechatronic systems

Throughout industry, a conscious requirements management (Holder et al. 2017) is established, which goes beyond a pure formulation and documentation of requirements towards, amongst others, a continuous tracking of their fulfilment. For consumer-oriented products, personnel were assigned to product management – persons who define what the technical system under development should deliver to the customer and who continuously monitor the realized functionalities and characteristics. A phase "systems design" or "functional development" is sometimes included in the process charts, but the sensible and necessary process steps and tools remain usually fuzzy. Verification processes on several levels are inevitable, however, often these different levels are not expressed and are hidden in large specification documents for components and sub-systems. In some cases, a certain degree of software support could be realized, but the system complexity and dynamics as well as the limited resources hinder the application. In the last decades, technical systems have become more and more complex and the share of electronics and software is increasing. Thus, faults cannot be completely avoided and an increased fault-tolerance is mandatory. Moreover, the complex interconnections in current systems primarily cause faulty behaviour in particular, not distinct "faults". These two circumstances lead to the necessity to incorporate a fault-tolerance development which includes fault-tolerant design (compare Stetter 2020b, Stetter et al. 2020) as well as integration and verification processes.

It can be mentioned that many of the regarded aspects are addressed by systems engineering or systems theory in the closer meaning of handling complexity, such as communication (i.e. coordination of behaviours) between various parts of both a product or an organisation, an enormous amount of testing (i.e. possible states of the system), clear modularisation (system thinking), differentiation of the domains (aspect systems in systems engineering or "systems" defined by media in general), delimitation of possibilities in the meaning of e.g. strict rules for software or electric engineering (closing of contingencies), etc. (Pulm 2005). These cannot be outlined in detail in the scope of this contribution.

Meanwhile, several enterprises (both original engineering manufacturers (OEMs) and suppliers) have created a function "systems architect" which is concerned with the technical functionality and realization on a cross-domain level. However, fundamental decisions are frequently made before the system architect is even involved in the project and his/her responsibilities are too large to allow a conscious decision how to realize a certain functionality. To conclude, detailed and sensible design process descriptions exist in industry, but relatively little attention is given to the function development, which is the topic of the subsequent section.

https://doi.org/10.1017/pds.2021.554 Published online by Cambridge University Press

3 SCOPE AND INDUSTRIAL PRACTICE

Though the term "function" is indeed fuzzy in industrial practice, it still has a very central meaning and high importance in the regarded companies regarding mechatronic design – which in this case is the automotive industry, especially the drive train design and here the development of the control unit. As intended by design methodology, functions

- serve as an abstract and solution neutral representation of the regarded component,
- by this, help communication between the different domains, and
- support breaking down requirements in terms of customer functions down to technical functions.

The latter can be imagined as some kind of "House of Quality" or "Quality Function Deployment", but focuses more on functions than on requirements (Figure 2). Such a representation can be enhanced by aspects regarding cloud-data and connectivity, field data, or legal constraints, showing which customer functions are supervised or legally controlled, and which technical functions either are controlled or can be supported by one of these aspects (e.g. emissions might be affected by connectivity in traffic jams, need field data for statistics, and is legally constrained; similar for technical functions).

				Technical Functions										Other Levels		
				Engine Speed Control			Combustion		Energy-/Torque Management				Diagnosis			
				Sense Speed	Control Speed	Limit Speed	Inject Fuel	Ignite Fuel Mix	Turn Engine	Generate Electricity	Control Torque	Control Energy	Get Signals	Show Warnings	Cloud and connectivity	Field data	Legal constraints
Customer Functions	Standstill	Start engine	...	x	x		x	x	x		x				x	x	
		Keep idling	...	x	x		x				x				x	x	
		Stopp engine	...				x	x							x	x	
	Drive	Accelerate	...	x	x		x	x			x		x	x	x	x	
		Decelerate	...	x			x	x		x	x	x			x	x	
		Driving	Parking	x	x						x	x			x	x	
		Manoeuvres	High Speed Driving				x	x			x			x	x	x	x
	Consumption	...	...				x	x		x		x			x	x	x
	Emissions	...	...	x			x	x					x	x	x	x	x
	Diagnose Engine	...	...	x	x								x	x	x	x	x
Domains	Mechanics					(x)	x	x	x	x			(x)	x			
	Electrics			x		(x)	x	x	x	x	x	x	x	x			
	Software				x	x	x				x	x		x			
Other Levels	Cloud and connectivity								x		x	x	x	x			
	Field data			x			x	x	x	x	x	x	x	x			
	Legal constraints					x		x	x				x	x			

