Boundary Artifacts in Interdisciplinary Engineering: Conceptualizing Rosetta Stones Between Domains

Thomas Alexander Voelk
IPEK – Institute of Product Engineering
Karlsruhe Institute of Technology (KIT)
Karlsruhe, Germany
thomas.voelk@kit.edu

Razieh Dehghani
KASTEL - Institute of Information Security and Dependability
Karlsruhe Institute of Technology (KIT)
Karlsruhe, Germany

Henriette Färber
KASTEL - Institute of Information Security and Dependability
Karlsruhe Institute of Technology (KIT)
Karlsruhe, Germany

Stefan Eric Schwarz
IPEK – Institute of Product Engineering
Karlsruhe Institute of Technology (KIT)
Karlsruhe, Germany

Romain Pascual
MICS, CentraleSupélec, Université Paris-Saclay, 9 rue Joliot Curie, 91190
Gif-sur-Yvette, France

Lars Gesmann
IPEK – Institute of Product Engineering
Karlsruhe Institute of Technology (KIT)
Karlsruhe, Germany

Arne Lange
KASTEL - Institute of Information Security and Dependability
Karlsruhe Institute of Technology (KIT)
Karlsruhe, Germany

Leow Zi Le
KASTEL - Institute of Information Security and Dependability
Karlsruhe Institute of Technology (KIT)
Karlsruhe, Germany

Terru Stübinger
KASTEL - Institute of Information Security and Dependability
Karlsruhe Institute of Technology (KIT)
Karlsruhe, Germany

Melih Sah Dogru
IPEK – Institute of Product Engineering
Karlsruhe Institute of Technology (KIT)
Karlsruhe, Germany

Mattias Ulbrich
KASTEL - Institute of Information Security and Dependability
Karlsruhe Institute of Technology (KIT)
Karlsruhe, Germany

Tobias Düser
IPEK – Institute of Product Engineering
Karlsruhe Institute of Technology (KIT)
Karlsruhe, Germany

Anne Koziolek
KASTEL - Institute of Information Security and Dependability
Karlsruhe Institute of Technology (KIT)
Karlsruhe, Germany

Albert Albers
IPEK – Institute of Product Engineering
Karlsruhe Institute of Technology (KIT)
Karlsruhe, Germany

***Abstract*—In interdisciplinary engineering, collaboration often breaks down due to fragmented models, inconsistent terminology, and incompatible representations. Translating meaning between domains remains a significant challenge. While model-based systems engineering (MBSE) and other single underlying model (SUM) approaches offer structural support, they often lack the flexibility and scalability needed for dynamic, cross-domain work. A common bottleneck is the modeler. Translating domain-specific language into shared models is time-consuming and often only consistent with the actual project the moment its models are created. To enable MBSE (and other modeling approaches), we need "Rosetta Stones": touchpoints between different epistemic domains combined in a single entity. This paper introduces boundary artifacts as such entities. They are intentionally designed to combine syntactic structure with semantic meaning. Unlike loosely defined boundary objects, boundary artifacts are purpose-built to support communication, coordination, and integration across disciplines. They can appear as parameters, diagrams, models, or documents, each containing overarching semantic overlaps between domains. The paper presents key hypotheses: boundary artifacts can help align viewpoints, support traceable decisions, and serve as anchors for shared meaning. We also discuss important features such as flexibility, robustness, and lifespan, and how these affect their usefulness in collaborative design. By formalizing the concept of boundary artifacts, this work offers a foundation for more scalable, resilient, and human-centered design practices. It supports both methodological development and practical use in complex engineering environments.**

Keywords—Boundary objects, boundary artifacts, interdisciplinary engineering, consistency management, interdisciplinary communication

I. Introduction

In 1799, the Rosetta Stone was rediscovered. Inscribed on this stone is the same decree written in Ancient Greek, Demotic, and Hieroglyphic text [1]. The decryption of the Rosetta Stone provided the hidden key to understanding the lost semantics of the hieroglyphs, thereby enabling us to understand inscriptions in that ancient language.

