Model Inconsistencies and Solution Approaches to Maintain Consistency in Model-based Systems Engineering

Thomas Schumacher, David Inkermann
Institut fuer Maschinenwesen
Technische Universitaet Clausthal
38678 Clausthal-Zellerfeld, Germany
Email: {schumacher, inkermann}@imw.tu-clausthal.de

***Abstract*—An effective interdisciplinary engineering process of modern systems requires linked system and domain models. This linkage results frequently in model inconsistencies, for instance conflicting information or different data types. In this paper, different types of inconsistencies, established solution approaches and challenges in Model-based Systems Engineering are derived to introduce an approach for maintaining model consistency. Therefore, a structured literature analysis is performed including 47 research papers. Based on the analysis five challenge types, ten inconsistency types and four approaches to maintain model consistency were identified. To foster model consistency this paper introduces the use of heterogeneous models. This approach combines different solution approaches to overcome crucial inconsistency and challenge types.**

Keywords-Model-based Systems Engineering; Model inconsistencies; Heterogeneuos models.

I. INTRODUCTION AND PROBLEM DESCRIPTION

Model-based Systems Engineering (MBSE) is a common approach to manage the complexity when developing modern system. MBSE aims at the introduction of a comprehensive system model instead of document-based approaches as a central information and communication basis [1]. A comprehensive system model integrates system requirements, behavioural and structural system definitions as well as design constraints or test results and enables the stakeholder to study the system from different viewpoints. Stakeholder, like systems engineers or mechanical engineers, viewpoints differ based on their concerns or interests on the system. Since an effective engineering requires intensive collaboration of different engineering domains and their specific viewpoints have to be integrated. However, this integration frequently causes model inconsistencies [2]. Thereby, model inconsistencies can be understood as logical contradiction or irrational existence among facts, artefacts or concepts [3]. Inconsistent models can lead to extraordinary increase of costs and development time [2] or can have serious consequences, like failed missions in aeronautics [4].

A. Linkage of System Model and Domain Models

MBSE including the different engineering tasks and a central system model that is typically modelled using the Systems Modeling Language (SysML), provides various benefits. Core advantages that are frequently reported are improved collaboration of different engineering domains, enhanced information consistency or increased system understanding, in comparison to traditional document-based System Engineering [5]. To achieve these advantages the use of an appropriate modelling method, modelling language and modelling tool are required [6]. In addition to this domain-independent models domain-specific models are needed, e.g., to describe the subsystems regarding geometry and spatial structure. Thereby an efficient engineering process of modern systems requires a linkage between these different model types (system and domain models), for instance by using information exchange procedures [1]. However, this information and data exchange is often facing different challenges, like incompatible data structures, since various models and tools are used and even when the information can be exchanged, the consistency of this model information still remains a challenge [7].

B. Research Objective

Objective of this contribution is to maintain model consistency during the development of modern systems. Therefore, a literature study was performed in order to identify challenges in MBSE based on model inconsistencies, different types of model inconsistencies and solution approaches to maintain consistency. Moreover, this contribution introduces the use of heterogeneous models within the engineering process as a combination of different solution approaches to ensure model consistency and to overcome crucial inconsistency and challenge types.

Based on the objective within this contribution the following research questions will be addressed:

- Which different types of model inconsistencies can be distinguished?
- Which different solution approaches are established in order to maintain model consistency?
- Which challenges in MBSE, and which model inconsistency types can be addressed by the application of heterogeneous models?

The paper is organized as follows: Section II explains what model inconsistencies in MBSE are and what solution approaches to maintain model consistency are available. Within Section III, the results of the study are presented, showing challenges within MBSE based on model inconsistencies, different types of model inconsistencies and solution approaches. The application of heterogeneous models and

 ISBN: 978-1-68558-037-7

their added value to maintain model consistency will be discussed in Section IV. The paper is concluded by a summary and an outlook on further research.

II. State of the Art

The following section introduces the term model inconsistency and presents approaches to handle inconsistencies or to maintain model consistency.