Figure 2: Assigning customer functions to technical functions

Supported by respective computer tools, this helps tracking down customer functions to detailed mechanical and electrical components as well as software functions (which in the following can be transferred to software code manually or automatically), together with boundary conditions. It also helps integrating systems both for project controlling (project management in terms of degrees of maturity) and for verifying them on the respective level of granularity, just as the V-model proposes.

While the addressed software tools are able to handle very complex systems, it is often hard to keep those up to date within large organisations and tight processes. Nevertheless, the underlying logic is manifested within the organisation itself by implementing a so-called "function development" department. In first instance, this refers to software functions, but practically goes far beyond, which is explained by Figure 3 showing the basic structure of a mechatronic system as in VDI 2206 (2004).

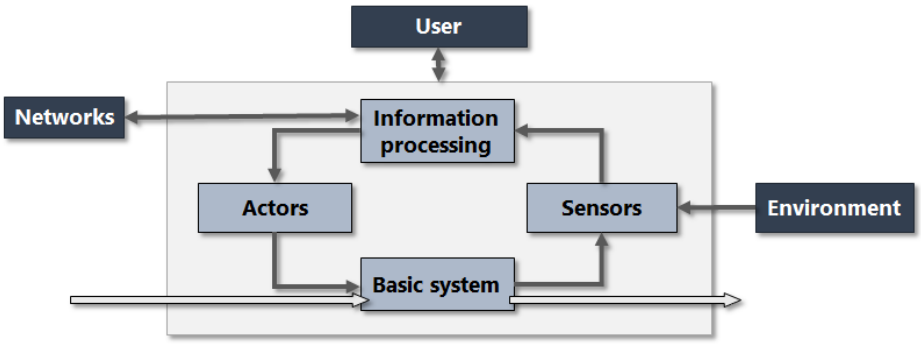

Figure 3: Basic structure of a mechatronic system

https://doi.org/10.1017/pds.2021.554 Published online by Cambridge University Press

The basic system is the main mechanical system realising the purpose of the mechatronic product. Its behaviour can be influenced and measured by certain electronic actuators and sensors respectively. The actuators are controlled by the software, i.e. the information processing – based on the information coming from the sensors. The whole mechatronic system is interfacing with the user, while mainly the software might be connected to a higher-level network – gaining and providing more information from and to other similar or related system. The software again is divided into the software code, i.e. parametric software functions, and calibration data determining the final behaviour of the product. Coming now from the other direction, there is a calibration department (or "application" as in "applying functions and solutions to the final product or project"), which sets the calibration data and by that is responsible for the final and detailed behaviour of the product. In doing so, they represent the customer or come closest to the customer in developing the product (on the synthesis side next to mere validation on the analysis side). But the responsibility or activity goes far beyond: They in fact have to overview the complete mechatronic system, which covers, amongst others,

- "applying" the actuators and sensors,
- testing the electronic system,
- assessing the mechanic basic system,
- understanding the customer and implementing their demands (often coming from the field),
- reacting to problems arising in the development process,
- and clearing the final product.

