Towards a Model-Based Approach to Link Data, Systems and Data Spaces

Yuanwei Qu[1], Dimitris Kyritsis[1] and Eduard Kamburjan[2,1]

[1]University of Oslo, Norway
[2]IT University of Copenhagen, Denmark

Abstract
Data Spaces are an important component in modern cyber-physical systems, for example, digital twins rely on a semantic foundation for data exchange. However, designing systems that use Data Spaces remains a challenge. Systems designs must consider not only the requirements and constraints of the cyber-physical system itself, but (a) incorporate semantic constraints integrated into the digital infrastructure for data generation, and (b) ensure that the data space is used correctly, to minimise overhead for developers. In this work, we describe ongoing work on using system models to integrate Data Spaces and their specification into Model-Based Systems Engineering workflows. Using an industrial case, we show how Data Spaces can be referred to from system models, and how this enables (a) a system approach to data space-aware components, and (b) a (semi-)automatic inclusion of semantic data quality attributes into the digital infrastructure.

Keywords
Data Spaces, Systems Engineering, System Models

1. Introduction

Data Spaces have emerged as the critical infrastructure in modern cyber-physical systems, which have often been used in digital twin solutions and rely on a semantic foundation for data exchange across industrial sectors. Driven by regulatory initiatives like the European Data Governance Act (DGA)[1], they provide the necessary governance and architectural blueprints for secure, fair, and interoperable data exchange.

However, while legal and architectural frameworks are maturing, the practical implementation of Data Spaces remains a significant engineering challenge. For users and developers, one technical challenge is to ensure that data exchanged across heterogeneous systems and organisational boundaries is interpreted consistently and correctly [1]. While the FAIR principles provide the goal, mature methods for automated, scalable integration in complex industrial settings are not available [2]. At the same time, Data Spaces must be integrated into existing digital solutions for cyber-physical systems and infrastructures to avoid long-term maintenance burdens or architectural complexity.

This paper presents a position on how to address the challenges of (a) enabling engineers to provide reusable blueprints[2] for developers, and (b) describing pipelining workflows[3] as part of these blueprints and discussing the use of model-based systems engineering as a solution. The difficulties in applying model-based systems engineering are based on the following two interconnected (sub-)challenges that currently hinder the development and broader adoption of Data Spaces from a systems engineering and developer perspective:

- **C1: Integrating Semantic Constraints into the System Design.** For data to be truly interoperable, its semantic meaning and quality must be assured at the point of origin. Currently, a gap exists between the industrial control and sensing systems that generate data and the semantic rules of the Data Space. This forces developers to conduct a *post-hoc* integration, where data are generated

The Fourth International Workshop on Semantics in Dataspaces (SDS 2026) in conjunction with ESWC 2026, Dubrovnik, Croatia
quy@ifi.uio.no (Y. Qu); dimitrky@ifi.uio.no (D. Kyritsis); eduard.kamburjan@itu.dk (E. Kamburjan)
0000-0003-3220-2101 (Y. Qu); 0000-0002-0996-2543 (E. Kamburjan)

[1]https://digital-strategy.ec.europa.eu/en/policies/data-governance-act
[2]https://w3c-cg.github.io/dataspaces/#blueprints
[3]https://w3c-cg.github.io/dataspaces/#pipelines

first and then manually follow some pipelines to ensure data semantic meaning and constraints to make sure it is ready for Data Spaces consumption. For example, a temperature sensor on a weaving machine generates some critical measurements. When semantic constraints are not considered during system design, the system designer typically lacks awareness of the semantic requirements imposed by Data Space connectors. As a result, the generated data cannot be directly consumed by a Data Space and need to undergo *post-hoc*, often semi-manual, integration steps to align with the required semantic rules. These rules are typically specified in Data Product definitions [3] or in the IDSA reference architecture, where mechanisms such as vocabulary hubs [4] and asset governance guidelines [5] define the expected semantics.

- **C2: Minimizing Developer Overhead**. The complexity of Data Space architectures and standards can overwhelm developers, acting as a barrier to entry. Making the generated data ready to be shared with a designed Data Space requires engineers to check the data's semantic meaning with system context information, both the engineering system design and how to connect to the Data Space, which is time-consuming and error-prone. To make data easily shared and used via Data Spaces, a developer building a data product should not need deep expertise in system design; the integration should be a simplified, model-driven task.