Collaboration in engineering still seeks such "Rosetta Stones": Even though people might find themselves in good communication environments, they tend to misunderstand one another. In a recent study, it was found that even in a nearly optimal communication environment in a cross-disciplinary setting, one third of the information is still lost [2]. One of the most pressing problems is misaligned language, specifically, misunderstanding relates primarily to unfamiliar terms or differences in meaning of the same terms [2]. While approaches like Systems Engineering (SE) should address this issue, their implementation in practice is still hindered by the amount of work needed to model a complex system or the lack of usability from a practitioner's point of view [3] .

In future, "advanced systems" will become more and more complex in development and operation, as their degree of

autonomy, networking, interaction, and smart services also become more and more relevant [3], e.g. in cyber-physical systems (CPS). CPS are the integration of computation and physical processes [4], expanding mechatronic systems through the possibilities of the "Internet of Things" [5]. This integration of the cyber domain and physical domains, such as mechanical or electrical engineering, will bring an increase in development complexity, cost pressure, and regulatory challenges [3] already present as the challenge of aligning development paradigms in agility in the development of physical products [6].

The development of a system like a CPS requires ongoing change. Albers et al. [7] introduce this process as an interplay between a system of objectives (containing requirements, boundary conditions, or specifications), an operation system (performing actions based on a solution space and the current state of knowledge), and a system of objects (containing both physical and virtual results from a synthesis activity, e.g. models, software code, prototypes, testbenches, etc.). In general, such elements of the system of objects can be called "development results" as they are emerging from development activities.

Aligning a set of development results (e.g. models, software code, prototypes, testbenches, in general everything that is the result of an engineering synthesis) in a way that those results do not contradict each other means creating a consistent state of the overall system [8]. While easy to define, its opposite in its different tones is actually the regular state of engineering projects, meaning that resolving inconsistency to reach a consistent state is one of the major tasks of engineering practitioners [9]. While tool-based problems and traceability issues are commonly found in inconsistency situations from industry, communication gaps and cross-domain coordination are still prominent nontrivial issues in inconsistency situations [10]. This results in a fundamental conflict: How can practitioners and modelers collaborate if they lack semantic knowledge (see [11]) about the system in development, the engineering practices, or even the epistemic foundations of a given engineering domain?

This work proposes a way to overcome this barrier by not trying to overcome boundaries themselves, but by emphasizing the boundaries of domains and their possible overlap in domain-crossing development results. Such results, which we later refer to as “boundary artifacts”, should enable us to identify the connecting pieces of information between domains, thereby facilitating the development of consistency preservation and reparation tools and methods. To clarify the concept of boundary artifacts, the following research questions are answered:

1. How are domain-crossing development results understood currently in their boundary function?
2. How can an understanding for domain-crossing development results be generated to make consistency preservation more accessible for practitioners and modelers?
3. What implications does the generated understanding of domain-overarching development results have for the implementation of consistency preservation, interdisciplinary development projects and cross-generational CPS development?

To answer the first research question, we conducted a literature review on concepts related to domain-crossing development results on boundaries. To answer the second question, section 3 introduces a concept for boundary artifacts building on previous theoretical research from formal methods, reasoning about a concept for boundary artifacts and their role in engineering processes. Based on those theoretical concepts, we want to introduce implications of boundary artifacts for for consistency management in CPS development practice. Therefore, we rely on literature and exemplary workshops in a collaborative research center on CPS engineering.

II. State of Research on Boundary Crossing Development Results

In literature, many different approaches for describing “boundary crossing development results” are established, but neither of them follows a precise definition or consistent wording. We therefore focus on the properties, the impact on consistency management, as well as on examples of boundary crossing development results given in literature rather than sticking to the unprecise wording commonly found in literature.

A. Properties of Boundary Crossing Development Results

Based on the understanding of Star [12] boundary crossing development results are plastic enough to be adaptable across multiple viewpoints, yet maintain continuity of identity. Boundary objects inhabit several communities of practice and satisfy informational requirements of each of them [13]. Objects on boundaries can fulfill two functions: They can be ambigous to foster healthy long-term group interactions, or they can be clarifying to reduce conflict to ideas [14]

Boundary crossing development results usually hold a high criticality. This high criticality refers to its significant role in maintaining system functionality. Studies highlight that failure of such critical boundary objects/artifacts can lead to severe damage, unscheduled maintenance, and high costs [15]. Selecting which boundary crossing development result holds high criticality due to the parallel existence of multiple artifacts in a system. To support transparent decision-making, such artefacts must have an appropriate level of granularity, allowing stakeholders to assess their importance and impact more effectively.