So, two more aspects are relevant here. First, the term functions has to be considered in a broader scope. Functions here cover functions, requirements, the behaviour of the product as well as properties, characteristics, and features as experienced by the user. On one hand, this might refer to qualities such as e.g. stability, performance, safety, etc. or specific properties such as consumption, pollution, noise, etc. which might be expressed as a function – but only at the expense of a comprehensible denomination. On the other hand, this might address certain levels of a function, as shown in Figure 4.

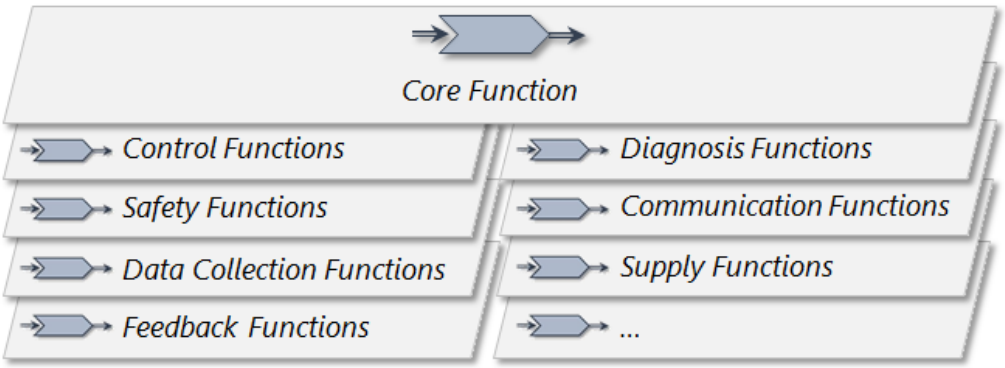

Figure 4. Levels of function development

The main function is split up or extended by other functions such as diagnosis, safety, data collection, or feedback to the user.

This finally leads to the second point: A function not only has to be assigned to a certain domain, the function rather exists in each domain to a certain degree and with certain characteristics. This again can be found in the fact that the basic mechanic system (from a functional view) is found within the software in some kind of digital representation or digital twin, "modelling" the whole product. And this again shows the responsibility of a calibration department for the whole mechatronic system. So, functions have to be both

- assigned to a certain domain (mechanics, electronics, or software), and
- split into these domains.

As a further notice, this assignment and splitting is not only limited to these domains and the mechatronic system, it might go further and address the environment, the customer, or a network as shown above in Figure 3. How this is linked to other aspects such as digitalisation, IoT, AI, cloud-based networks, etc. will be addressed in the outlook and described in further contributions.

https://doi.org/10.1017/pds.2021.554 Published online by Cambridge University Press

4 FUNCTION DEVELOPMENT EXAMPLES

This section describes three case studies concentrating on realizing certain functionalities – taken from a first collection of more examples derived from various projects, discussions, and small surveys (Figure 5). In the subsequent section insights and initial guidelines will be derived from these case studies.

Function	Aspect	Mechanics	Electronics	Software	Other	Criteria
Misfire	Detection/Prevention		(x)	x		Accuracy
Knocking	Control/Prevention		x	(x)		Reliability
Gear Switch	Comfort/Support	(x)	(x)	x		Time
Gear Switch	Accuracy (Right Gear)	x	x	x		Time
Catalyst	Endurance	x				Reliability
Camera	Resolution	x				Accuracy
Camera	Stabilisation	(x)	x	(x)		Comfort
Steering	Safety/Comfort/Space	x	x	x		Comfort with Reliability
Lock	Safety	x		x		Reliability
Safety Relief Valve	Safety	x	(x)	x		

x best suited *(x) suited with limitations*

Figure 5: Examples of functions possibly assigned to all domains

4.1 Ignition system of a motorcycle engine

The ignition system of a motorcycle is a mechatronic subsystem covering the combustion chamber and fuel-mixture, the fuel injector as the main actuator and the engine speed and positioning control as the main sensor, as well as the software within the electronic control unit (ECU). We will not discuss this main function but a legally required misfire detection function, i.e. preventing misfiring, which can damage the exhaust system and deteriorate emissions. This legal requirement had been adopted from passenger cars, but it had to be adjusted to motorcycle engines, which run with much higher engine speed and under much rougher boundary conditions. In passenger cars, the misfiring is (normally) detected by analysing the smoothness of the running engine via the engine speed sensor, i.e. a software function, which is challenging in motorcycles due to the mentioned reasons.
Possible solutions (Figure 6) are to improve the basic system ("mechanics") by implementing a second spark, improving the combustion by reducing e.g. the maximum engine speed – of course at the expense of performance –, or abstaining from a technical solution, since a sensitive driver will recognise a misfiring engine.

