# Identification of Inconsistencies in Agile CPS Engineering with Formula Student

Albert Albers

Karlsruhe Institute of Technology, Kaiserstrasse 10, 76131 Karlsruhe, Germany.
E-mail: albert.albers@kit.edu

Anne Koziolek

Karlsruhe Institute of Technology, Kaiserstrasse 10, 76131 Karlsruhe, Germany.
E-mail: anne.koziolek@kit.edu

Thomas Alexander Völk*

Karlsruhe Institute of Technology, Kaiserstrasse 10, 76131 Karlsruhe, Germany.
E-mail: thomas.voelk@kit.edu

Monika Klippert

Karlsruhe Institute of Technology, Kaiserstrasse 10, 76131 Karlsruhe, Germany.
E-mail: monika.klippert@kit.edu

Felix Pfaff

Karlsruhe Institute of Technology, Kaiserstrasse 10, 76131 Karlsruhe, Germany.
E-mail: felix.pfaff@kit.edu

Robert Stolpmann

Karlsruhe Institute of Technology, Kaiserstrasse 10, 76131 Karlsruhe, Germany.
E-mail: ugoix@student.kit.edu

Stefan Eric Schwarz

Karlsruhe Institute of Technology, Kaiserstrasse 10, 76131 Karlsruhe, Germany.
E-mail: stefan.schwarz@kit.edu

## 1 Problem

Everyone who collaboratively worked on shared files has experienced the effects of inconsistencies: Changes made by one contributor may not appear in another contributor's working file. Merging different versions of the same file into one consistent state becomes challenging. This is already frustrating in an easy and less complex setting. It becomes even more relevant in the development of complex systems containing multi-domain models and artifacts in different contexts. A prime example of complex systems are Cyber-Physical Systems (CPS). CPS integrate computation and physical processes (Lee 2010). Their character as enabling technology has a huge innovation potential for future applications (acatech - Deutsche Akademie der Technikwissenschaften, ed. 2011: 11). However, CPS engineering also presents challenges. These problems arise not only from the technological complexity of CPS but also from the interdisciplinary nature of development projects: Experts from different domains work together on a single System in Development (SiD), each following different engineering cycles. Each domain is working on different models representing distinct views of the SiD. Unfortunately, these models are often not interoperable (Feichtinger et al., 2022). Inconsistencies emerge when working with different views of CPS, whether in separate models or tools. Keeping the involved models consistent requires specific coordination activities and thus high manual effort. To address this, we propose two future visions for interdisciplinary inconsistency management: 1) Demand-orientated situational process kits to quickly handle inconsistency situations and 2) (semi-)automatic consistency preservation to reduce the manual effort of keeping models consistent. This requires an in-depth understanding of the inconsistencies occurring in practical applications. While formal research in software engineering has contributed to our understanding, we lack insights from practitioners across different engineering domains. Therefore, we aim to explore practitioners' perspectives on inconsistency in everyday agile engineering practice, identify contradictions in designing processes for consistent CPS engineering, and propose a strategy to create an empirical dataset to address the current lack of knowledge needed to understand current (in-)consistency management in practice.

## 2 Current Understanding

**Agile Product Development**

Agile approaches are increasingly used in companies' processes to be able to react to unexpected and expected changes in the dynamic context of product development (Atzberger et al., 2020). This is particularly important when dealing with problems arising from inconsistencies. Workflows for agile collaboration between developers in Software Engineering, mainly for working with the distributed version control solution git, for example in WunderFlow, were proposed (Rios et al., 2022). However, several problems arise when introducing agile approaches in a non-software environment due to major differences in the development areas (Albers et al., 2019a). Atzberger et al. identified different challenges in the agile development of physical products, such as the common understanding of agile development in physical products, time restrictions in manufacturing processes, physical restrictions, and appropriate incrementation of the product. (Atzberger et al., 2020).

Industry practitioners have reported that agility forms a challenge when facing the development of cyber-physical production systems, especially regarding experience with agile methods. Agile verification and validation of models changed during the development and management of different life cycles of system elements for each engineering domain. (Feichtinger et al., 2022)
Hybrid approaches can minimize these challenges in mechatronic systems (Boehm and Turner, 2005). Implementing such hybrid approaches requires the assessment of planning stability of individual process elements and different process levels which can be achieved with the ASD – Agile Systems Design (Albers et al., 2019b). The introduction of agile approaches in existing process models in practice is usually limited to individual teams, departments, and projects, as common process models are reaching their limits in the development of CPS.
Nevertheless, an unchanged adaption of software development approaches is not possible to redesign non-software development processes (Schmidt, Paetzold, 2017), (Dumitrescu et al., 2021). Agile methods deployment in CPS engineering has not been demonstrated in industrial studies yet (Ahmad, 2020).

**ASE – Advanced System Engineering**
The increasing complexity and dynamic interrelations of technical systems with other (socio)-technical systems within systems-of-systems lead to new challenges in the development of these systems (Dumitrescu et al., 2021). This applies to CPS. ASE combines the strengths of the three previously unconnected areas of *advanced systems* (AS), *systems engineering* (SE), and *advanced engineering* (AE) to address these challenges.