B. *Impact on consistency management of boundary crossing development results*

Boundary crossing development results are used both for functional and political decision-making because they promote shared representations, transform design knowledge, mobilize for design action, and legitimize design knowledge [16]. Challenges in management of boundary crossing development results usually are mostly related to keep the information updated in large-scale systems including multiple stakeholders [15, 17, 18]. To address these issues, Wohlrab et al. [15] recommend using tools which help

visualize item relationships and support better decision-making based on the scale and complexity of boundary crossing development results.

Also, information dispersal is highlighted as a challenge [17, 19]. Inadequate information sharing, fragmentation of knowledge, and difficulty in capturing implicit knowledge hinder the creation of clear and comprehensive boundary crossing development results.

C. Examples of boundary crossing development results

In general, boundary crossing development results mostly can be clustered into the following categories: results clarifying the system of objectives (as in [7]), results defining the systems architecture and function, and results abstracting a system in development for validation. All of those results have in common, that they preserve domain-overarching knowledge or make domain-specific knowledge accessible for other development domains.

The system of objective is describable through e.g. product profiles as the modeling of a need situation through customer, user, and provider benefits [20], (top-level) requirements [15, 17] serving as a shared reference for product management, development, and testing or system specification [18].

The definition of architecture and function builds on the description of the system of objectives: Product profiles are used to define early-phase functions across multiple domains [21]. In several sources, the Reference Architecture Model (RAM) is identified as a boundary crossing development result [17, 19, 22]. It provides a "single common view" of the system, enabling stakeholders from diverse domains to understand and collaborate more effectively.

Validation is the core activity of product engineering [23]. Therefore, results enabling the shared understanding of validation are crucial. They are usually represented as visualizations and prototypes [18, 24, 25] used to improve "communication, learning, and decision-making throughout the design process" iteratively [26]. Mueller et al. [27] further distinguish virtual prototypes, which integrate embedded software with the physical environment and support virtual integration, enabling system improvements.

D. Conclusion for current understanding of boundary crossing development results

In general, literature provides a broad picture of boundary crossing development results, including objects as well as artifacts, both focusing on different aspects of their boundary function. However, the lack of a clear conceptualization of boundary crossing development results makes it non-trivial to adapt established terms and connected concepts for consistency preservation, thereby answering research question 1.

Considering the above challenges and features, a concept for domain-crossing development results should:

- adapt multiple domain-specific points of view for a single development result.
- be able to be constructed for clarification and communication purposes
- be generic enough to apply the transferred information to different engineering domains.
- include multiple forms of objects, like general models, but also digital and physical representations of the system in development.

III. Boundary Artifacts as a Concept for Domain-Overarching Consistency Preservation

Across disciplines, the terms “object”, “artifact”, and “model” as development results are used interchangeably in some instances yet sometimes denote distinct concepts. In the following, we distinguish between these terms to avoid ambiguity:

1. An object is any discrete, identifiable entity, either physical or immaterial. Objects may be naturally occurring or human-made and generally have no inherent purpose.
2. An artifact is an object that has been intentionally created to serve a specific function.

Our focus lies on artifacts because system development is inherently intentional: all relevant entities have a purpose within the development process. Thus, we use the term “artifact” exclusively when referring to development-relevant entities. This distinction becomes especially useful when looking at domain boundaries: A developer from one domain could see a set of objects from another domain but would not be able to understand their purpose or function without any fluency in the other domain’s knowledge. In this case, the respective engineer would not be able to identify those objects as artifacts, as he or she cannot infer their function. The distinction between artifacts and objects therefore can never be purely objective but is always dependent on the understanding of different stakeholders in a design process.

A model, in general, is an artifact whose purpose is to represent some original, like a real-world system or process, by selectively capturing aspects of it for a particular purpose. Following Stachowiak [28], a model is characterized by:

- Mapping: The model corresponds to its original.
- Reduction: The model captures only certain aspects of the original relevant to the intended use.
- Pragmatism: The model is created and used for a specific purpose within a particular context.