A. Model Inconsistencies within MBSE

Inconsistencies between different models is a key challenge in MBSE [8]–[10]. Basically, inconsistency can be understood as logical contradiction or irrational existence among facts, artefacts or concepts [7][11]. In the context of this paper, we understand model inconsistency as the violation of domain-specific or domain-independent engineering rules or constraints, as stated by Vogel-Heuser et al. [7]. There are many examples for model inconsistencies, like violation of well-formedness rules, inconsistencies in redundant information, mismatches between model and test data and not following heuristics or guidelines [3][11]. Based on the high variety of inconsistencies, this paper will introduce different types of model inconsistencies that can occur within MBSE. Herzig et al. [11] investigated the fundamentals of model consistency and concluded that it is impossible to maintain model consistency during the development of complex technical systems. Therefore, in the following section different approaches for management of model inconsistencies will be presented, based on literature.

B. Approaches for Inconsistency Management

One of the major challenges of a classical document-based Systems Engineering is to ensure that system specification does not contain any contradicting information [11], which represents one typical kind of inconsistency. To handle this challenge MBSE introduces the idea of using a comprehensive system model, as cross-linked set of computer-interpretable models [1], to specify the system and thereby increasing the level of formalism [11]. The application of a formalized system model supports maintaining consistency in early design phases when for example the overall system architecture will be developed. With progressive development the domain-specific engineering domains are required to develop the detailed system design. Typically, domains applying domain-specific engineering approaches and models, like state-machine diagrams in the software domain or CAD-models in the mechanical domain. Accordingly, especially at the interface between system model and domain models inconsistencies can occur. To reduce the amount of model inconsistencies typically three different approaches for inconsistency management can be distinguished: proof-theory-based, rule-based and synchronization-based approaches [10][12]. Following these different approaches will be explained.

1) Proof-theory-based approaches

The application of a proof-theory-based approach for inconsistency management was initially proposed by Finkelstein et al. [13] for model-driven software engineering. They transform multi-view software models (e.g., class and sequence diagrams) to a first-order logic to identify inconsistencies using automated theorem prover and domain-specific rules (specified as a temporal logic) [10][12]. Core idea of this approach is the transformation of graphical representations (diagrammatic models [14]) in more formal, mathematical terms, in which inconsistencies can be identified [12].

2) Rule-based approaches

In a rule-based approach rules are used to describe the conditions that a model must satisfy. Thereby these conditions can used as positive or negative constraints. Satisfying positive constraints indicate that the model can be considered as consistent. Satisfying negative constraints conversely indicates model inconsistencies [12]. For this approach different applications are available in literature, like [15][16]. For mechatronic systems, Feldmann et al. [8] propose the Resource Description Framework (RDF) as a concrete representational formalism for models. By applying query languages, like SPARQL, different inconsistency types can be identified [12].

3) Synchronization-based approaches

The target of synchronization-based approaches is to synchronize semantically related models [12]. Therefore, model transformation is required, to provide linkage of model elements between different model types [10]. Model transformation can be distinguished into two categories: (1) use of customized modelling languages and (2) use of transformation rules [10]. Customizing modelling languages aims at an appropriate linkage of different model types. In literature SysML is often customized, by creating domain-specific profiles. An example is the profile SysML4Modelica which was created to link models based on the complementary languages SysML and Modelica [17]. The second opportunity to synchronize models is the use of transformation rules, like Triple Graph Grammars (TGGs) [18]. These rules ensure the linkage between different model elements. If a model element is changed (e.g., changed property of a model element) the change will propagate to all related model elements [12]. Overall, model transformation can ensure interoperability between different modelling languages and thereby enable domain-specific development teams to apply their known modelling languages and tools [10].

In this section, an overview about established approaches for inconsistency management was provided rely on either proof-theoretic, rule-based or model synchronization (using model transformation) approaches. To generate a more detailed understanding about different kind of inconsistencies and solution approaches within the following section different types of inconsistencies and solutions approaches will be derived and allocated.

III. Challenges in MBSE, Types of Model Inconsistencies and Solution Approaches

This section will introduce challenges in MBSE, different types of model inconsistencies and solution approaches to maintain model consistency based on an exploratory study. Figure 1 illustrates the procedure of the study. Based on a

 ISBN: 978-1-68558-037-7

defined searching string 47 publication were identified within Scopus. The first ten publications were used to define types (categories) for challenges, inconsistencies, and solution approaches. Therefore, all mentioned challenges, inconsistencies and solutions approaches in regards to MBSE were identified and afterwards categorized into different types. The categorizing was performed by thematical and namely merging. For example, all inconsistencies regarding information or data were combined into one category or the solution type model execution contains all types of model simulation. Afterwards the remaining publications were reviewed by considering the defined categories. Finally, the review results were evaluated. We consider the amount of 47 publications for the literature study as sufficient to identify crucial MBSE challenges, inconsistency types and solution approaches. A higher number of publications will probably change the enumerations and the presented ranking, but we assume that the identified categories will still the same. Sub-sections A, B and C present the result of the literature study. In sub-section D, solution approaches will be allocated to inconsistency types.