- AS describe the market performance of tomorrow, which is mapped via stronger integration of services and internet- and platform-based services (Piller, 2013).
- SE is a transdisciplinary and integrative approach to enable the successful realization, use, and decommissioning of systems, using systems principles and concepts, as well as scientific, technological, and management methods. (Walden et al., 2015)
- AE describes the support provided to the developer by intelligent tools, such as artificial intelligence, digital twins, or so-called virtualized single underlying models (V-SUM) (Stark and Damerau, 2019), (Klare et al., 2021).

**Change Management in CPS**
In the understanding of CPS as advanced systems (AS), changes play a central role in mitigating deviations between the current state of development and the desired development goals (Lindemann, 2016).
The development of CPS can be described with the model of SGE – System Generation Engineering. The model is based on two hypotheses: Firstly, every development of a new system generation $G_{i=n}$ relies on a reference system $R_{i=n}$ where n stands for the generation in development that will be launched on the market next. This reference system comprises elements from existing or planned socio-technical systems (referred to as reference system elements, RSE). Secondly, each new system generation is developed with three types of variation: carryover variation (CV), attribute variation (AV), and principle variation (PV). (Albers et al., 2015; Albers and Rapp, 2022)

Changes result in different increments, iterations, or maturity levels in the development of a new system generation. These increments can be described as engineering generations $E_{i=n,\ j=m}$ of the system generation $G_{i=n}$, also developed with three different types of variation, where j=m can be defined as the next engineering generation according to the project plan. (Albers et al. 2016)

Although change management processes are already established in the industrial practice, CPS engineering challenges and shorter innovation cycles demand new approaches for engineering change management (Gausemeier, 2014). Methods supporting aspects of change propagation are mostly based on Model-Based Systems Engineering (MBSE) or Engineering Change Management (Weilkiens, 2006; Walden et al., 2015; Altner et al. 2022; Martin et al. 2022). The usage and acceptance of such model-driven approaches highly depend on the development situation: exploratory development in early stages differs from the development of mature features in later stages (Kuhn et al., 2012).

The vision implicated by these developments is close to the tight and "flow-like" connection of development activities enabled by automation in software engineering which is called "continuous software engineering" (Fitzgerald and Stol, 2017). Approaches for continuous model-based engineering of CPS are already discussed in the context of CPS engineering, e. g. by Giaimo et al. (Giaimo et al., 2020) or Karsai et al. (Karsai et al., 2008). Following this ideal, the next step of future development processes would require a (semi-)automated feedback of information into models and views of a CPS. This perception of *continuous engineering* for CPS is related to the notion of a digital twin: a model of an object, an evolving set of data relating to the object, and a means of dynamically updating or adjusting the model following the data (Wright and Davidson, 2020). This would require consistency checks and consistency preservation rules in a runtime scenario. The first approaches in this direction address these scenarios for specific applications in mechatronic system development (Gausemeier et al., 2009) or for software systems with simplifying assumptions (Mazkatli, 2020).

## 3 Research Aim and Research Questions

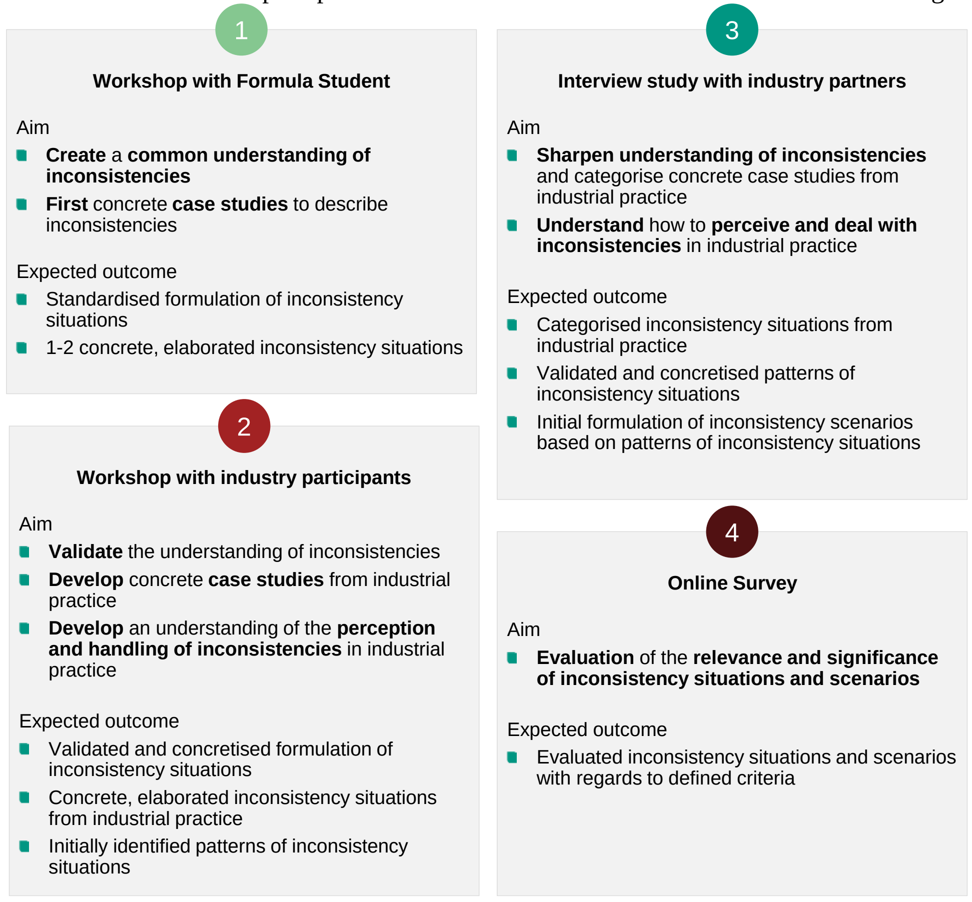
As changes serve as the primary driver of progress in CPS engineering, inconsistencies will occur repeatedly during various stages of the engineering process, particularly in the incremental work within agile development environments. To ensure seamless collaboration between engineers from different disciplines and design spaces with different perspectives on CPS, it is necessary to analyze the inconsistencies that occur. This leads us to the following three research questions, that we want to answer in this paper:

