THE AGILE TOOLBOX - ADAPTATION OF AGILEMPPS TO THE MECHATRONIC DEVELOPMENT PROCESS

Goevert, Kristin; Baumgartner, Michael; Lindemann, Udo
Technical University of Munich, Germany

Abstract

Product development is challenged by the customer needs of individualization or own space of action. To solve these challenges, new methods, procedures and principles for product development are necessary. Agile development methods from software development are a promising approach, which could be adapted to mechatronic development. However, many adaptions fail. Therefore this paper develops the agile toolbox, which supports the selective use of agile methods, procedures and principles (agileMPPs). One reason of failed adaptions is a non-consistent understanding of what agile development is. This paper first provides an overview of agile methods, procedures, and principles. They are analyzed and the results are structured in the agile toolbox. The toolbox allocates the agileMPPs at the phases of the v-model. So, the toolbox supports the selective use of agileMPPs in the mechatronic development process. Only the selective use of agileMPPs made the benefits of agile development possible. The contributions of the agile toolbox are evaluated within a series of expert interviews.

Keywords: Agile, Flexible development, Design methods, Organisation of product development, Mechatronics

Contact:
Kristin Goevert
Technical University of Munich
Chair of Product Development
Germany
goevert@pe.mw.tum.de

Please cite this paper as:
Surnames, Initials: *Title of paper*. In: Proceedings of the 21st International Conference on Engineering Design (ICED17), Vol. 4: Design Methods and Tools, Vancouver, Canada, 21.-25.08.2017.

1 INTRODUCTION

The modern customer likes to be flexible, choose from many options, make late decisions and have their own space of action. These factors influence society and thus the customer of every company. To meet the needs of customers, companies have to integrate the factors into the company culture. (Reichwald and Piller, 2009)
One option for meeting these challenges is development of "agile". Agile is defined as being "able to move quickly and easily" and stands for "quick, smart and clever" (Webster, 2016). In the field of product development, agile methods are mostly used in the field of software development. However, many companies are now trying to implement and adapt agile methods, procedure, and principles to mechatronic development processes as well. As advantages, companies see shorter development time, an increase in the creativity of developers, an opportunity to involve the customer and the possibility to allow later changes (Dullemond *et al.*, 2009). But the unstructured implementation of agile methods frequently ends in chaotic developments.
These companies understand that agility can have positive effects on development, but only if companies find the optimal balance between agile development and highly structured development. If the optimum could be found, on the one hand, creativity expands, ability to manage late changes successfully increases, and development becomes shorter. On the other hand, the necessary planning security and organisational structure can remain (see Figure 1). Structure in agile projects is equally important as in standard product developments to develop the right product and remain focused. In both types of projects, "structure" prescribes a procedural framework and not how things have to be done. However, in standard development projects this structure is more restrictive. (Schmidt and Paetzold, 2016)
From this, the aim of the paper is to develop an agile toolbox with agile methods, procedures, and principles for mechatronic processes. It shall enable companies to obtain an overview of agile development options as well as to understand that agile development is structured and does more than just give additional freedom to the developers.

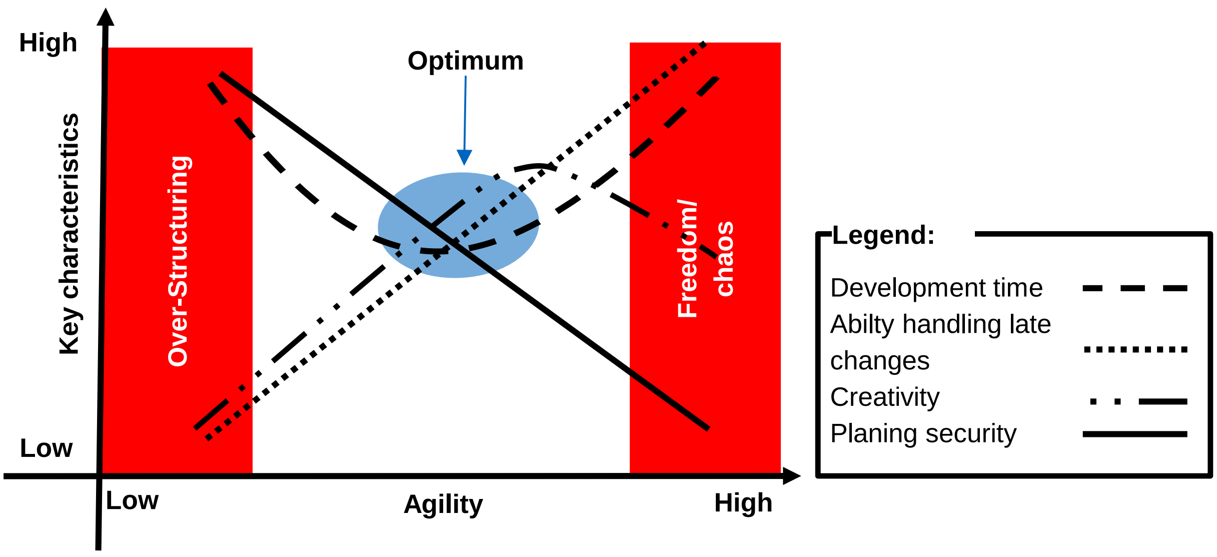


Figure 1. Optimum between over-structuring and high agility

2 LITERATURE BACKGROUND

This section presents definitions and differentiates between methods, procedures, and principles. Furthermore, it describes the basics of agile product development and provides examples of agile methods, procedures, and principles.