As models are artifacts, they may be physical entities (e.g., a test bench for a braking system) or immaterial (e.g., a CAD file). In the context of model-driven systems engineering (MBSE) or specifically in software-related model-driven development (MDD), models encapsulate specific purpose-oriented abstractions of systems. Thus, different models of the same system can be used to separate the concerns of different domain experts who are involved in the system development. The semantics, or meaning, of an artifact is assigned to it by human stakeholders.

In practice, model semantics may only be intuitive or informally stated. From a formal perspective, model

semantics are mappings from model syntax to a well-defined semantic domain, such as a mathematical structure or a set of system behaviors [29]. It is important to note that artifacts, including models, do not possess inherent semantics. The same model may be referenced by different stakeholders, who assign distinct semantics based on their needs. This is especially relevant in multi-domain development, as the same artifact can be assigned different semantics, even within a single project. While this is a powerful property, it also introduces a risk for inconsistency if not done conscious.

Pascual et al. [29] showed that the semantics for models over a given meta-model, i.e., models following the same syntactical rules, form a rich algebraic structure. Among the various elements of this structure, two play a specific role: the bottom and top element. The bottom element is the coarsest semantics giving the same value to all models. It is maximally abstract and loses any distinguishing information between models. Dually, the top element is the most fine-grained semantics, giving a unique semantic value to each model, preserving all model-specific distinctions. Every other element in the algebraic structure represents a semantics that lies somewhere between these extremes, retaining some distinctions but abstracting away others. The structure also contains ways to combine and compare different semantics for a given model. This means that different stakeholders who assign different semantics to the same model can find a common abstraction from their assumed model semantics. However, this common abstraction may be the trivial semantics that lose all information.

We assume that stakeholders from the same domain assign the same semantics to any given model, such that each domain can be characterized by the semantics its stakeholders assign to models. When stakeholders from different domains use the same artifact, their respective semantics may overlap - that is, share a non-trivial intersection. This overlap then allows for meaningful communication and comparison of the assumptions on models from these domains.

Based on this understanding of artifacts and semantics, we characterize boundary artifacts by the function that they should fulfill in the context of a development process: A boundary artifact is an artifact used by stakeholders from different domains to coordinate their work based on the overlapping semantics they share. It serves as a concrete point of reference that both parties can interpret in their own domain-specific way, while still retaining enough common ground to support meaningful communication. This idea is illustrated in Fig. 1.

It must be mentioned that following this logic, “boundary objects” are not possible: Only if an object has a purpose, there is the possibility to construct a semantic overlap. Therefore, only boundary artifacts are viable boundary crossing development results.

Such boundary artifacts are not limited to shared models. Some boundary artifacts may take unstructured or semi-structured forms, such as reports, emails, or meeting notes. Due to their flexibility and expressiveness, such artifacts may support communication between stakeholders, establishing a shared vocabulary across domains. On the other hand, boundary artifacts can be highly structured, e.g., as diagrams or test datasets, and could be used to (semi-) automatically manage consistency between other artifacts used within different domains. That’s why - in our opinion - boundary artifacts are a powerful and necessary tool to introduce consistency management in engineering environments, as they disclose the epistemic differences and homogenize the different understandings in different domains.

IV. Application of Boundary Artifact Concept in a Cross-Disciplinary Workshop on Brake System Development

In the collaborative research center (CRC) 1608 "Convide", one of the main objectives is to investigate on the development of CPS on the example of cyber-physical brake system development. Based on tools for consistency preservation, faster development cycles for CPS should be enabled [30]. A major initial challenge was the lack of a shared language between physical engineering practitioners and experts in formal methods and consistency specification. These domains typically operate in isolation, resulting in no