- What is the current understanding of inconsistency and inconsistency situations in the development of CPS?
- Where do inconsistency situations arise in the agile development processes of CPS?
- How can inconsistency situations in the development of CPS be characterized?

The data set of inconsistency situations presented in this paper will be expanded in further studies. The results of those studies are necessary to cluster the inconsistency situations into inconsistency scenarios with different levels of agility that can be formalized to investigate systematic consistency preservation in CPS engineering.

## 4 Research Design

Our current understanding of inconsistency occurring in CPS development is limited to a few examples from literature (e. g. Gausemeier et al., 2009). To describe inconsistencies arising in the agile development processes of CPS and to describe them in formalized situations and scenarios, an in-depth understanding of inconsistencies from practitioners' experience is necessary. To answer the research questions, workshops, and interviews are conducted with multiple partners. We consider a multi-level research design:



1

**Workshop with Formula Student**

Aim
- **Create** a **common understanding of inconsistencies**
- **First** concrete **case studies** to describe inconsistencies

Expected outcome
- Standardised formulation of inconsistency situations
- 1-2 concrete, elaborated inconsistency situations

2

**Workshop with industry participants**

Aim
- **Validate** the understanding of inconsistencies
- **Develop** concrete **case studies** from industrial practice
- **Develop** an understanding of the **perception and handling of inconsistencies** in industrial practice

Expected outcome
- Validated and concretised formulation of inconsistency situations
- Concrete, elaborated inconsistency situations from industrial practice
- Initially identified patterns of inconsistency situations

3

**Interview study with industry partners**

Aim
- **Sharpen understanding of inconsistencies** and categorise concrete case studies from industrial practice
- **Understand** how to **perceive and deal with inconsistencies** in industrial practice

Expected outcome
- Categorised inconsistency situations from industrial practice
- Validated and concretised patterns of inconsistency situations
- Initial formulation of inconsistency scenarios based on patterns of inconsistency situations

4

**Online Survey**

Aim
- **Evaluation** of the **relevance and significance of inconsistency situations and scenarios**

Expected outcome
- Evaluated inconsistency situations and scenarios with regards to defined criteria

**Figure 1: Multi-level research design to identify inconsistencies in agile CPS engineering**

We conducted a workshop with Formula Student to achieve a common understanding of inconsistency based on a proposed standardized description of inconsistency situations (*the focus of this paper*). Then we would conduct workshops with experts from different organizations to challenge our understanding created in the first workshop, to collect more descriptions of inconsistency situations in industrial practice, and to discover how

inconsistency is perceived by industry professionals. Afterward, we would conduct interviews to understand how representative our created situation sample will be and a survey to collect a widely spread perspective on recurring patterns discovered in our situation samples.
This paper presents the results of the first workshop. We decided to collaborate with a Formula Student racing team. Our main goal was to better understand inconsistencies and to compare research and practice. The Formula Student is an international construction competition addressing the engineering, manufacturing, and marketing of a race car. The Formula Student racing team contains participants from multiple engineering domains. They develop a new generation of racing cars every year, which sets them apart from car manufacturers in terms of development cycles. The team is self-organized, has over 10 years of experience in participation in the event, and won several Formula Student events. The team is split into different sub-teams covering different aspects of the development of the new racing car generation. Each team member works self-organized and self-responsible. Topics concerning the overall project are discussed in weekly management meetings. The development of the race car can be described as agile. The developed race car is a battery electrical vehicle (BEV) able to operate autonomously. Therefore, the race car can be classified as a CPS.

We used the "Think-Pair-Share"-methodology (Macke et al., 2016) to conduct the workshop: The workshop contained three separate phases during which different tasks had to be performed by the workshop participants:

- **Think**: The participants had a clear task to do in individual work. After this phase, the participants were able to present their results to a partner. In our workshop, the participants had to describe their personal experience with inconsistency, its causes, and its effects occurring in their work at the formula student team based on a given example organized in a template. They had 35 minutes to come up with at least two individual descriptions of inconsistencies, that they experienced.
- **Pair**: The participants paired up and discussed their results. Every participant should be able to present the results of their partners. In our workshop, the partners also discussed solution strategies to resolve the inconsistencies. They had 45 minutes to compare their results, discuss their situations, and come up with solution strategies.
- **Share**: The pairs of participants presented their results to the plenum. The plenum had 60 minutes to discuss their results.