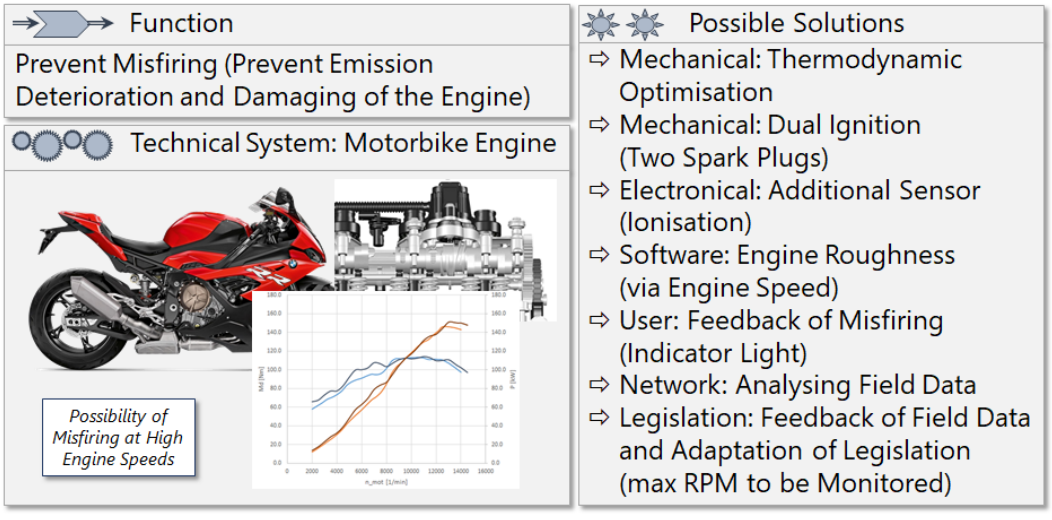

Figure 6. Preventing misfiring of a motorbike engine

The electronic solution would be a cost-intensive additional sensor measuring the ionisation in the combustion chamber. The third and finally chosen solution is the detection of rough engine running by analysing the speed signal of the engine – improved and extensively tested for motorcycles. Though this is the standard solution, this example shows some interesting aspects, because the system is no longer restricted to the technical system:

https://doi.org/10.1017/pds.2021.554 Published online by Cambridge University Press

- The user gets feedback from the engine via the cockpit, that misfiring occurred and he has to change his driving style or have his engine checked at a dealer.
- Field data is collected and analysed, whether engine misfiring occurred often (and under which circumstances), in order to optimise the system in the future.
- Possible engine faults are (statistically) reported to homologation authorities in order to control the emissions of the whole fleet. Additionally, legislation was adjusted to motorcycles beforehand, since legislations from passenger cars cannot be taken-over one-to-one

So, even more levels are added to the basic structure of a mechatronic system, maybe leading to a basic structure of a digitalised system.