2.1 Definition of method, procedure and principle

This section defines and differentiate between the terms method, procedure, and principles (MPPs). In the following, the definitions help to classify and structure agileMPPs.

In general a method is "a systematic plan followed by presenting material for instruction" and "a body of skills or techniques" (Webster, 2016). The community of product development defines a method as a systematic plan of tasks intended to achieve a specific goal (Lindemann, 2009).
A procedure is "a series of actions that are done in a certain way or order" (Webster, 2016). A specific definition of procedure in the product development community is a series of methods used to achieve a specific subgoal (Lindemann, 2009).
Principles describe "an idea that forms the basis of something" (Webster, 2016). Transferred to product development, principles define basic rules of methods and procedures (Lindemann, 2009).
This paper follows the community specific definitions of the three terms.

2.2 Manifesto of agile product development

This section defines agile development. "Agile development combines creative teamwork with an intense focus on effectiveness and manoeuvrability" (Highsmith and Cockburn, 2001). A detailed definition of agile development is given in the agile manifesto of (Beck *et al.*, 2001). They are part of the software development community, where agile development originated. The agile manifesto defines 12 principles of agile software development (Beck *et al.*, 2001). The following list shows the principles. The authors of the paper adapt the software specific principles from Beck *et al.* (2001) to generally valid mechatronic principles:

1. "Our highest priority is to satisfy the customer through early and continuous delivery of valuable [products].
2. Welcome changing requirements, even late in development. Agile processes harness change for the customer`s competitive advantage.
3. Deliver working [product components] frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.
4. Business people and developers must work together daily throughout the project.
5. Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.
6. The most efficient and effective method of conveying information to and within a development team is a face-to-face conversation.
7. Working [product component] is the primary measure of progress.
8. Agile processes promote sustainable development. The sponsor, developers, and users should be able to maintain a constant pace indefinitely.
9. Continuous attention to technical excellence and good design enhances agility.
10. Simplicity - the art of maximizing the amount of work not done - is essential.
11. The best architectures, requirements, and designs emerge from self-organizing teams.
12. At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behaviour accordingly."

By following these principles, the four core values of the manifesto are realized. These values are the interaction between individuals, a working product as the focus of the development process, customer collaboration given importance, and responds to changes given more importance than following a plan (Beck *et al.*, 2001).

2.3 Examples of agile methods, processes and principles (agileMPPs)

This section presents examples of existing agile methods, processes, and principles (agileMPPs). Table 1 gives an overview. This table structures the agileMPPs in to methods, processes and principles, indicates if software is the domain source of the agileMPPs, and gives a short description of each agileMPP. Applicable categories are marked with an "x" in the table. Twenty agileMPPs are analyzed, categorized, and defined.
In the following, the User Story, as one example of agileMPP in Table 1, is described in detail. A User Story is an agile method, which comes from the domain of software development. User Stories help to identify requirements of the product at a high level of abstraction (Choma *et al.*, 2016; Cohn, 2010). It is a "short, simple description of a feature told from the perspective of a person who desires the new capability" (Cohn, 2010). So the user story is structured in three parts (Gloger and Margetich, 2014): the role (to identify the perspective of the story), the function or the feature of the product, and the value of the function or feature. Choma *et al.* (2016) use syntax to describe a similar user story "As a <type of user>, I want <some goal> so that <some reason>". Mostly the user stories are written on cards and

added to a poster (Cohn, 2010). The value of the method is that it is a defined requirement on a level which the customer understands and only the results of the development are in focus and not how to proceed (Gloger and Margetich, 2014).