The workshop was held with 10 participants from different positions within the Formula Student racing team: They covered a range from single-domain engineering team members (e.g., aerodynamics, electric drivetrain, monocoque) to interdisciplinary team leaders. All participants had at least one season of experience in developing race cars. As not all team members from the Formula Student racing team were German-speaking participants, the workshop was held in English as their main communication language. The following materials were used: A presentation, a document for the synchronized documentation, a detailed task description containing step-by-step instructions, a template, pens, and Post-its in different colors and sizes as well as recording equipment. In addition, an example of an inconsistency situation was provided, using a template, that contains the following information: interviewee, role in the company, experience in the field, background, inconsistency situation, effects, and solution strategies. Two exemplary inconsistency situations are shown in Figure 2 and Figure 3 using the template as a visualization as described above.

## 5 Results

During the workshop, the participants came up with a total of 18 inconsistency situations. Table 1 summarizes eight exemplary inconsistency situations, which are anonymized and adjusted to the table format.

**Table 1: Exemplary inconsistency situations as a result of a workshop with a Formula Student racing team**

| *No.* | *Icon* | *Name of inconsistency* | *Description* |
|---|---|---|---|
| 1 | | Uncertainty errors through the transfer of models | Inconsistency occurred in modeling (e.g., Computer-Aided Design and Computational Fluid Dynamics models), and the transferability of modeling results to reality |
| 2 | ↑↑ | Parallel development of different models | Inconsistency occurred between the results of the different sub-teams, as individual modules were developed independently of each other |
| 3 | | Independent development | Sub-teams developed subsystems independently of each other, sometimes without any knowledge of the dependencies of other subsystems. |
| 4 | | Knowledge loss | Inconsistency occurred due to loss of knowledge (e.g., specific parameters in a model). Especially in a constantly changing engineering environment such as long-term student-driven development projects. |
| 5 | | Unstructured communication | Inconsistency occurred due to unstructured communication between team members and information discrepancies. |
| 6 | | Differences in targets and requirements documentation | Inconsistency occurred due to different target and requirements documentation between the developer and manufacturer from another branch (e.g., no data sheets available). |
| 7 | | Incorrect models | Inconsistency occurred due to errors in the creation of a model. The new model is based on an incorrect or outdated model. Hence the solution does not meet the requirements and target specifications. |
| 8 | | Different milestones | Inconsistency occurred due to unsynchronized development cycles and schedules (e.g., sub-team A is still in development and wants to make changes, that affect other sub-teams. Whereas, sub-team B already reached the next milestone). |

*Further findings*: The participants did not always describe the effects of an inconsistency situation and/or possible solution strategies. Those are either not known or not consistently integrated into the engineering process. Two examples are given in Sec. 6 of this paper.

## 6 Discussion

The results shown in Table 1 can be discussed on multiple levels regarding different aspects of the research goal of the paper.