4.2 Gear shifting system for a race-car

The gear box of the formula student driverless race-cars of the Ravensburg-Weingarten University (RWU) is the integrated sequential gear box which is attached to the four-cylinder in line motorcycle engine (Honda CBR 600). In the original application in the motorcycle, this gear-box is manually operated through a foot lever. In a driverless vehicle an automated shifting system is required and was developed by the student team at the RWU (Stetter et al. 2020). This gear shifting system disposes of an electrical motor equipped with an incremental encoder which allows obtaining the exact rotary position of the shaft of the electrical motor. Using a long shaft and a pair of spur gears the motion of the output shaft of this electrical motor is transferred to a modified shift drum. This shift drum causes a movement of the gear sleeves that allow the selection of a gear similar to the situation in the motor-bike. This shift drum also is connected to a potentiometer, located at the end opposite of the pair of spur gears. This potentiometer delivers an analogous resistance which is depending of the rotary position of the shifting drum. Additionally, at the end of the shift drum a module is connected which consists of an element formed like a star and a roller which is pressed against this star using a spring. This module forces the shift drum to certain rotary positions which corresponds to a position of the shift drum within which a gear is exactly engaged. The system and solutions which may enable the function "ensure engagement of gear" are shown in Figure 7.

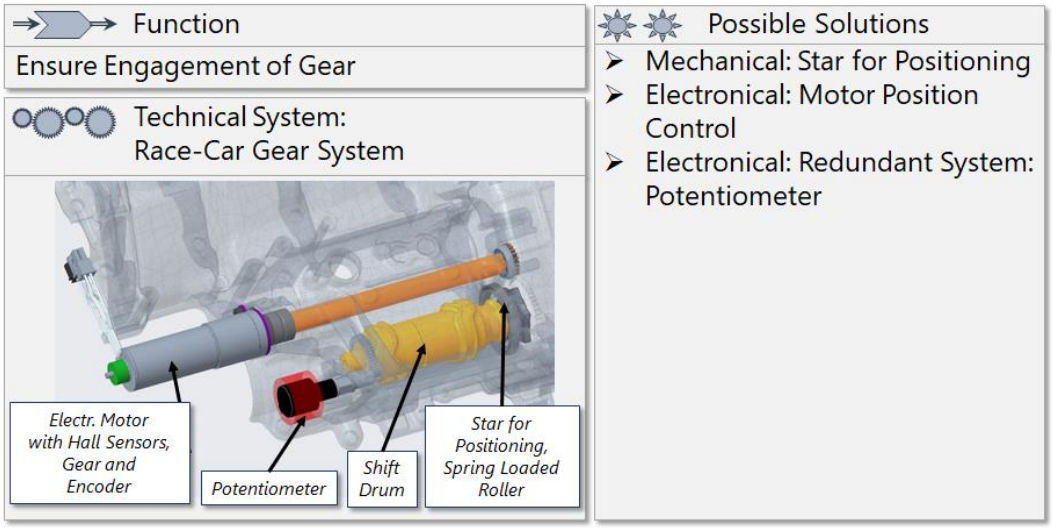

Figure 7. Ensure the engagement of gears in a race-car gear system

In the formula student competition, reliability is extremely important and a fault-tolerant design is of paramount importance (compare Stetter et al. 2020). Therefore, the redundant systems are present in the final design, which even differ on a functional and physical level, which increases the fault-tolerance (compare Stetter 2020b):

- the direct control is carried out by the position control of the electrical motor,
- the star module always supports this control and limits the influence of vibrations and
- the redundant potentiometer allows a monitoring of the system and an emergency operation in case of a fault of the position control.

https://doi.org/10.1017/pds.2021.554 Published online by Cambridge University Press

4.3 Power lift for a universal vehicle

The system under development was a power lift in the back of a vehicle (Reinprecht 2020). Such power lifts are used, for instance, for lifting ploughs at agricultural vehicles. In this case, the vehicle is not a typical agricultural tractor but a universal vehicle which also disposes of a platform in the back of the driver cabin. In a usual agricultural tractor, the driver can see the power lift through the back window of the cabin. He/she will watch the power lift and, for safety reasons, has to continuously press the operation lever or button. In the given case, the driver cannot see the power lift from the cabin, therefore another safe solution to ensure obstacle free movement is desirable. This function and some potential solutions are shown in Figure 8.