A. *Types of Challenges in MBSE*

The first intent of the literature study was the identification of challenges within MBSE based on model inconsistencies. Figure 2 presents five challenges which are frequently reported in literature. Thereby, the challenge ***maintaining consistency*** between different models can be considered as the main challenge in MBSE. Further presented challenges are ***interoperability of modelling tools***, ***visualize specific model views***, ***management of inconsistencies*** and ***maintaining traceability***.

These challenges particularly occur due to the collaboration of multiple domains during the development of modern systems [10]. Different domains develop the system from different viewpoints because they have different interests on the system. Consequently, inconsistencies may occur in the course of the entire engineering process, like architecture definition, domain-specific implementation, integration and verification and validation. The major target is to maintain model consistency during the entire engineering process or if required to identify and solve any model inconsistencies at the time of creation in order to minimize costs and development time.

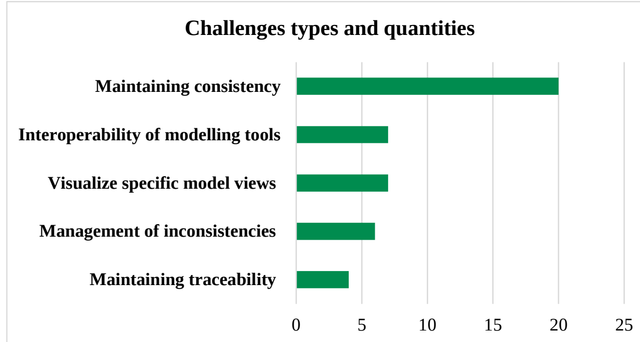

Figure 2. Identified challenge types and quantities.

The following sub-section presents different types of inconsistencies in order to avoid their creation or to locate and maintain them during the engineering process.

B. *Types of Model Inconsistencies*

Based on the performed exploratory study ten different types of model inconsistencies were identified. Figure 3 presents each type and how often they were determined during the study. By evaluating the study result, it can be concluded that six out of ten inconsistency types are more frequently stated in literature (at least five enumerations). In the following each type of inconsistency is described.

Data and information inconsistency could be determined most during the study (18 times). This inconsistency type reflects model elements with conflicting information or different data types. For instance, the system properties described in a CAD-model contradicts the initial defined properties in a system model. ***Representation inconsistency*** describes inconsistencies within the model representation. Typically, this kind of inconsistencies is caused by application of multiple models and different views and perspectives. ***Refinement inconsistency*** is typically caused by modelling of elements on different abstraction level. ***Viewpoint inconsistency*** emerges by overlapping viewpoints. The definition of viewpoints is based on various factors, like concerns of interest or responsibility.

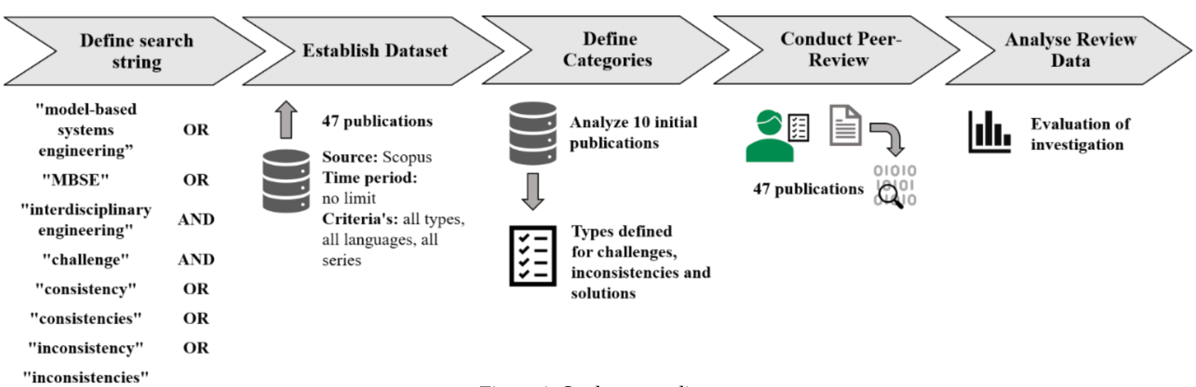

Figure 1. Study proceeding.