**RQ1: What is the current understanding of inconsistency and inconsistency situations in the development of CPS?**

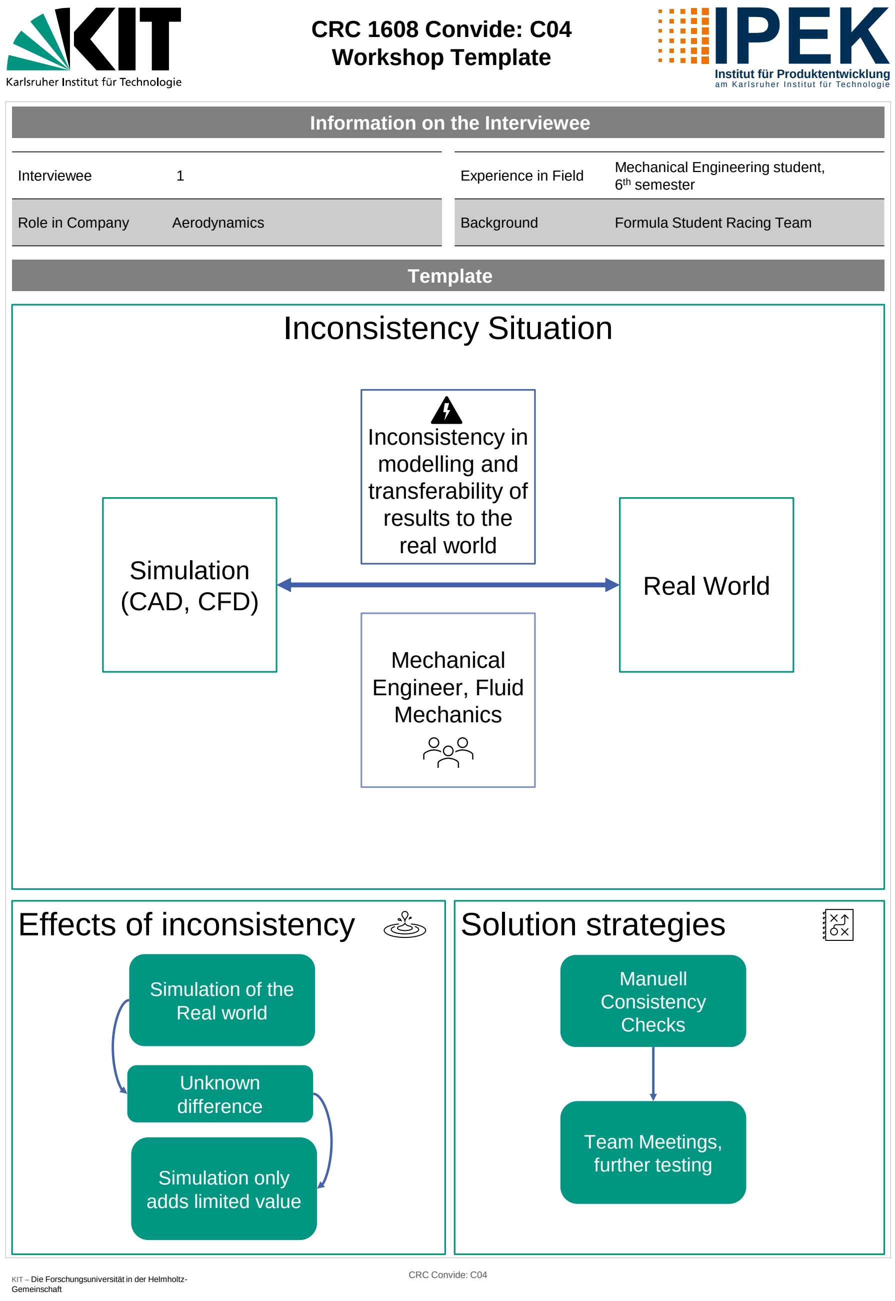
The quality and the fit of the given descriptions in the workshop differ by a wide range. According to the participants, this is mainly due to the unclear understanding of inconsistency in engineering practices. This results in different described situations focusing on inconsistency situations due to inconsistency of models to models, inconsistency of models to reality, and inconsistency in modeling approaches. Besides that, situations not addressing concrete inconsistencies, but rather overall problems of interdisciplinary collaboration are described. This shows us that “inconsistency” as a common term in engineering is currently not framed concretely and allows a broad range of interpretations.

Besides the described problems of a unified understanding of inconsistency, formalization issues are detected, especially during the processing of the given data. Some participants told us that the chosen initial example for an inconsistency out of an already known context was helping. The situation was presented in a PowerPoint presentation and a pre-formalized structure on a blank template. This structure was sparsely used as a reference, but most of the participants used plain text answers to describe their respective inconsistency situations.

When processing the participants' responses, we propose an approach to formalize the description of the inconsistency situations that arise. By arranging the situations like those presented in Table 1, the given answers can be compared and discussed. This requires much post-processing of the collected data with possible knowledge loss and biased interpretation of the collected data by researchers. Therefore, more formalized support in identifying inconsistency situations for practitioners is needed for future research activities. We suggest the creation of a template, describing the exact type of data required and supporting participants in giving comparable descriptions of their identified inconsistency.

**RQ2: Where do inconsistency situations arise in the agile development processes of CPS?**

The inconsistency situations in Table 1 will not individually be discussed in this section. We want to provide a description of two selected examples directly addressing inconsistency in different aspects of agile development (Figure 2 and Figure 3). Those visualizations of the two exemplary inconsistency situations are a summary of the workshop results proposed by the researchers. The structure of the template follows the workshop methodology (see Sec. 3). As mentioned in the answer to RQ1 this visualization has some improvement potential.



**CRC 1608 Convide: C04**
**Workshop Template**

**Information on the Interviewee**

| | | | |
|---|---|---|---|
| Interviewee | 1 | Experience in Field | Mechanical Engineering student, 6th semester |
| Role in Company | Aerodynamics | Background | Formula Student Racing Team |

**Template**

Inconsistency Situation

Simulation (CAD, CFD)