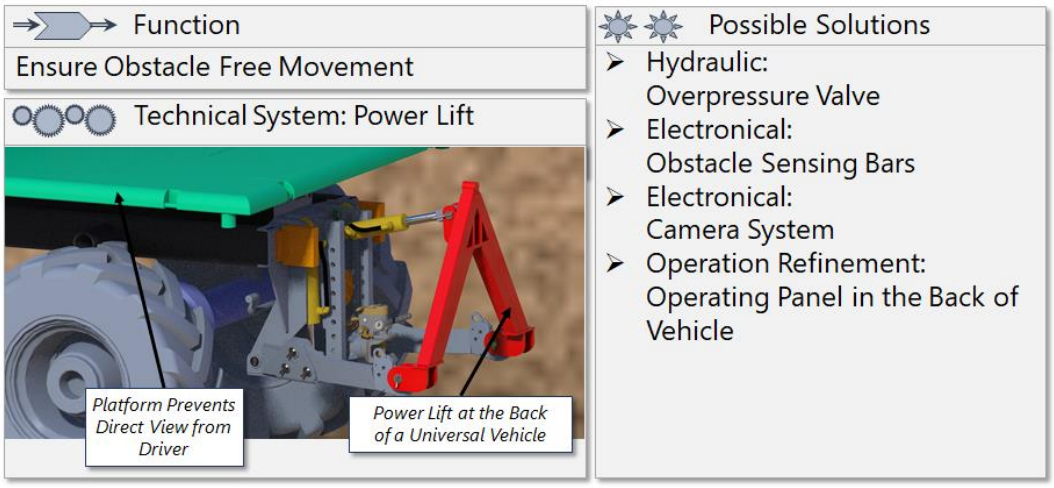

Figure 8. Ensure obstacle free movement of a power lift

In theory, it would be possible to use an overpressure valve for limiting the force of the power lift. However, the forces required e.g. for lifting a plough prevent this solution. Still, this solution is implemented, but only for preventing catastrophic consequences e.g. in case of a fault of the operation system. Another solution is necessary; theoretically it would be possible to attach a sensing bar on the moving elements of the power lift. However, in the harsh working environment of the power lift this solution is also not feasible. A solution based on a camera system, which allows the driver to watch the power lift is also problematic, because a sensible placement of the camera is nearly impossible and the lens would soon be dirty. The most reasonable solution is therefore to attach solid stare operating elements at the back of the vehicle similar to a common garbage truck.

5 DERIVED INSIGHTS AND GUIDELINES

In all three case studies, solution aspects can be found in all domains. It is a central insight from all case studies that the final solution is frequently a combination of mechanical, electrical, and software elements, and that it stretches beyond a pure technological solution. As frequently highlighted, a general shift towards software functions can be observed, because they can be realized rather quickly (though software integration and testing may still be an enduring and time-consuming task) and with low costs. Design engineers should be aware about the potential of multi-domain solution components and should be able to apply them in a systemic manner (e.g. to be aware of the multitude of consequences in a complex system and to realize a systematic application). Besides this general trend toward software functions, several functionalities still require mechanical or electrical solution components because these functionalities include energy or matter flows or concern forces and moments. A functional analysis employing methods and frameworks such as the integrated function modelling (IFM) framework (Eisenbart et al. 2016) allows to detect certain functional necessities and can assist the search for solution components which are directly connected to the functionality. The experience in the case study led to the hypothesis that a hierarchical function model showing substance, energy, signal and software levels and links between them might be helpful – this is, however, a topic for further research. In the function analysis, special attention should be given to control and diagnosis functionality, because these functionalities as well as user interface functionality will enable solution components in the software domain.

https://doi.org/10.1017/pds.2021.554 Published online by Cambridge University Press

From the case studies, two assistance ideas could be developed:

- Helpful could be an abstract database of solution elements with a description of certain functional characteristics. This database should include electronic, software and cloud solutions elements and should go beyond pure technical solution components including solution areas such as the user interaction and the legal domain.
- In order to assist fault-tolerant development, it would be sensible to develop methods to model redundancies on different level of abstraction, to support both synthesis and analysis, and to allow a functional topology optimization.