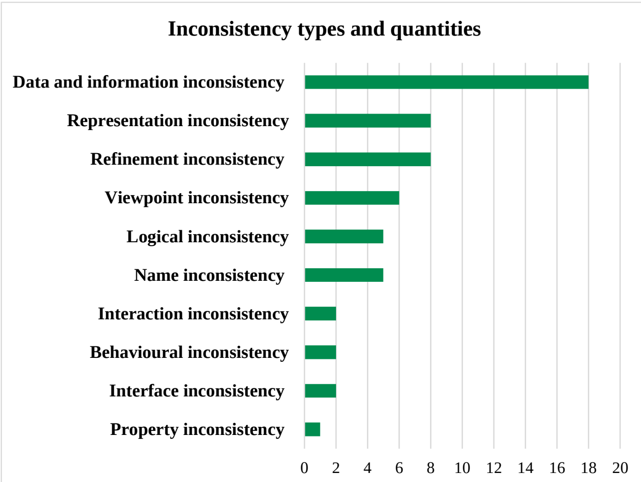

Figure 3. Identified inconsistency types and quantities.

Logical inconsistency summarizes all inconsistencies which are caused by applying different or informal modelling languages. Not following a given formal semantics and syntax can lead to logical contradictions within the model. ***Name inconsistency*** can arise when model elements have same or unconventional name, or the naming conventions are not followed. Based on literature four additional inconsistency types with sparsely amount of enumerations can be determined. ***Interaction inconsistency*** describes the execution of model operations which violates interaction constraints, like the order of model operations. ***Behavioural inconsistency*** contains an unexpected behaviour of model elements. ***Interface inconsistency*** arises when model elements describing interfaces have conflicting values, terminologies, or schemas. ***Property inconsistency*** occurs when model elements contradict constraints regarding element properties or values.

C. Types of Solution Approaches

The papers out of the study were also analysed regarding approaches to handle model consistencies. These solutions can be classified into four overall categories: ***model execution***, ***tool interoperability and data exchange***, ***model abstraction*** and ***model formalization***. Thereby *model execution* and *tool interoperability and data exchange* are the most frequently reported approaches, see Figure 4.

Model execution contains all approaches regarding actively checking for model inconsistency by model simulation, for instance a model element expects an energy flow but has only material or informational relations, or the use of inconsistency pattern. For safety relevant systems model assessments applying simulation-based fault injection approaches to identify failures in the system design and behaviour are established. Moreover, all approaches explained in Section II.B are included in this category. ***Interoperability of modelling tools*** are focusing on establishing standardized interface specifications, like Open Services for Lifecycle Collaboration (QSLC) or Functional Mock-up Interface (FMI). Also, part of this solution type are all approaches considering *data exchange* among modelling tools. Therefore, many contributions propose the application of universal data formats and

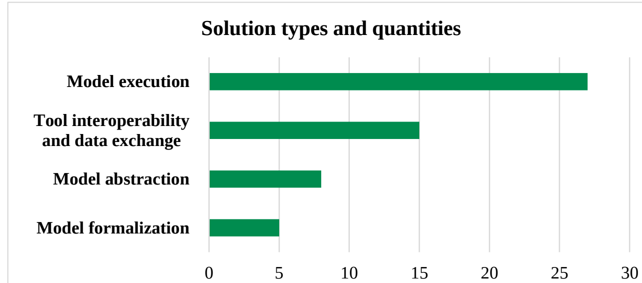

Figure 4. Identified solution types and quantities.

schemas, like Automation Markup Language (AML) or Standard for the Exchange of Product Data (STEP). A further approach is the use of meta-models and ontologies to link model elements on a more abstract level. These are classified into the solution type ***model abstraction***. The use of (semi)formal modelling language is an additional approach to reduce model inconsistencies, this goes along with a higher ***model formalization*** and reuse of knowledge, like model pattern. These two types were distinguished because a model formalization does not always go along with an abstraction and vice versa. The following sub-section will allocate these solution approaches to the identified inconsistency types.