Inconsistency in modelling and transferability of results to the real world

Real World

Mechanical Engineer, Fluid Mechanics

Effects of inconsistency

Simulation of the Real world

Unknown difference

Simulation only adds limited value

Solution strategies

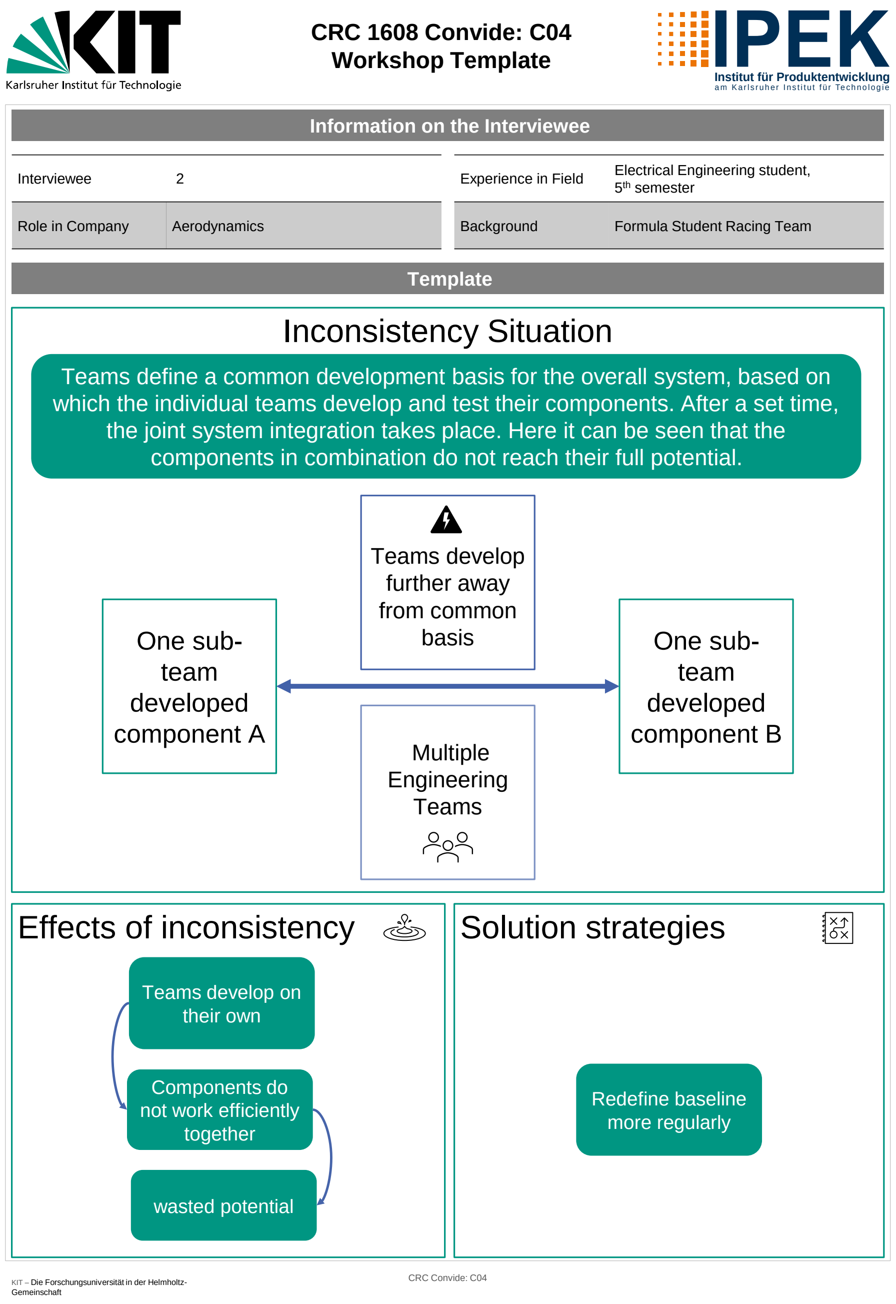
Manuell Consistency Checks

Team Meetings, further testing

KIT – Die Forschungsuniversität in der Helmholtz-Gemeinschaft

CRC Convide: C04

**Figure 2: Description of Inconsistency Situation No. 1 “Uncertainty errors through transfer of models” on the Basis of the Workshop Results**



CRC 1608 Convide: C04
Workshop Template

**Information on the Interviewee**

| | | | |
|---|---|---|---|
| Interviewee | 2 | Experience in Field | Electrical Engineering student, 5th semester |
| Role in Company | Aerodynamics | Background | Formula Student Racing Team |

**Template**

## Inconsistency Situation

Teams define a common development basis for the overall system, based on which the individual teams develop and test their components. After a set time, the joint system integration takes place. Here it can be seen that the components in combination do not reach their full potential.

Teams develop further away from common basis
One sub-team developed component A
One sub-team developed component B
Multiple Engineering Teams

## Effects of inconsistency

Teams develop on their own
Components do not work efficiently together
wasted potential

## Solution strategies

KIT – Die Forschungsuniversität in der Helmholtz-Gemeinschaft

CRC Convide: C04

**Figure 3: Description of Inconsistency Situation No. 2 “Parallel development of different models” on the Basis of the Workshop Results**

**RQ3: How can inconsistency situations in the development of CPS be characterized?**
Based on the results of the workshop inconsistencies always occurred between artefacts. By analyzing the inconsistencies, three initial patterns can be identified:

- Inconsistency inbetween one model
- Inconsistency between two or more models
- Inconsistency between a model and reality (e.g., physical artifacts, model boundaries, and limitations)

These patterns are preliminary, as further research is needed to validate or adjust the described patterns. The identified categories are based on the development experience of students that might be different from professional engineering activities. Nevertheless, the categories enable a better understanding of different situation categories. They can potentially help future participants to better understand and describe the inconsistencies in their daily work in agile environments.

## 7 Contribution and Practical Implications

We want to connect experts from different engineering backgrounds in a common empirical-based understanding of inconsistencies in CPS engineering. By doing so we aim to enable cross-industry awareness and different solution approaches. With this network and empirical evidence of the need for processes for consistent CPS development, we want to enable future research in this area.

The given study is a first attempt to describe inconsistencies occurring in an agile development environment. It identifies a broad range of understanding of the term inconsistency in a CPS development environment. The study shows that even in small development projects, several inconsistencies arise that potentially hinder the development of CPS. Inconsistencies mainly occur between two or more models or between models and the reality.