Obviously, these are initial concepts and the realization of these ideas will require intensive research work. The current trend towards more complex systems limits the applicability of a safe-life philosophy (to design components so that they never will fail under normal circumstances – compare Stetter 2020b) and design engineers need to be able to include a fail-safe philosophy in their solution search. They should also be aware of solution elements such as virtual sensors, virtual actuators, sensor fusion and sensor overlap (compare Stetter 2020b).

A systematic evaluation of solution components and combinations of solution components is inevitable in the process of choosing optimum solutions. Possible evaluation criteria with a preliminary mapping to the domains are shown in Figure 9. The analysis of the case studies made it apparent, that often no generally valid answer exists, but that several selection criteria need to be considered – while this assessment is being aggravated by many combined solution possibilities.

		Domain		
		Mechanics	Electronics	Software
Criteria	Cost	+	-	++
	Weight	o	o	++
	Space	+	+	++
	Reliability/Safety	++	+	-
	Accuracy	o	++	++
	Integration (Old Systems)	o	o	+
	Comfort	o	+	++
	Time	-	+	++
	Complexity/Functionality	-	++	++
	Performance	++	+	o
	Endurance/Wear Resistance	++	o	+
	Fault Tolerance	++	-	+
	Serviceability	++	+	+
	Boundary Conditions	o	o	o

++ most appropriate
+ appropriate
o needs consideration
- not appropriate

Figure 9: Criteria for the assignment of functions to certain domains

Furthermore, the evaluation should check whether the possibility to realize compensatory system, not cumulating systems is present (a compensatory system in contrast to a cumulating system is a system that tends to achieve a safe and satisfactory system state in the case of disturbances and faults – in a cumulating system a deviation from the normal system state may build up possibly leading to catastrophic consequences). The evaluation should also check whether a capability for self-diagnosis or integrated diagnosis, a capability to detect maintenance necessities and a capability to contribute to system communication are present. An essential criterion is also the testability of the technical system under development. All in all, there seems to be a general trend towards the assignment of functions to the digital software domain. In basically mechanical engineering products, this trend is limited, but it leads to further research and our outlook.

6 SUMMARY AND OUTLOOK

During the direct participation in and consulting of mechatronic design processes, the authors became aware that processes in the functional development stage are still rather fuzzy in industrial companies and that little assistance exists for design engineers faced with the question how to realize a certain functionality (how in the meaning of mechanical, electronical, with software or cloud-based solutions, or with combinations thereof). They became further aware that non-technological aspects play a decisive role and that sensible solutions frequently necessitate a systemic approach. Another important insight was the enormous importance of fault-tolerance for more and more complex technical systems. This paper explains and underlines these findings based on three case studies. Obviously, the gathered

https://doi.org/10.1017/pds.2021.554 Published online by Cambridge University Press

insights cannot deliver a complete picture of the current industrial situation and the derived guidelines only deliver an initial guidance. The authors wish to initiate research efforts and an intensive discussion concerning the function development – the decision made in this stage are of paramount importance for the performance, quality, and fault-tolerance of the technical system under development. The experience in the case study indicates a merit of hierarchical function models model showing substance, energy, signal and software levels and links between them – further research is planned in order to analyse this possibility. Regarding fault-tolerance and digitalisation, a reverse research approach or respective research questions might be helpful. For fault-tolerance, this might be not to prevent faults, but to generally assume that there will be many faults or faulty behaviour in the system (and the design process), thus one has to ask how to deal with or react to those faults. Regarding digitalisation, the question might be which functions can be digitalised and which not – or even more generally, how any function in a broad sense can be digitalised or digitally supported. Are there any functions that cannot be digitalised?