Table 1. Detail overview of the agileMPPs

	AgileMPPs	Method	Procedure	Principle	Source Domain Software	Short describtion
1	User Stories	x			x	Method to identify requirements at a high level of abstraction from the customer perspective (Choma et al., 2016; Cohn, 2010)
2	Scrum-Board	x				It presents the user stories/ tasks on a card for every team member and supports a self-organized team. It clusters the tasks/ user stories in ToDo, in progress and done. (Pries-Heje, 2011)
3	Product Backlog	x			x	Includes every user story of the product (Gloger and Margetich, 2014)
4	Pair Programming			x	x	It is one principle of extreme programming. Two people work together. One person develops software and the other one plans the next steps (Sun et al., 2016)
5	Burndown-Charts	x			x	Presents the current state of the project progress to provide an overview of the remaining effort and wheather the defined goal is realistic (Gloger and Margetich, 2014)
6	Pull-Principle			x		Information, components etc. are requested from the next working step. (Tegel, 2012)
7	Scrum		x		x	Scrum is an iterative process which uses different other agileMPPs like sprints, user stories, scrum-board, product backlog or scrum master, product owner (Beedle et al., 1999)

2.4 Related Work

This section describes examples of related work. For example Sommer *et al.* (2013) and Klein (2016) describe the adaptation of agileMPPs of software to other domains. (Sommer *et al.*, 2013) integrate scrum into the stage-gate model. A stage-gate model defines development projects as a process of different activities (Sommer *et al.*, 2013), (Browning and Ramasesh, 2007).
(Klein, 2016) describes agile engineering in machinery and plant construction. His work is limited to the adaption of scrum to the new domain. To adapt the methods and principles of scrum he analyses the dependencies of the methods and principles and defines necessary fields of action (Klein, 2016).
Both focus only on Scrum and the adaption of this procedure. They don´t present a method toolbox for other agileMPPs.
Furthermore, software developers focus especially on agileMPPs. Based on the research of Komus *et al.*, (2014) currently 90% of the time agileMPPs are used in the context of software development. Therefore its principles and methods are combined into agile procedures like scrum or extreme programming. Those procedures describe new kinds of development processes, which are only adaptable to other domains in a few cases. An agile toolbox that provides an overview of agileMPPs and additionally allows an adaptation of those methods to other development processes which are not particularly connected to computer science doesn´t yet exist.

3 RESEARCH METHODOLOGY

First, this section describes the research methodology and research gap the paper tries to close. After that, a description of the requirements for the agile toolbox and a description of the analysis process of the agileMPPs follows.

3.1 Design Research Methodology

The research project applies the Design Research Methodology (DRM) of Blessing and Chakrabarti, (2009). Figure 2 shows an overview of the DRM adapted to the research project. It starts with the research clarification. This phase defines the goal of developing an agile toolbox, identifies the research gap, analyses the initial situation, and describes requirements for the agile toolbox (section 1 and 3.2 to 3.3). The second step of the descriptive study is an empirical data analysis. The initial situation becomes more detailed, an analysis of agileMPP data follows, and a deep understanding of the research field occurs (section 1, 2, 3.4 and 4.1). A prescriptive study is the third step (Blessing and Chakrabarti, 2009). The result of this step is an agile toolbox of methods, procedures and principles (section 4.2). Lastly, the descriptive study II implements expert interviews for the evaluation of the agile toolbox.

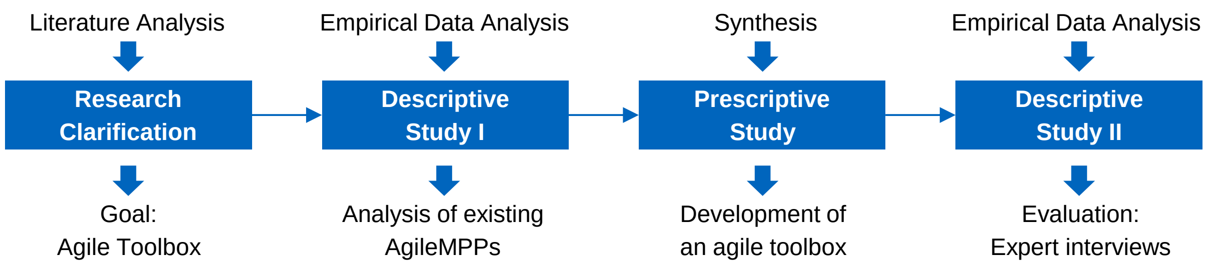


Figure 2. DRM adapted to the research of the paper inspired by (Blessing and Chakrabarti, 2009)

3.2 Research gap

The differences between the demands of the initial situation (section 1) and the literature background of agileMPPs (section 2) make the research gap clear. On the one hand, the industry demands increased agility and uses agileMPPs to develop mechatronic products. On the other hand, most of the agileMPPs developed for software development are not adapted to mechatronic development, and no overview of agileMPPs in the literature is suitable. Only scrum has been adapted partially to other domains (see section 2.4). From this research gap, the research question of this project was derived: how can existing agileMPPs be presented to support the mechatronic development process?

3.3 Requirements on the agile toolbox

The requirements for the agile toolbox were derived from the initial situation and basic requirements of toolboxes. There are three categories to classify the requirements: use, reference to the development process and content.