Model-Based Systems Engineering has been used to tackle similar challenges in interdisciplinary, large-scale development efforts for cyber-physical systems, where constraints from one engineering domain must be considered, integrated and referred to from another [6, 7, 8, 9]. It is based on the use of semi-formal models to describe requirements, system architecture and developed product. These models may be domain-specific and refer to one another. Semantic technologies are used in model-based systems engineering to ensure correctness between models or integrate exchanged data [10, 11, 12]. Our core insight is that Data Spaces must be considered at the systems engineering level, and their specification treated as a model that can be referred to and used for semantic integration of data.

In this work, we report on our ongoing work on using system models with semantic technologies for addressing these two challenges. We propose a model-based co-design approach where Data Space semantics are embedded as constraints within the system engineering models and the corresponding system model knowledge graphs. Building on this foundation, we show how Data Spaces can be referred to from system models, and how this enables (a) a system approach to build Data Space-aware components, and (b) a (semi-) automatic inclusion of semantic data quality attributes into the digital infrastructure tackling the above challenges.

2. Context and Challenge Alignment

This section positions our work within the evolving landscape of Data Spaces, semantic technologies and model-based systems engineering. We analyse current research to clarify the specific gap our approach aims to fill.

2.1. Semantic Interoperability in Data Spaces

The reference architecture model for Data Spaces is well-established, with the International Data Spaces Reference Architecture Model (IDS-RAM) serving as the blueprint [13]. Aiming to provide a basis for smart services and cross-company business processes, while at the same time guaranteeing data owners' sovereignty over their content, an ontology has been proposed following IDS-RAM [14]. Current research increasingly focuses on addressing semantic interoperability, which remains a key barrier to scalable integration [15]. One notable direction is the Semantic Data Link (SDL) framework [16], which aims to democratize data description through a layered architecture that separates definitional, structural, and contextual aspects, independent of specific domains. One complementary work is building a semantic layer in Data Space, such as federated vocabulary hubs [4], to enable agreement on semantics across interconnected Data Spaces. In addition to semantic layers, novel methods are being proposed to automate the matching of heterogeneous data assets using graph and contextual embeddings [17], and pipelines that extend existing ontologies with a toolkit aiming to enable semantic interoperability across local data silos and sharing via IDSA connectors [18]. These works address how to describe, connect, and evaluate data that has already been produced but do not solve the upstream problem of how to generate data that is natively compliant with these descriptions and quality metrics from its origin **C1**.

2.2. Lowering the Barrier to Enter Data Spaces

Minimizing developer' efforts in building and operating data products for Data Spaces has been recognised as one key challenge [19]. Data Spaces related organizations have already taken initiatives: works such as Tekniker Data Space Connector tries to pack complex functions (contract negotiation, policy enforcement) into reusable components with a clean interface [20], and the Simpl-Open initiative fthe rom European Commission [21] aims to pack various critical and modular components for operating Data Spaces that users can adopt whole or in part. At the same time, recent research is moving toward model-driven approaches that hide architectural complexity. Klímek et al. [22] provide a model-driven framework that enables domain experts to define data exchange models and automatically generate the necessary artifacts. Despite these advances, no solutions provide a methodology that binds Data Space requirements to system design, while it can be generalised and reduce the workload of Data Space developers **C2**.

2.3. Model-Based Systems Engineering

To bridge the methodological gaps identified above, we look beyond traditional IT integration patterns and aim to employ the benefits of Model-Based Systems Engineering (MBSE). Systems engineering is a transdisciplinary and integrative approach to managing complex engineering projects [23]. At its core, MBSE aims to connect different engineering disciplines by providing an integrative, overarching perspective across the entire design and operation lifecycle of a system. We argue that by extending the system model to include digital semantic contracts, we can shift Data Space compliance from a fragile, post-hoc IT task to a rigorous, systems engineering property.

3. Model-Based Approach to Data Space Readiness

Next, we report on ongoing work that utilises system models to address the challenges of semantic interoperability between system design and Data Space compliance **C1** and developer overhead **C2**.