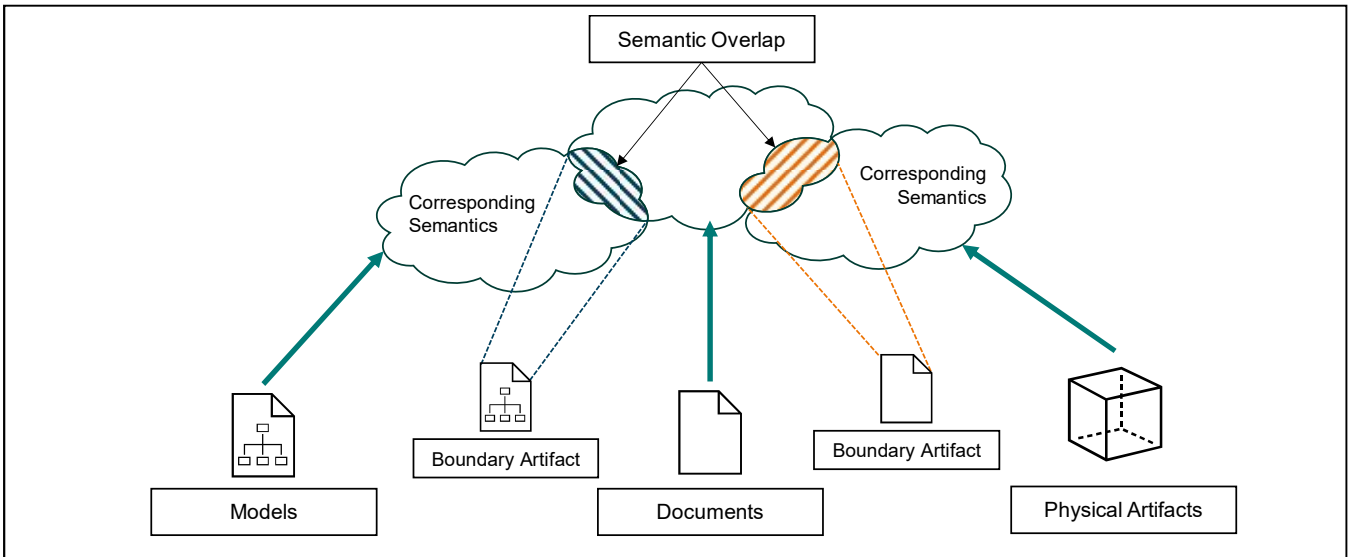


Fig. 1. Assessing meaning from artifacts to domains of discourse and the role of boundary artifacts in describing semantic overlap

clear semantic overlap. This gap hindered collaboration and made it difficult to initiate work on consistency preservation. Since no established methodology exists for identifying or creating boundary artifacts, a joint workshop was organized.

Participants included: two design researchers (methodologists), three brake system experts, one control systems expert, one software engineering expert, and four formal methods researchers. The workshop was not intended to produce boundary artifacts directly but to foster cross-disciplinary understanding within the research platform, avoiding bias in participant contributions.

Prior to the workshop, each participant was introduced to a curated set of inconsistency scenarios observed in industrial contexts, along with their associated patterns [10]. We identified such example patterns as first boundary artifacts, as they introduce a connection between inconsistency occurrence in engineering results and formal descriptions of the propagation of the inconsistency. Still, they were too general to work for specific consistency preservation, resulting more in the function of creating a common ground.

During the workshop, physical engineering experts presented a synthetic inconsistency scenario based on the CPS brake system. This included typical workflows, design artifacts (e.g., CAD models, simulation results), degrees of freedom in system design, boundary conditions, and domain-specific knowledge elements. While the given information was helpful for a basic understanding, only the concrete workflow containing an inconsistency situation would be classified as a boundary artifact, as it provided the linkage between physical engineering practice and formalization needs for analysis.

Throughout the presentation, participants from other domains were encouraged to ask clarifying questions, leading to deep discussions on technical details such as architecture of a hybrid vehicle, test bench configuration, and differences in brake system and electrical engine development. This interactive format helped uncover implicit assumptions and domain-specific constraints. The workshop resulted in a required list of artifacts needed for the development of formal method validation datasets providing a base for formalization approaches.

Although the workshop did not aim to generate boundary artifacts explicitly, it demonstrated that even informal, unstructured formats can reveal domain overlaps and interface requirements. Developing such exchange formats further to enable the identification, construction and maintenance of boundary artifacts could be a major contribution to enabling consistency preservation in practice.