Table 2. Requirements of the agile toolbox

Categories	**Nr.**	**Requirements**
Use	1.	Applicabilty of the agile toolbox
	2.	Adabitibilty and Expandability of agileMPPs
	3.	Self-Explanatory presentation
	4.	Clearness of the agile toolbox
Reference to the development process	5.	Toolbox shows the phases of the development process
	6.	Clear categorization of agile MPPs to the development process phases
Content	7.	Integration of benefits and limitations of agileMPPs
	8.	Integration of methods independent of computer science

The aim of the agile toolbox is an improvement of the development process based on the principles of agile development. Accordingly, the agile toolbox will be used in companies to implement agileMPPs. Therefore the usability (use) of the toolbox is an important category in the list of requirements. Moreover the reference to the development process is necessary in order to provide user suitable agileMPPs at different phases of product development. Hence the reference to the development process leads to the

definition of additional requirements. Finally the information given, such as benefits or limitations of the agileMPPs determine which method is the most appropriate. In conclusion, the content of the agileMPPs is a further category for the characterization of requirements.
An overview of the requirements is given in Table 2. The requirements are derived from the initial situation and requirements, which can be found in the literature.

3.4 Analysis Process of agileMPPs

The analysis of the agileMPPs is a structured four step process. AgileMPP descriptions provide the input for the analysis. First, agileMPPs are assigned to the 12 principles of the agile manifesto. The second step describes the benefits and challenges of each agileMPP. In the next step, the adaptability of the agileMPPs is analysed. The last step categorizes in which phase of the v-model the agileMPPs can support the development process of mechatronic products. The v-model is chosen because it is one of the most popular procedure models of mechatronic development. To categorize the agileMPPs, criteria for each v-model phase were defined (see Table 3). Criteria were derived from v-model descriptions of (VMCD, 1997).

Table 3. Criteria of v-model phases to categorize agileMPPs in accordance to (VMCD, 1997)

Phase of the v-model	Categorization criteria
Overarching MPP	Only an integration over the whole process is possible
Requirements definition	Improvement of content related documentation
	Improvement of requirements visualization
	Optimization of the requirements identification process
Outline Design/ Detailed Design	Reduction of the development time
	Support of creativity and interaction
	Possibilty to integrate customer requests
	Optimization of the implementation process
Module implementation	Systematization and structuring of the process
	Optimization of testing process
Module-/ Integration-/ System test	Improvement of test sequence and order
	Optimization of testing process
Acceptance Test	Increased number of customer requests
	Optimization of testing process