3.1. A Systems Approach to Data Space-Aware Components

We present our core position that the challenges **C1** and **C2** cannot be solved solely by more comprehensive ontologies or better data connectors. These issues are raised from a *data-first* paradigm, where integration is treated as a reactive, post-hoc task. To overcome these challenges, we propose a shift toward a system-first approach, enabling the creation of natively *Data Space-aware* components.

The core of this shift is a design framework that positions the system model as the authoritative source for orchestrating Data Space integration and explicitly describes not only the internal logic of designed systems but also their conceptual data connectors. To be specific, this design requires system models to include:

- **Where the Data Space connector is connected**: Explicitly modelling where the Data Spaces connector resides in the system model and identifying where information exits the system boundary to be delivered to the Data Space. This establishes that the integration point is an intrinsic part of the system design.
- **What part of the Data Space specification is related**: Defining what information is being shared, its intended purpose, and the specific semantic and technical requirements it must fulfill. This ensures that the system's output is aligned with the requirements of the target Data Space.

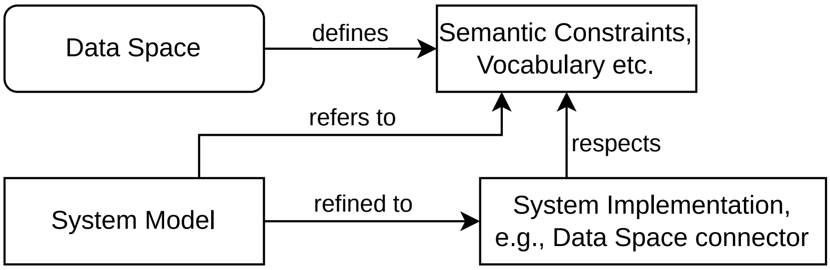

Figure 1: The system model describes the relation of the Data Space connectors to the rest of the system, as well as their relation to the semantic constraints defined by the Data Space. The implementation can then make use of these explicit references during development.

By leveraging semantic technologies, modern modelling languages, such as SysML v2 [24], the Ontological Modeling Language (OML) [25], or the Information Modelling Framework (IMF) [26], are able to serialise system models into RDF-based knowledge graphs. This semantic grounding allows engineers to incorporate digital exchange requirements directly into the engineering model, transforming the system model into a unified foundation for both physical and digital design. Consequently, this facilitates a *design-and-assure* workflow that automates the generation of compliant, self-describing information units. By capturing the essential context of data exchange at the source, this approach yields several advantages.

For one, it is easier to *integrate* semantic constraints directly into the system model and *enforce* them in the final system. Instead of treating technical and semantic requirements as secondary considerations, they become intrinsic to the component's definition. This fusion ensures the model captures the system context from the beginning to prevent the typical loss of constraints in post-hoc integration, where the integration is not enforced by the model.

As the constraints are integrated, it is possible to validate them earlier, which is indeed one of the main advantages of model-based systems engineering [27]. Because the data-generating components and conceptual Data Space connectors are modelled in one system model, engineers can perform early validation and verification on a system level. This ensures that the system-data design is physically and semantically consistent before deployment. As the system model is designed to be Data Space-aware, the system integration and data integration can operate on a shared foundation.

The relation between the system model, which describes where the Data Space connector is w.r.t. other components of the system, the system implementation, including this very Data Space connector, and the semantic constraints and vocabulary defined by the Data Space specification are illustrated in Fig. 1.

3.2. System Modeling

In the following, we describe how we apply the system-first paradigm using the Information Modelling Framework (IMF) [26] to structure the system design. While other semantic-enabled systems modelling languages, such as SysML [24] or the OML [25] possess the theoretical capability to support this workflow, we employ IMF as the primary workflow within the context of our current EU research activities.

The IMF provides a structured methodology for describing complex systems through three core elements: blocks, terminals (ports), and connectors. One defining feature of the IMF is its multi-aspect approach, where every element is characterised by aspects such as product, function, or location to ensure the design structure and context of the system are preserved. The framework is built upon a formal semantics that enables the serialisation of the system model into RDF knowledge graphs.