Once a variety of concrete inconsistency situations from practice have been found, key factors will be derived from these. These key factors make it possible to characterize each inconsistency situation. The next step is to define inconsistency scenarios. A scenario comprises several situations and abstracts them into formalized patterns for further use. To enable the formalization of inconsistency scenarios for further application, i. e. in automatic consistency checking, further formalization of inconsistency situations is needed. The empirical dataset to enable inconsistency scenario deriving is not mature enough to give a valid explanation of occurring situations. The given dataset serves as an explorative fundament for future investigation.

In the long term, requirements from the industry for the implementation of methods, processes, and tools shall be identified. We want to achieve a formalization approach for inconsistency scenarios to preserve consistency through a specification language. This is necessary to easily maintain consistency in the increasingly complex development processes. Furthermore, it enables the creation of automated consistency preservation rules that allow simple consistency management. This should be based on the models from continuous software engineering.

## 8 Acknowledgements

This research is funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – SFB 1608 – 501798263.

## 9 Feedback

We hope for input and feedback on the following questions:

- Which understanding do other researchers have of inconsistency and inconsistency situations?
- How could we identify inconsistencies and inconsistency situations in industry practice and research?
- How could we describe inconsistencies and inconsistency situations systematically?
- Which patterns in inconsistency situations do other researchers know?

## References and Notes

Ahmad, M.O., 2020. Agile Methods and Cyber-Physical Systems Development—A Review with Preliminary Analysis. In *Big Data and Security: First International Conference, ICBDS 2019, Nanjing, China, December 20–22, 2019, Revised Selected Papers 1* (pp. 274-285). Springer Singapore.

Albers, A. and Rapp, S. (2022): "Model of SGE: System Generation Engineering as Basis for Structured Planning and Management of Development", In: Krause, D. and Heyden, E. (Eds.): *Design Methodology for Future Products. Data Driven, Agile and Flexible*, Springer International Publishing, pp. 27–46, online version: https://link.springer.com/book/10.1007/978-3-030-78368-6.

Albers, A., Bursac, N. and Wintergerst, E., 2015. Product generation development–importance and challenges from a design research perspective. *New developments in mechanics and mechanical engineering*, pp.16-21.

Albers, A., Heimicke, J., Müller, J. and Spadinger, M., 2019a. Agility and its features in mechatronic system development: A systematic literature review. In *ISPIM Conference Proceedings* (pp. 1-13). The International Society for Professional Innovation Management (ISPIM).

Albers, A., Heimicke, J., Spadinger, M., Reiss, N., Breitschuh, J., Richter, T., Bursac, N. and Marthaler, F., 2019b. A systematic approach to situation-adequate mechatronic system development by ASD-Agile Systems Design. *Procedia CIRP*, *84*, pp.1015-1022.

Albers, A., Haug, F., Heitger, N., Arslan, M., Rapp, S., Bursac, N. (2016) Produktgenerationsentwicklung - Praxisbedarf und Fallbeispiel in der automobilen Produktentwicklung. In: *Vorausschau und Technologieplanung*. 12. Symposium für Vorausschau und Technologieplanung. 8. und 9. Dec 2016 Berlin. Heinz Nixdorf Institut, Paderborn, pp 227–242.

Moritz Altner, Hans Redinger, Benjamin Valeh, Eder Kevin, Jonas Neckenich, Simon Rapp, Roland Winter, Albert Albers, Improving engineering change management by introducing a standardised description for engineering changes for the automotive wiring harness, *Procedia CIRP*, Volume 109, 2022, Pages 538-543, ISSN 2212-8271, https://doi.org/10.1016/j.procir.2022.05.291.

Atzberger, A., Nicklas, S.J., Schrof, J., Weiss, S. and Paetzold, K., 2020. Agile Entwicklung physischer Produkte. Eine Studie zum aktuellen Stand in der industriellen Praxis. Neubiberg: Universitätsbibliothek der Universität der Bundeswehr München.

Boehm, B. and Turner, R., 2005. Management challenges to implementing agile processes in traditional development organizations. *IEEE software*, *22*(5), pp.30-39.

Dumitrescu, R., Albers, A., Riedel, O., Stark, R. and Gausemeier, J., 2021. Engineering in Deutschland-Status quo in Wirtschaft und Wissenschaft: Ein Beitrag zum Advanced Systems Engineering. *Fraunhofer IEM, Paderborn*, pp.88-92.

Feichtinger, K., Meixner, K., Rinker, F., Koren, I., Eichelberger, H., Heinemann, T., Holtmann, J., Konersmann, M., Michael, J., Neumann, E.M. and Pfeiffer, J., 2022, September. Industry voices on software engineering challenges in cyber-physical production systems engineering. In *2022 IEEE 27th International conference on emerging technologies and factory automation (ETFA)* (pp. 1-8). IEEE.

Fitzgerald, B. and Stol, K.J., 2017. Continuous software engineering: A roadmap and agenda. *Journal of Systems and Software*, *123*, pp.176-189.