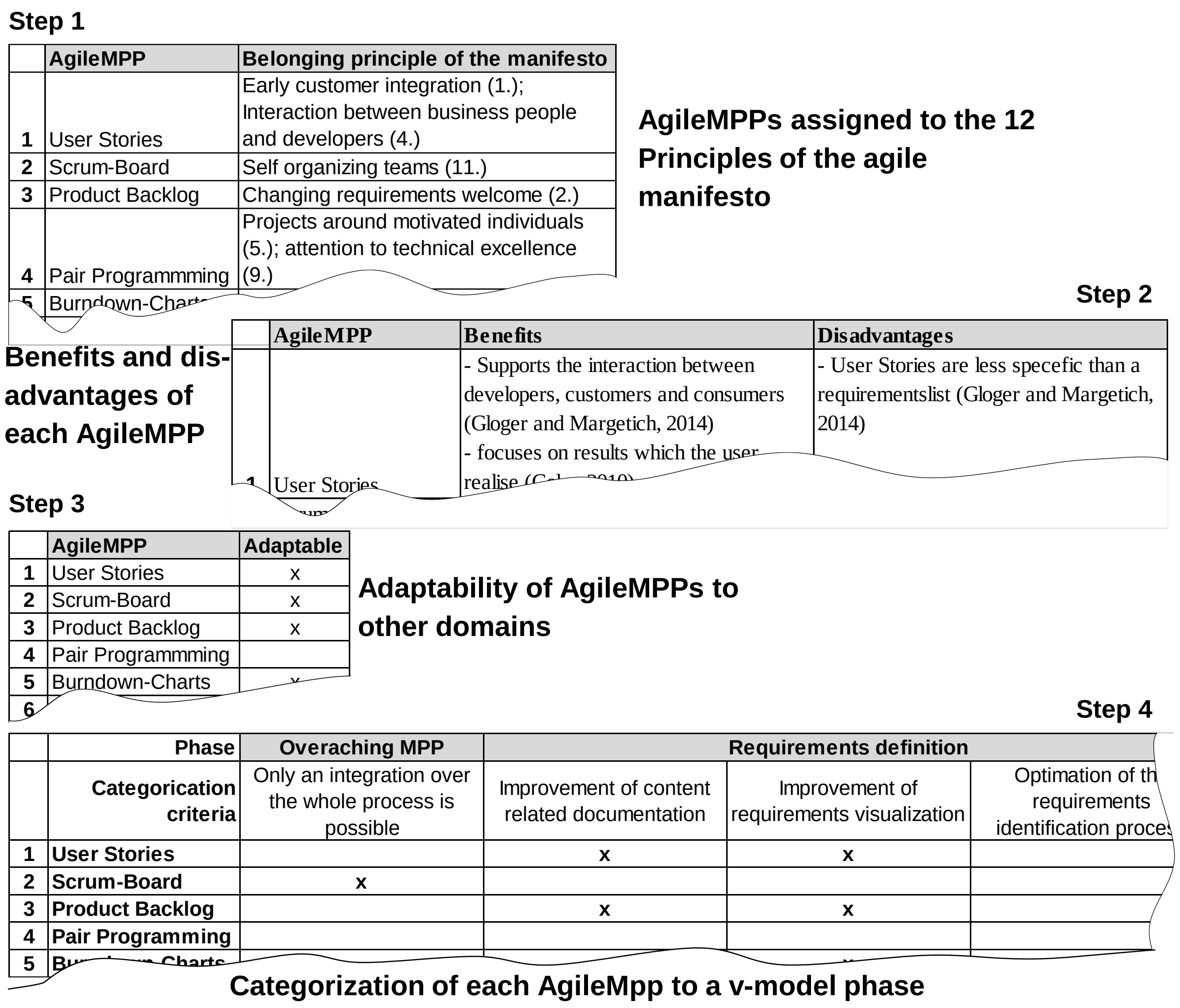
4 RESULTS OF THE AGILE-MPP-ANALYSIS AND TOOLBOX DEVELOPMENT

Section 4 presents the analysis results of the agile methods, procedures, and principles. Furthermore, this section shows an overview of the main result, the agile toolbox, and the implemented method sheets.

4.1 Analysis results of agile principles, methods and processes (agileMPP)

This section presents the analysis results of the four-step process, which is described in 3.4. Figure 2 presents an overview of the results of each step.
Step 1 shows a detailed view of the categorization of agileMPPs according to the 12 principles of the agile manifesto. For example, the agileMPP User Stories are assigned to two principles of the manifesto. First, customers are integrated early into the development process (principle 1). Second, the interaction between developers and business people must be supported because the requirements are on a high level of abstraction.
The second step identifies the benefits and disadvantages of each agileMPP. Benefits of the user stories, for example, is that they support of the interaction between customers and developers because they are written from a consumer perspective (Gloger and Margetich, 2014). Another benefit is the focus on results which the user perceives (Cohn, 2010). These benefits help to meet consumer needs and prevent

overengineering. A disadvantage of the user stories is the high level of abstraction (Gloger and Margetich, 2014). On the one hand, they are a benefit because more liberties for developers exist. On the other hand, uncertainties about the goal become greater for some people.
Step three presents the adaptability of each agileMPP. The User Stories source is the software domain, but this agileMPP is adaptable to mechatronic development processes. Even in the case of mechatronic development, requirements/ user stories could be written from a customer perspective and include the function and values of the function.
The fourth step assigned each agileMPP to a v-model phase. Therefore, the criteria of table 2 are used. User Stories are assigned to the phase requirements definition and acceptance tests. They are similar to requirements, and requirements can be used as acceptance criteria.



Step 1

	AgileMPP	Belonging principle of the manifesto
1	User Stories	Early customer integration (1.); Interaction between business people and developers (4.)
2	Scrum-Board	Self organizing teams (11.)
3	Product Backlog	Changing requirements welcome (2.)
4	Pair Programmming	Projects around motivated individuals (5.); attention to technical excellence (9.)
5	Burndown-Charts	

AgileMPPs assigned to the 12 Principles of the agile manifesto

Step 2

Benefits and dis-advantages of each AgileMPP

	AgileMPP	Benefits	Disadvantages
1	User Stories	- Supports the interaction between developers, customers and consumers (Gloger and Margetich, 2014) - focuses on results which the user realise	- User Stories are less specefic than a requirementslist (Gloger and Margetich, 2014)

Step 3

	AgileMPP	Adaptable
1	User Stories	x
2	Scrum-Board	x
3	Product Backlog	x
4	Pair Programmming	
5	Burndown-Charts	x
6		