D. Allocation of Solution Approaches to Inconsistency Types

To achieve the overall objective to maintain model consistency it is important to locate and solve the different model inconsistencies. Therefore, Table I contains an allocation between the identified inconsistency types and solution types.

TABLE I. ALLOCATION OF SOLUTION AND INCONSISTENCY TYPES

Inconsistency type	Solution type
Data and information inconsistency	Model execution
	Tool interoperability and data exchange
Representation inconsistency	Model execution
Refinement inconsistency	Model abstraction
Viewpoint inconsistency	Tool interoperability and data exchange
Logical inconsistency	Model formalization
Name inconsistency	Model execution
	Tool interoperability and data exchange
Interaction inconsistency	Model execution
Behavioural inconsistency	Model execution
Interface inconsistency	Model execution
	Tool interoperability and data exchange
Property inconsistency	Model execution
	Tool interoperability and data exchange

This allocation gives advice which solution types can support managing the different inconsistency types.

In general, it can be conducted that caused by the high variety of model inconsistencies a consistent model-based engineering process requires different solution approaches in parallel to ensure model consistency.

 ISBN: 978-1-68558-037-7

Therefore, we introduce the application of heterogeneous models as an approach combining different solution approaches to maintain model consistency.

IV. APPLICATION OF HETEROGENEOUS MODELS TO MAINTAIN MODEL CONSISTENCY

The establishment of heterogeneous models in MBSE supports by overcoming the following identified challenges: ***maintain consistency***, ***interoperability of modelling tools***, ***visualize specific model view*** and ***maintaining traceability***. Thereby following inconsistency types will be addressed: ***data and information inconsistency*** due to the linkage of model elements based on data structure. ***Representation inconsistency*** due to the integration of different views and perspectives into one model and ***refinement*** and ***logical inconsistencies*** due to the integration of model elements with different abstraction or different semantics and syntax into one presentation. This is made possible due to the combination of following solution types: ***model synchronization*** as part of model execution, ***tool interoperability and data exchange***, ***model abstraction*** as well ***model formalization***.

Following the application of heterogeneous models will be explained by visualization of heterogeneous models and presentation of a technical interface concept to link different model types. Thereby, we are focusing on model consistency between SysML and CAD-models, which represent domain-independent and domain-specific models.

Heterogeneous models offer the possibility to integrate different sub-models or model elements into one model presentation [19]. As an example, Jansen presents a mechatronic leg as a heterogeneous model, which integrates three-dimensional objects, two-dimensional substitute models including their relations and additional information about the context of the system, like assembly space restrictions [20]. Thus, heterogeneous models can be applied to integrate model elements with different abstraction and formalization level, like SysML-elements as domain- independent models and CAD-elements as domain-specific models, into one model [21]. Figure 5 presents a mock-up of a heterogeneous model, which combines CAD- and stereotyped SysML model elements. This model can support for example by system architecture definition, due to the integration of behaviour descriptions (yellow oval) and their allocation to the physical system structure and the combination of interface descriptions based on SysML-elements (ports and interface blocks) and their relations to the physical system elements.

Premise for consistent heterogeneous models are linked model elements. Therefore, the data structure of each model element needs to be investigated and linked. Figure 6 presents a concept for linking SysML- and CAD model elements based on the universal data types XML and STEP [22].

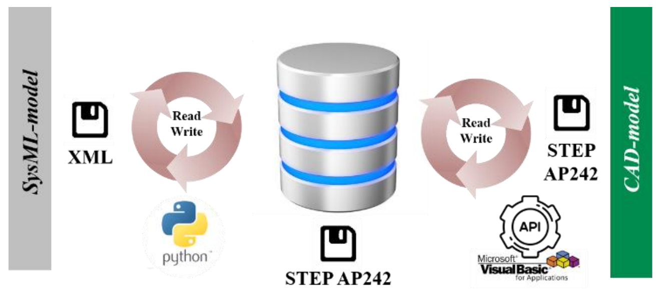

Figure 6. Technical interface concept [22].

Key elements of these interface are two applications for data transfer. First a Python-based application which interprets the data between XML and STEP files and second a VBA-based application programming interface (API), which controls the data in- and output into the CAD-tool. More details about the technical interface and their application can be seen in [22].