Gausemeier, J., Schäfer, W., Greenyer, J., Kahl, S., Pook, S. and Rieke, J., 2009. Management of cross-domain model consistency during the development of advanced mechatronic systems. In *DS 58-6: Proceedings of ICED 09, the 17th International Conference on Engineering Design, Vol. 6, Design Methods and Tools (pt. 2), Palo Alto, CA, USA, 24.-27.08. 2009*.

Gausemeier, J., 2014. *Strategische Planung und integrative Entwicklung der technischen Systeme von morgen*. Paderborn: Schöningh.

Giaimo, F. and Berger, C., 2020. Continuous experimentation for automotive software on the example of a heavy commercial vehicle in daily operation. In *Software Architecture: 14th European Conference, ECSA 2020, L'Aquila, Italy, September 14–18, 2020, Proceedings 14* (pp. 73-88). Springer International Publishing.

Karsai, G. and Sztipanovits, J., 2008. Model-integrated development of cyber-physical systems. In *Software Technologies for Embedded and Ubiquitous Systems: 6th IFIP WG 10.2 International Workshop, SEUS 2008, Anacarpi, Capri Island, Italy, October 1-3, 2008 Proceedings 6* (pp. 46-54). Springer Berlin Heidelberg.

Klare, H., Kramer, M.E., Langhammer, M., Werle, D., Burger, E. and Reussner, R., 2021. Enabling consistency in view-based system development—the vitruvius approach. *Journal of Systems and Software*, *171*, p.110815.

Kuhn, A., Murphy, G.C. and Thompson, C.A., 2012. An exploratory study of forces and frictions affecting large-scale model-driven development. In *Model Driven Engineering Languages and Systems: 15th International Conference, MODELS 2012, Innsbruck, Austria, September 30–October 5, 2012. Proceedings 15* (pp. 352-367). Springer Berlin Heidelberg.

Lindemann, U. ed., 2016. *Handbuch Produktentwicklung*. Carl Hanser Verlag GmbH Co KG.

Macke, G., Hanke, U. & Raether, W. (2016) *Kompetenzorientierte Hochschuldidaktik: lehren – vortragen – prüfen – beraten*. Weinhem Basel, Beltz.

Martin, A., Kasper, J., Pfeifer, S., Mandel, C., Rapp, S. & Albers, A. (2022). Advanced Engineering Change Impact Approach (AECIA) – Towards a model-based approach for a continuous Engineering Change Management. In *8th IEEE International Symposium on Systems Engineering*. ISSE 2022 conference proceedings. Piscataway: IEEE.

Mazkatli, M., Monschein, D., Grohmann, J. and Koziolek, A., 2020, March. Incremental calibration of architectural performance models with parametric dependencies. In *2020 IEEE International Conference on Software Architecture (ICSA)* (pp. 23-34). IEEE.

Piller, F., 2013. *Mass customization: ein wettbewerbsstrategisches Konzept im Informationszeitalter*. Springer-Verlag.

Rios, J.C.C., Embury, S.M. and Eraslan, S., 2022. A unifying framework for the systematic analysis of git workflows. *Information and Software Technology*, *145*, p.106811.

Schmidt, T.S. and Paetzold, K., 2017. Challenges of agile development: A cause-and-effect analysis. In *Complex Systems Design & Management: Proceedings of the Seventh International Conference on Complex Systems Design & Management, CSD&M Paris 2016* (pp. 237-237). Springer International Publishing.

Stark, R. and Damerau, T., 2019. Digital Twin. In *CIRP Encyclopedia of Production Engineering* (pp. 1-8). Springer Berlin Heidelberg

Walden, D.D., Roedler, G.J. and Forsberg, K., 2015, October. INCOSE systems engineering handbook version 4: updating the reference for practitioners. In *INCOSE International Symposium* (Vol. 25, No. 1, pp. 678-686).

Weilkiens, T., 2006. *Systems engineering mit SysML/UML: modellierung, analyse, design*. dpunkt-Verlag.

Wright, L. and Davidson, S., 2020. How to tell the difference between a model and a digital twin. *Advanced Modeling and Simulation in Engineering Sciences*, *7*(1), pp.1-13.

## Image References

<a href="https://www.flaticon.com/free-icons/simulation" title="simulation icons">Simulation icons created by syafii5758 - Flaticon</a>

<a href="https://www.flaticon.com/free-icons/parallel" title="parallel icons">Parallel icons created by Sergei Kokota - Flaticon</a>

<a href="https://www.flaticon.com/free-icons/weakness" title="weakness icons">Weakness icons created by Freepik - Flaticon</a>

<a href="https://www.flaticon.com/free-icons/brain" title="brain icons">Brain icons created by Freepik - Flaticon</a>

<a href="https://www.flaticon.com/free-icons/chaos" title="chaos icons">Chaos icons created by Freepik - Flaticon</a>

<a href="https://www.flaticon.com/free-icons/requirement" title="requirement icons">Requirement icons created by Soremba - Flaticon</a>

<a href="https://www.flaticon.com/free-icons/incorrect" title="incorrect icons">Incorrect icons created by juicy_fish - Flaticon</a>

<a href="https://www.flaticon.com/free-icons/timeline" title="timeline icons">Timeline icons created by Freepik - Flaticon</a>