V. Implications of the Concept of Boundary Artifacts for Consistency Management Application

A. Implications for coordination of interdisciplinary development projects

Wohlrab et al. [31] explain that managing boundary objects in general is important based on their case study on agility in automotive engineering. Boundary artifacts should be defined early on but refinable during the development of a system, while also relying on locally relevant artifacts i.e. relevant artifacts that only affect a selected group of developers like sub-teams, sub-domains, and others. The same applies to the introduced concept of boundary artifacts in consistency management: Right now, state of the art tools like Vitruvius only introduce static connections between models that are represented in views. If we understand boundary artifacts as such views, we must include mechanisms that allow for refinement and freezes of such objects that gradually adapt to the current development state a system is in, acting as an example for a prototype of the overall system in development as indicated by Subrahamanian et al. [25]. Therefore, the main challenge of consistency preservation approaches in practice will be the investigation of a single-domain change to other domains models as well as the (semi)-automatic deployment of changes to other models. Boundary artifacts can be a useful addition for such challenges, as indication of change is not only possible via KPIs, but discussable on shared "anchors" that can indicate if a change has critical impact on shared views of the system that cannot be easily overlooked from a single engineer or single domain point-of-view.

B. Implications for cross-generational CPS development

In the context of cross-generational CPS development, boundary artifacts function as anchors that persist beyond the lifecycle of a single system generation. Such anchors are critical in facilitating the integration of technological evolution, changing requirements, and evolving stakeholder constellations over time. This approach aligns with the principles of interdisciplinary collaboration, emphasizing the importance of early definition, gradual refinement, and, where needed, temporary freezes to preserve a stable reference state. Equally crucial are mechanisms for maintaining contextual knowledge and design rationale, particularly for applications spanning multiple generations. Without such continuity, valuable domain-specific insights may be lost between generations, leading to costly (re-) discovery efforts or the propagation of inconsistencies into subsequent designs.

By embedding historical context, generational history, and traceable links to previous system models, boundary artifacts can bridge temporal knowledge gaps and enable systematic impact analysis of design changes considering earlier constraints and decisions. This not only supports consistency preservation but also facilitates informed innovation by allowing prior architectures, prototypes, and proven solutions to be reevaluated against emerging requirements and technologies. In doing so, boundary artifacts provide stable yet adaptable semantic anchors, balancing the need for reuse with the capacity to evolve in the long-term development of complex CPS.

C. Implications for tool-based consistency checking and preservation

Especially in large-scale systems, the application of the concept of boundary artifacts only becomes viable with tool-based support for consistency preservation to keep the system updated. Such support in heterogenous development environments needs consistency specification: Such specification becomes necessary when models, and/or their parts, have a semantic space in common, i.e., have a common

representation in the "real world". The CRC 1608 “Convide” wants to address this need based on the Vitruvius tool [32], which uses the software engineering domain-specific “reactions” language, to specify consistency rules between models. The language resembles a mixture of transformation and action languages, where users can declare what changes have to happen to which model element to trigger a consistency repair action and restore overall consistency. This way, the tool never reaches a state of inconsistency (at least in theory). To this date, Vitruvius does not offer any support to detect semantic overlap or any facility to find/create boundary artefacts collaboratively.

Moving forward, tools as Vitruvius have to evolve to become user-friendly and usable for other domains besides Software Engineering. Those actions include algorithmic semantic overlap detection and collaborative workflows in Vitruvius to detect and generate boundary artefacts.

The former is planned as a feature in the future iterations of Vitruvius and is going to use techniques from the ontology, semantic web research community, where lexical and structural characteristics [33] are used to determine overlap between ontologies.

The latter requires more research and needs to explore the workflow of creating the boundary artefacts in a V-SUM, with a strong focus on human interaction in collaborative systems. Both ways boil down to the inspection of the language that is being used to describe parts of the system, and both could benefit from the use of Large-Language-Models.

VI. Conclusion

This paper introduces the concept of boundary artifacts to improve semantic alignment and consistency management in interdisciplinary CPS development. By formalizing boundary artifacts as development results with overlapping semantics across domains, the study provides a foundation for enabling meaningful cross-domain collaboration. A cross-disciplinary workshop demonstrated the practical relevance of this concept, revealing its potential to surface shared understanding and guide consistency preservation efforts. The introduced approach promises potential in tool applications such as Vitruvius, both for construction of and development in boundary-crossing model landscapes. Still, the given concept remains a theoretical discussion. While a first explorative workshop was conducted to showcase the general usability of such an approach, there is still need for further investigation of boundary artifacts and their application as design supports rather than descriptive elements.