Adaptability of AgileMPPs to other domains

Step 4

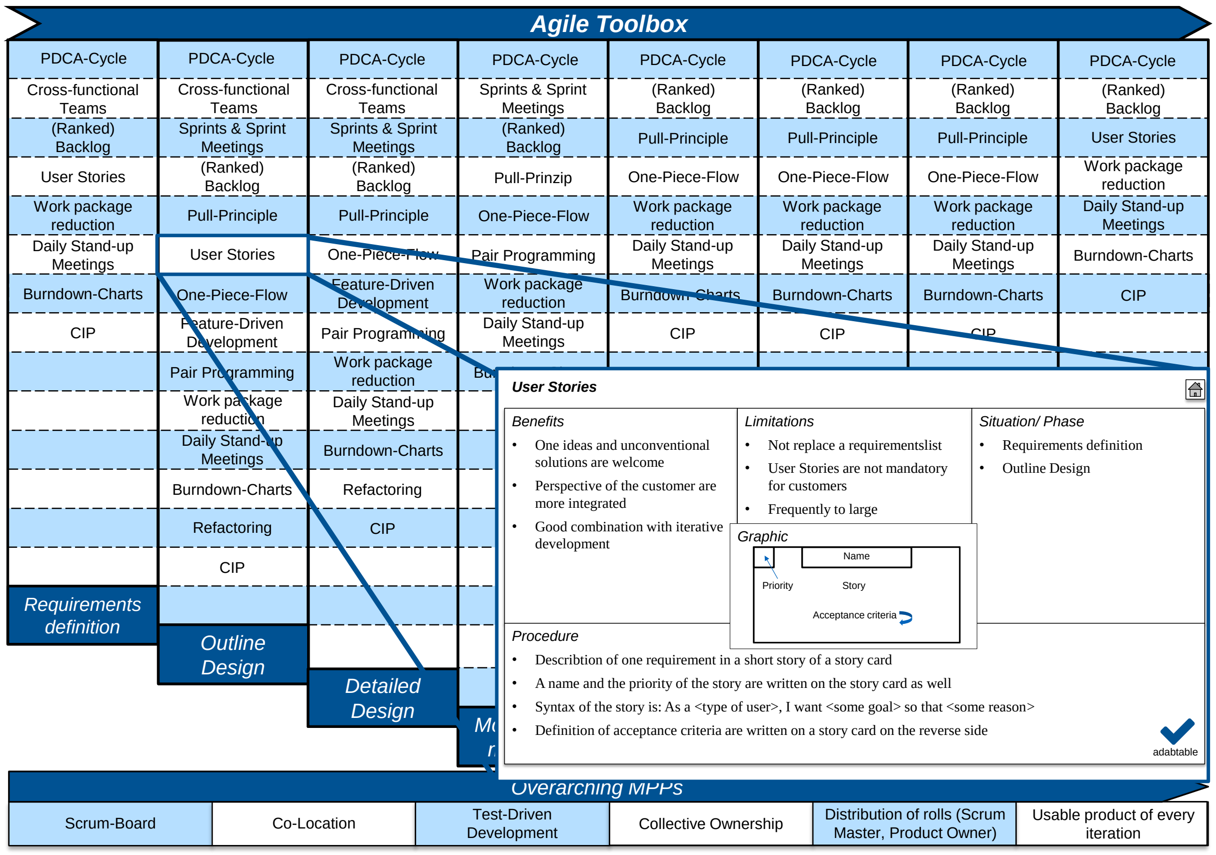
	Phase	Overaching MPP	Requirements definition		
	Categorication criteria	Only an integration over the whole process is possible	Improvement of content related documentation	Improvement of requirements visualization	Optimation of th requirements identification proces
1	User Stories		x	x	
2	Scrum-Board	x			
3	Product Backlog		x	x	
4	Pair Programming				
5	Burndown-Charts			x	

Categorization of each AgileMpp to a v-model phase

Figure 3. Overview of the agileMPP analysis process

4.2 Agile Toolbox

This section describes the main result of the paper. Figure 4 presents a detailed view of the agile toolbox. The basic structure of the agile toolbox is the v-model. The phases of the v-model are assigned to the agileMPPs. This results from the fourth analysis step (see section 4.1). One extra phase/ category is added to the structure. This category shows the overarching MPPs. These agileMPPs can only be implemented if they are implemented for the whole development process. One example is the scrum board because it captures the current situation of the development process. AgileMPPs, which are not in this category, are independently usable in each phase in which they are categorized. This is the reason that the agileMPP PDCA-Cycle is assigned to each phase of the v-model (Lindemann, 2009). The PDCA-Cycle can be used in each phase independently if the agileMPP has been used in the phase before. The v-model with the phases is an overview of the agileMPPs. Behind every agileMPP, a slide with detailed information about the agileMPP is deposited. Figure 4 shows the slide with detailed information about the user stories. It is a slide with six information fields. It presents benefits, limitations, usable situations/ phases, graphics, procedures and adaptability to other domains. This information helps the user to get a first overview of the agileMPP and supports the decision process for or against, the agileMPP.



User Stories

Benefits

- One ideas and unconventional solutions are welcome
- Perspective of the customer are more integrated
- Good combination with iterative development

Limitations

- Not replace a requirementslist
- User Stories are not mandatory for customers
- Frequently to large

Situation/ Phase

- Requirements definition
- Outline Design

Graphic

Procedure

- Describtion of one requirement in a short story of a story card
- A name and the priority of the story are written on the story card as well
- Syntax of the story is: As a <type of user>, I want <some goal> so that <some reason>
- Definition of acceptance criteria are written on a story card on the reverse side

Figure 4. Overview of the toolbox and agileMPP slide

5 EXPERT INTERVIEWS AND DISSCUSION

This section presents the validation of the requirements defined in section 3.3. To evaluate the fulfillment of the requirements, interviews where held with two experts in the field of customer quality engineering and quality management of the German automotive company Knorr-Bremse AG. Table 4 summarizes the results of the expert talks.