It can be conducted that the application of heterogeneous models can be a great support by maintaining model consistency due to the combination of different solution approaches. Thereby, the number of model inconsistencies based on different inconsistency types can be reduced.

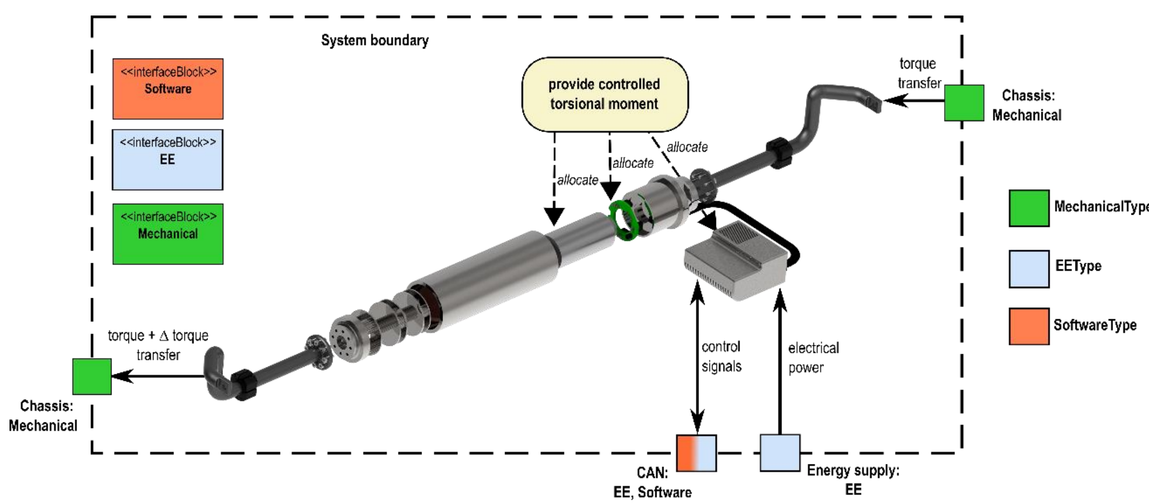

Figure 5. Visualization of a heterogeneous model, based on [21].

V. Conclusion

This paper depicts the need of linked system and domain models in interdisciplinary engineering. This linkage is often a common cause for model inconsistencies. Therefore, within this paper different challenges in MBSE, types of model inconsistencies and established solution approaches to maintain model consistency were determined. Based on a performed literature study, ten different inconsistency types, four solution types and five challenges were identified. Following the solution types were allocated to inconsistency types to locate and solve potential model inconsistencies. Furthermore, this contribution introduces heterogeneous models as an approach for maintaining model consistency. Heterogeneous models based on a linked data structure combine different solution types to maintain model consistency and offer the opportunity to create meaningful models for specific engineering activities. With these models, the occurrence of different inconsistency types can be prevented and thus substantial challenges within MBSE supported. Future research is focusing on the evolution and application of heterogeneous models based on a linked data structure. Therefore, different modelling tools will be investigated to integrate CAD- and SysML-model elements into one model.

Acknowledgment

Funded by the Lower Saxony Ministry of Science and Culture under grant number ZN3493 within the Lower Saxony "Vorab" of the Volkswagen Foundation and supported by the Center for Digital Innovations (ZDIN).

References

[1] S. Friedenthal, A. Moore, and R. Steiner, A practical guide to SysML: The systems modeling language (The MK/OMG Press). Waltham, MA: Elsevier/Morgan Kaufmann, 2015.

[2] G. Spanoudakis and A. Zisman, "Inconsistency Management in Software Engineering: Survey and Open Research Issues," in Handbook of Software Engineering and Knowledge Engineering, 2001, pp. 329–380.

[3] S. J. I. Herzig and C. J. J. Paredis, "A conceptual basis for inconsistency management in model-based systems engineering," Procedia CIRP, vol. 21, pp. 52–57, 2014, doi: 10.1016/j.procir.2014.03.192.

[4] B. Nuseibeh, "Ariane 5: Who Dunnit?," IEEE Softw., vol. 14, no. 3, pp. 15–16, 1997, doi: 10.1109/MS.1997.589224.

[5] K. Henderson and A. Salado, "Value and benefits of model-based systems engineering (MBSE): Evidence from the literature," Systems Engineering, vol. 24, no. 1, pp. 51–66, 2021, doi: 10.1002/sys.21566.