ACKNOWLEDGMENTS

The project "digital product life-cycle (ZaFH)" is supported by a grant from the European Regional Development Fund and the Ministry of Science, Research and the Arts of Baden-Württemberg, Germany (information under: https://efre-bw.de/).

REFERENCES

Barbieri, G., Fantuzzi, C. and Borsare, R. (2014) "A model-based design methodology for the development of mechatronic systems". Mechatronics 24, 2014, pp. 833 – 843, http://dx.doi.org/10.1016/j.mechatronics.2013.12.004.

Eisenbart, B., Gericke, K., Blessing, L. and McAloone, T (2016) "A DSM-based Framework for Integrated Function Modelling: Concept, Application and Evaluation", Research in Engineering Design, 2016, Vol. 28 No. 1, pp. 25-51. https://doi.org/10.1007/s00163-016-0228-1.

Holder, K., Zech, A., Ramsaier, M., Stetter, R., Niedermeier, H.-P., Rudolph, S. and Till, M. (2017) "Model-Based Requirements Management in Gear Systems Design based on Graph-Based Design Languages". Appl. Sci. 2017, 7, 1112, https://dx.doi.org/10.3390/app7111112.

Pulm, U. (2005) "Product Development as a Complex Social System". Proceedings of the 15th International Conference on Engineering Design 2005 (ICED05), Melbourne: The Design Society 2005.

Rajkumar R. (2012) "A cyber-physical future", Proceedings of the IEEE 2012, Vol. 100, Special Centennial Issue, pp. 1309 – 1312.

Reinprecht, M. (2020) "Entwicklung eines Heckkrafthebers mit dem Schwerpunkt Fehlertoleranz". Thesis, RWU.

Restrepo, E., Lovik, A.N., Widmer, R., Wäger, P., and Müller, D.B. (2020) "Effects of car electronics penetration, integration and downsizing on their recycling potentials". Resources, Conservation & Recycling: X 6 (2020) 100032.

Stetter, R. (2020a) "Approaches for Modelling the Physical Behavior of Technical Systems on the Example of Wind Turbines". Energies (2020), Vol. 13, No. 8, 2087, https://doi.org/10.3390/en13082087.

Stetter, R. (2020b) "Fault-Tolerant Design and Control of Automated Vehicles and Processes; Insights for the Synthesis of Intelligent Systems"; Springer: Berlin, Germany, 2020, https://dx.doi.org/10.1007/978-3-030-12846-3.

Stetter, R.; Göser, R.; Gresser, S.; Till, M. and Witczak, M. (2020) Fault-tolerant design for increasing the reliability of an autonomous driving gear shifting system. Maintenance and Reliability, Vol. 22, No. 3, 2020, pp. 482 – 492. http://dx.doi.org/10.17531/ein2020.3.11.

VDI 2206 (2004) "Design methodology for mechatronic systems". Beuth.

Walter, B., Kaiser, D. and Rudolph, S. (2019) "Machine-executable Model-based Systems Engineering with design languages". In Complex Systems Design & Management; Banach, R., Razavi, J., Lesecq, S., Debicki, O., Mareau, N., Foucault, J., Correvon, M., Dudnik, G., Eds.; Springer: Berlin, Germany, 2019; https://dx.doi.org/10.1007/978-3-030-04209-7_25.

Zech, A., Stetter, R., Holder, K., Rudolph, S. and Till, M. (2019) "Novel approach for a holistic and completely digital represented product development process by using graph-based design languages". Procedia CIRP 2019, 79, 568–573, https://dx.doi.org/10.1016/j.procir.2019.02.102.

Zheng, C., Hehenberger, P., Le Duigou, J., Bricogne, M. and Eynard, B. (2017) "Multidisciplinary design methodology for mechatronic systems based on interface model". Research in Engineering Design, 2017 28, pp. 333 – 356. https://doi.org/10.1007/s00163-016-0243-2.

https://doi.org/10.1017/pds.2021.554 Published online by Cambridge University Press