As Table 4 shows, all defined requirements are either completely or partly fulfilled. To provide an insight into the expert interviews, the three partly fulfilled requirements will be discussed. Furthermore suggestions given for improvements will be described.

The first requirement that was evaluated during the expert interviews was the applicability of the agile toolbox. To be able to choose suitable agileMPPs regardless of the state of development, it is important to provide all essential information in an overview. In this context, the experts liked that all agileMPPs and phases of the v-model were pictured in a single cover sheet. Moreover the validation of the toolbox showed that the clear structure of lines (agileMPPs) and columns (phases) make the agile toolbox an applicable tool which is very easy to use. Nevertheless, one of the experts mentioned that especially for inexperienced users in the field of agile development, the names of the agileMMPs have little meaning. Therefore, he suggested providing a real-life example for each agileMPP, connected to the slide with detailed information (see Figure 4).

As a further requirement of the agile toolbox, the adaptability and expandability of agileMPPs was discussed. Adaptability means whether the agileMPP can be used for mechatronic product development or is limited to computer science. During the expert interviews, sixteen of the twenty agileMPPs could be considered as not limited to computer science. In the experts´ opinion, the percentage of 80% of adaptable agileMPPs is very high. The four computer science specific agileMPPs are Feature-Driven Development, Pair Programming (Sun et al, 2016), Collective Ownership, and Refactoring. The

expandability of the agileMPPs in the toolbox depends on the size of the cover slide. Because of the necessity of a one-page cover slide for reasons of clarity, expandability is limited by the size of a regular page or slide.
The third partly fulfilled requirement is about the integration of the benefits and limitations of the agileMPPs in the toolbox. It was discussed whether the overview given of the benefits and limitations is sufficient to make a decision based on this content. The experts liked the variety of different benefits from the agileMPPs, which makes it easier to find a suitable method for a specific project or state of development. Furthermore, the position of the benefits and limitations in the agile toolbox was evaluated. As already described in the upper section, one of the experts recommended adding the benefits of the agileMPPs to the cover sheet in order to support the decision making of inexperienced users. Based on the statement of the other expert, the layout and integration of content as shown in Figure 4 is appropriate to the applicable use.

Table 4. Fulfilment of requirements on the agile toolbox based on expert interviews

Categories	**Nr.**	**Requirements**	**Fulfillment of requirements**
Use	1.	Applicabilty of the agile toolbox	**o**
	2.	Adaptability and Expandability of agileMPPs	**o**
	3.	Self-Explanatory presentation	+
	4.	Clearness of the agile toolbox	+
Reference to the development process	5.	Toolbox shows the phases of the development process	+
	6.	Clear categorization of agile MPPs to the development process phases	+
Content	7.	Integration of benefits and limitations of agileMPPs	**o**
	8.	Integration of methods independent of computer science	+

Legend:
+ requirement completely fulfilled o requirement partly fulfilled **-** requirement not fulfilled

6 CONCLUSION AND OUTLOOK

Companies will face challenges implementing of agileMPPs in the mechatronic development process. Sometimes agileMPPs adapt in a random way and the challenge of finding the optimum of high agility and over-structuring isn´t managed successfully. Furthermore, the differentiation of methods, procedures, and principles is not so clear in agile development. To make it more clear for non-software developers, first agile product development is defined following the agile manifesto of (Beck *et al.*, 2001), and related work in the field of agileMPPs adaption is presented. This shows the research gap because only scrum has been tested in mechatronic development processes, and now an overview of general agileMPPs exists. From the research gap, the following research question, can be derived: how can existing agileMPPs support the mechatronic development process?
In a first step, requirements of the agile toolbox were determined based on the initial situation defined by the research gap. Subsequently a four step agileMPP analysis was developed. In the first step of the analysis, agileMPPs were studied and connected to the 12 principles of the agile manifesto. The second step was to determine the benefits and limitations of the agileMPPs. As a third step, it was discussed whether the agileMPPs are adaptable to other domains or limited to computer science. The fourth and last step of the analysis was the categorization of the agileMPPs into the phases of the mechatronic development process of the v-model. Finally the toolbox was evaluated on the basis of expert interviews. As a result, the fulfilment of each requirement of the agile toolbox was discussed.
This paper presents an actual state of the research project and this section suggests further research. Until now the evaluation has been based on expert interviews. The next step will be to apply the agile toolbox in use cases of mechatronic development processes which follow the v-model. The second step is to evaluate the toolbox for other mechatronic development processes such as stage gate process.
Furthermore, already developed agileMPPs of the project or from the software-adapted agileMPPs can be added to the toolbox. For agileMPPs based from the software to the mechatronic process or to the

current situation, we will research on a methodical support. A method support helps to maintain the agile character and to find the optimum between over-structuring and high agility.
For the target implementation of agileMPPs and to understand the current situation of the companies. we also research a company-analysis from the agile perspective. The results of the analysis support tailor-made agileMPP choice in the situations.