[6] L. Delligatti, SysML distilled: A brief guide to the systems modeling language. Upper Saddle River, NJ: Addison-Wesley, 2014.

[7] B. Vogel-Heuser et al., "Challenges of interdisciplinary development of Cyber-Physical Production Systems," At-Automatisierungstechnik, vol. 67, no. 6, pp. 445–454, 2019, doi: 10.1515/auto-2018-0144.

[8] S. Feldmann et al., "Towards effective management of inconsistencies in model-based engineering of automated production systems," IFAC-PapersOnLine, vol. 28, no. 3, pp. 916–923, 2015, doi: 10.1016/j.ifacol.2015.06.200.

[9] A. M. Madni and M. Sievers, "Model-based systems engineering: Motivation, current status, and research opportunities," Systems Engineering, vol. 21, no. 3, pp. 172–190, 2018, doi: 10.1002/sys.21438.

[10] A. Berriche, F. Mhenni, A. Mlika, and J.-Y. Choley, "Towards model synchronization for consistency management of mechatronic systems," Applied Sciences (Switzerland), vol. 10, no. 10, pp. 1–20, 2020, doi: 10.3390/app10103577.

[11] S. J. I. Herzig, A. Qamar, A. Reichwein, and C. J. J. Paredis, "A conceptual framework for consistency management in model-based systems engineering," Proceedings of the ASME Design Engineering Technical Conference, vol. 2, pp. 1329–1339, 2011, doi: 10.1115/DETC2011-47924.

[12] S. Feldmann et al., "A comparison of inconsistency management approaches using a mechatronic manufacturing system design case study," in 2015 IEEE International Conference on Automation Science and Engineering (CASE), Gothenburg, Sweden, 2015, pp. 158–165, doi: 10.1109/CoASE.2015.7294055.

[13] A. Finkelstein, G. Gabbay, A. Hunter, J. Kramer, and B. Nuseibeh, "Inconsistency Handling in Multi-Perspective Specifications," in 1993 -Proceedings of the 4th European Software Engineering Conference, pp. 84–99.

[14] "Diagrammatic Models," in Intuition in Science and Mathematics (Mathematics Education Library), Dordrecht: Kluwer Academic Publishers, 2002, pp. 154–166.

[15] T. Mens, R. van der Straeten, and M. D'Hondt, "Detecting and Resolving Model Inconsistencies Using Transformation Dependency Analysis," in Model Driven Engineering Languages and Systems (Lecture Notes in Computer Science), D. Hutchison et al., Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 200–214.

[16] A. Egyed, "Automatically Detecting and Tracking Inconsistencies in Software Design Models," IIEEE Trans. Software Eng., vol. 37, no. 2, pp. 188–204, 2011, doi: 10.1109/TSE.2010.38.

[17] C. J. Paredis et al., "An Overview of the SysML-Modelica Transformation Specification," INCOSE International Symposium, vol. 20, no. 1, pp. 709–722, 2010, doi: 10.1002/j.2334-5837.2010.tb01099.x.

[18] F. Klar, A. Königs, and A. Schürr, "Model Transformation in the Large," in Proceedings of the 6th joint meeting of the European software engineering conference, Dubrovnik Croatia, I. Crnkovic and A. Bertolino, Eds., 2007, pp. 285–294, doi: 10.1145/1287624.1287664.

[19] S. Jansen and E. G. Welp, "A Heterogeneous Modelling Approach for Domain Allocation in Mechatronics," Guidelines for a Decision Support Method Adapted to NPD Processes, 2007.

[20] S. Jansen, A methodology for model-based partitioning of mechatronic systems. Aachen: Shaker, 2007.

[21] T. Schumacher and D. Inkermann, "Heterogeneous models to Support Interdisciplinary Engineering - Mapping Model Elements of SysML and CAD," Procedia CIRP, vol. 109, pp. 653–658, 2022, doi: 10.1016/j.procir.2022.05.309.

[22] T. Schumacher, D. Kaczmarek, D. Inkermann, and A. Lohrengel, "Fostering Model Consistency in Interdisciplinary Engineering by Linking SysML and CAD-Models," in 2022 IEEE International Symposium on Systems Engineering (ISSE), Vienna, Austria, 2022, pp. 1–7, doi: 10.1109/ISSE54508.2022.10005327.