Acknowledgment

This work was funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – SFB 1608 – 501798263. This paper has been edited by our textician, Daniel Shea.

References

[1] C. A. R. Andrews, *The Rosetta Stone*: British Museum Publications London, 1981.

[2] T. A. Voelk *et al.,* “Misunderstand me correctly - comprehensibility in interdisciplinary collaboration,” *Proceedings of the Design Society*, vol. 5, pp. 2441–2450, 2025, doi: 10.1017/pds.2025.10258.

[3] R. Dumitrescu, A. Albers, O. Riedel, R. Stark, and J. Gausemeier, *Engineering in Germany - The status quo in business and science, a contribution to Advanced Systems Engineering*. Paderborn. [Online]. Available: www.advanced-systems-engineering.de

[4] E. A. Lee, “CPS Foundations,” in *Proceedings of the 47th Design Automation Conference*, 2010, pp. 737–742.

[5] I. Graessler and J. Hentze, “The new V-Model of VDI 2206 and its validation,” *at - Automatisierungstechnik*, vol. 68, no. 5, pp. 312–324, 2020, doi: 10.1515/auto-2020-0015.

[6] S. Weiss, K. Paetzold-Byhain, M. Michalides, M. Pendzik, F. Scharold, and L. Stoiber, *Agile Entwicklung physischer Produkte 2023*: Technische Universität Dresden, 2023.

[7] A. Albers, B. Ebel, and Q. Lohmeyer, “Systems of objectives in complex product development,” *9th International Symposium on Tools and Methods of Competitive Engineering TMCE 2012, Karlsruhe, Germany*, 2012.

[8] G. Spanoudakis and A. Zisman, “Inconsistency Management in Software Engineering: Survey and Open Research Issues,” in *Handbook of Software Engineering and Knowledge Engineering*, S. K. Chang, Ed.: World Scientific, 2001, pp. 329–380.

[9] A. Albers *et al.,* “Identification of Inconsistencies in Agile CPS Engineering with Formula Student,” in *Proceedings of The XXXV ISPIM Innovation Conference*, 2024, pp. 1–15.

[10] T. A. Voelk *et al.,* “Consistently Inconsistent: AI-Analyzed Patterns in Inconsistency-Situations of Cyber-Physical Systems Development,” in *2025 IEEE International Symposium on Systems Engineering (ISSE)*, 2025, in press.

[11] K. North and G. Kumta, “Knowledge in Organisations,” in *Springer Texts in Business and Economics, Knowledge Management*, K. North and G. Kumta, Eds., Cham: Springer Nature Switzerland, 2025, pp. 37–76.

[12] S. L. Star, “The Structure of Ill-Structured Solutions: Boundary Objects and Heterogeneous Distributed Problem Solving,” in *Distributed Artificial Intelligence*: Elsevier, 1989, pp. 37–54.

[13] G. C. Bowker and S. L. Star, *Sorting Things Out*: The MIT Press, 1999.

[14] W. C. Barley, P. M. Leonardi, and D. E. Bailey, “Engineering Objects for Collaboration: Strategies of Ambiguity and Clarity at Knowledge Boundaries,” *Human Communication Research*, vol. 38, no. 3, pp. 280–308, 2012, doi: 10.1111/j.1468-2958.2012.01430.x.

[15] R. Wohlrab, P. Pelliccione, E. Knauss, and M. Larsson, “Boundary objects and their use in agile systems engineering,” *Journal of Software: Evolution and Process*, vol. 31, no. 5, e2166, 2019, doi: 10.1002/smr.2166.

[16] G. Mark, K. Lyytinen, and M. Bergman, “Boundary Objects in Design: An Ecological View of Design Artifacts,” *JAIS*, vol. 8, no. 11, pp. 546–568, 2007, doi: 10.17705/1jais.00144.