REFERENCES

Beck, K., Beedle, M., van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., Mellor, S., Thomas, D., Grenning, J., Highsmith, J., Hunt, A., Jeffries, R., Kern, J., Marick, B., Schwaber, K. and Sutherland, J. (2001), "Manifesto for Agile Software Development", [online] Available at: https://www.agilealliance.org/agile101/the-agile-manifesto/

Beedle, M., Devos, M., Sharon, Y., Schwaber, K. and Sutherland, J. (1999), "SCRUM: An extension pattern language for hyperproductive software development", [online] available at: www.jeffsutherland.org/scrum/scrum_plop.pdf

Blessing, L.T. and Chakrabarti, A. (2009), *DRM, a Design Research Methodology*, Springer, London.

Browning, T.R. and Ramasesh, R.V. (2007), "A Survey of Activity Network-Based Process Models for Managing Product Development Projects", *Production and Operations Management*, Vol. 16 No. 2, pp. 217–240.

Choma, J., Zaina, L.A.M. and Beraldo, D. (2016), "UserX Story: Incorporating UX Aspects into User Stories Elaboration", In: Kurosu, M. (Ed.), *Human-computer interaction: 18th international conference, HCI International 2016*, Toronto, ON, Canada, July 17-22, 2016 proceedings, Lecture notes in computer science Information systems and applications, incl. Internet/Web, and HCI, Vol. 9731, Springer, Cham, Heidelberg, pp. 131–140.

Cohn, M. (2010), "Succeeding with agile: Software development using Scrum", *The Addison-Wesley signature series A Mike Cohn signature book*, Addison-Wesley, Upper Saddle River, NJ.

Dullemond, K., van Gameren, B. and van Solingen, R. (2009), "How Technological Support Can Enable Advantages of Agile Software Development in a GSE Setting", In: *Fourth IEEE International Conference on Global Software Engineering*, 2009: ICGSE 2009 ; 13 - 16 July 2009, Limerick, Ireland ; proceedings ; [including workshop papers], Limerick, Ireland, IEEE, Piscataway, NJ, pp. 143–152.

Gloger, B. and Margetich, J. (2014), "Das Scrum-Prinzip: Agile Organisationen aufbauen und gestalten", *Schäffer-Poeschel, Stuttgart*.

Highsmith, J. and Cockburn, A. (2001), "Agile software development. The business of innovation", *Computer*, Vol. 34 No. 9, pp. 120–127.

Klein, T.P. (2016), *Agiles Engineering im Maschinen- und Anlagenbau*, Dissertation Technische Universität München, Forschungsberichte IWB, Vol. 323, Utz, München.

Komus, A., Kuberg, M., Atinc, C., Franner, L., Friedrich, F., Lang, T., Makarova, A., Reimer, D. and Pabst, J. (2014), "Status Quo Agile 2014", [online] available at: http://www.status-quo-agile.de/

Lindemann, U. (2009), *Methodische Entwicklung technischer Produkte*, 3rd ed., Springer, Berlin.

Reichwald, R. and Piller, F. (2009), *Interaktive Wertschöpfung: Open Innovation, Individualisierung und neue Formen der Arbeitsteilung*, 2nd ed., Gabler Verlag, Wiesbaden.

Schmidt, T.S. and Paetzold, K. (2016), "Agilität als Alternative zu traditionellen Standards in der Entwicklung physischer Produkte: Chancen und Herausforderungen", In: Krause, D., Paetzold, K. and Wartzack, S. (Eds.), *Design for X: Beiträge zum 27*. DfX-Symposium Oktober 2016, TuTech, Hamburg, pp. 255–267.

Sommer, A.F., Slavensky, A., Thuy Nguyen, V., Steger-Jensen, K. and Dukovska-Popovska, I. (2013), "Scrum integration in stage-gate models for collaborative product development — A case study of three industrial manufacturers", In: *IEEE International Conference on Industrial Engineering and Engineering Management (IEEM)*, 2013, Bangkok, IEEE, Piscataway, pp. 1278–1282.

Sun, W., Marakas, G. and Aguirre-Urreta, M. (2016), "The Effectiveness of Pair Programming. Software Professionals' Perceptions", *IEEE Software*, Vol. 33 No. 4, pp. 72–79.

VMCD (1997), *Entwicklungsstandard für IT-Systeme des Bundes No. 251*, [online] Available at: http://www.v-modell.iabg.de/au251htm/index.htm

Webster, M. (2016), The Merriam-Webster dictionary, Merriam-Webster, Springfield.

ACKNOWLEDGMENTS

In this section the authors want to thank the experts for taking part in the expert interviews. Both specialists are employees of the German automotive company Knorr-Bremse AG.