[17] C. Matthies, R. Heinrich, and R. Wohlrab, “Investigating Software Engineering Artifacts in DevOps Through the Lens of Boundary Objects,” in *Proceedings of the 27th International Conference on Evaluation and Assessment in Software Engineering*, 2023, pp. 12–21.

[18] J. K. Blomkvist, J. Persson, and J. Åberg, “Communication through boundary objects in distributed agile teams,” in *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, 2015, pp. 1875–1884.

[19] M. R. Ramli, F. Asplund, and M. Torngren, “Towards Reference Architectures for Trustworthy Collaborative Cyber-Physical Systems: Reference Architectures as Boundary Objects,” *arXiv preprint arXiv:2108.12771*, 2021.

[20] A. Albers *et al.,* “Product Profiles: Modelling customer benefits as a foundation to bring inventions to innovations,” *Procedia CIRP*, vol. 70, pp. 253–258, 2018.

[21] A. Albers, J. Fahl, T. Hirschter, S. Haag, S. Hünemeyer, and T. Staiger, “Defining, Formulating and Modeling Product Functions in the Early Phase in the Model of PGE — Product Generation Engineering,” in *2020 IEEE International Symposium on Systems Engineering (ISSE)*, 2020, pp. 1–10.

[22] J. Köhlke, S. Hanna, and J. Schütz, “Cross-domain stakeholder-alignment in collaborative SoS-lego\textregistered as a boundary object,” in *2021 16th International Conference of System of Systems Engineering (SoSE)*, 2021, pp. 108–113.

[23] A. Albers, “Five hypotheses about engineering processes and their consequences,” in *Proceedings of the TMCE*, 2010, pp. 343–356.

[24] K. Henderson, “Flexible Sketches and Inflexible Data Bases: Visual Communication, Conscription Devices, and Boundary Objects in Design Engineering,” *Science, Technology, & Human Values*, vol. 16, no. 4, pp. 448–473, 1991, doi: 10.1177/016224399101600402.

[25] E. Subrahmanian *et al.,* “Boundary Objects and Prototypes at the Interfaces of Engineering Design,” *Computer Supported Cooperative Work (CSCW)*, vol. 12, no. 2, pp. 185–203, 2003.

[26] C. A. Lauff, D. Kotys-Schwartz, and M. E. Rentschler, “What is a Prototype? What are the Roles of Prototypes in Companies?,” *Journal of Mechanical Design*, vol. 140, no. 6, p. 61102, 2018.

[27] W. Mueller, M. Becker, A. Elfeky, and A. DiPasquale, “Virtual prototyping of cyber-physical systems,” in *17th Asia and South Pacific design automation conference*, 2012, pp. 219–226.

[28] H. Stachowiak, *Allgemeine Modelltheorie*. Wien New York: Springer, 1973.

[29] R. Pascual, B. Beckert, M. Ulbrich, M. Kirsten, and W. Pfeifer, “Formal Foundations of Consistency in Model-Driven Development,” in *Leveraging Applications of Formal Methods, Verification and Validation. Specification and Verification*, 2025, pp. 178–200.

[30] R. Reussner, I. Schaefer, B. Beckert, A. Koziolek, and E. Burger, “Consistency in the View-Based Development of Cyber-Physical Systems (Convide),” in *2023 ACM/IEEE International Conference on Model Driven Engineering Languages and Systems Companion (MODELS-C)*, 2023, pp. 83–84.

[31] R. Wohlrab, P. Pelliccione, E. Knauss, and M. Larsson, “Boundary objects in Agile practices: continuous management of systems engineering artifacts in the automotive domain,” in *Proceedings of the 2018 International Conference on Software and System Process*, Gothenburg Sweden, 2018, pp. 31–40.

[32] H. Klare, M. E. Kramer, M. Langhammer, D. Werle, E. Burger, and R. Reussner, “Enabling Consistency in View-Based System Development — The Vitruvius Approach,” *0164-1212*, vol. 171, p. 110815, doi: 10.1016/j.jss.2020.110815.

[33] Y. R. Jean-Mary, E. P. Shironoshita, and M. R. Kabuka, “Ontology Matching with Semantic Verification,” *1570-8268*, vol. 7, no. 3, pp. 235–251, doi: 10.1016/j.websem.2009.04.001.