During the system modelling phase, our approach extends the traditional systems engineering architecture by explicitly integrating the conceptual Data Space connectors into the design. We model the conceptual Data Space connector as an *imf:Block*, which interacts with internal system components

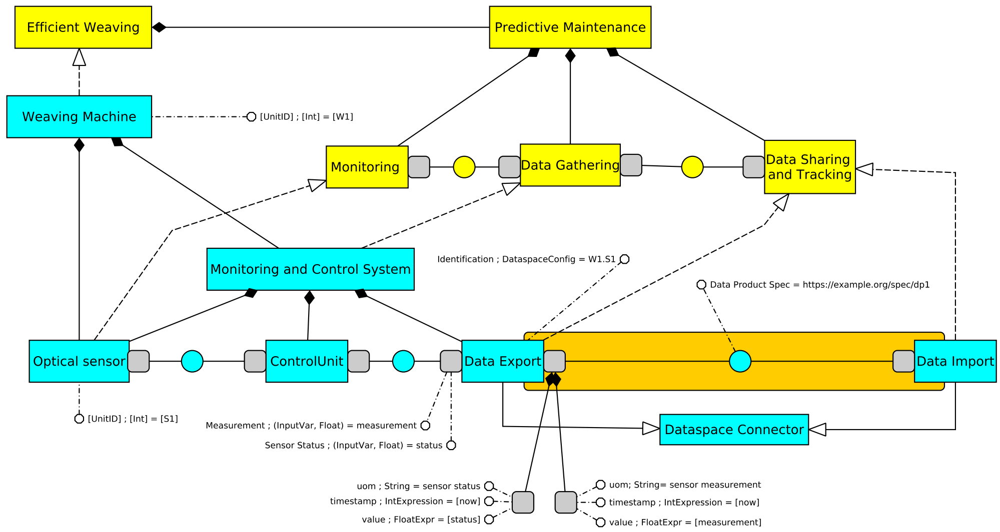

Figure 2: Semantic System Model referring to a Data Space Specifications

through its terminals. This fulfills the "Where" requirement of our framework: it establishes a definitive interface where internal operational data transitions into a sovereign, externally shared Data Space. By anchoring the connector as a distinct block, we explicitly define the boundary of data sovereignty within the system's topology.

To address the "What" requirement, we configure these connector terminals to act as binding semantic contracts. Based on the target Data Space's specifications, we enrich the terminals with attributes with precise quantifiers such as required units of measure, measure constraints (e.g., lower limit, upper limit), and mode (e.g., design, operating, allowable). In this way, we describe exactly what type of data, and under what quality constraints, the system is authorised to exchange.

Because these rigorous exchange specifications are embedded directly into the system model, serialising the model into an RDF knowledge graph yields a powerful result: the Data Space requirements become intrinsic to the graph. During the serialisation, we can specify which external ontologies mandated by the Data Space that we need to map to. The resulting system model knowledge graphs serve as a computable *ground truth* that binds the generated data to its engineering context. This unified graph foundation not only dictates the generation of compliant data products but also supports advanced reasoning, allowing engineers to automatically query and verify that the physical system design satisfies all external Data Space constraints long before actual deployment.

4. Use Case in the Data Space for Manufacturing Equipment

Fig. 2 shows a system model in the IMF language for a simplified use case in the SM4RTENANCE Data Space for Data Exchange for Manufacturing Equipment. IMF is a system modelling language which interprets system models as knowledge graphs[4].

Itema is a manufacturer of weaving machines, which transmit information about their state, concretely the data collected by an optical sensor about the wear on a certain part of the machine. Itema also offers a cloud-based predictive maintenance service. This service relies on data products exchanged between the owner of the weaving machine and the cloud infrastructure to function effectively across organisational boundaries. The system must securely share high-fidelity operational data.

The model consists of three parts. The yellow elements at the top denote the required functions to monitor the system, transmit the data from the monitor to an interface and gather the data from all

[4]https://www.imfid.org

monitors in that interface. The blue elements below are a system breakdown of the weaving machine, annotated with data, such as the ID of the sensor and the sensor.

Finally, the Data Export and Data Import realise the connection of the data space, modelled by extending the Dataspace Connector block. The export is our focus here. At its input, its port describes what data is collected (e.g., `Sensor Status`), its type (e.g., `Float`) and an internal name (e.g., `status`). Its output port has three parts. Two describe outputs (e.g., `sensor status`) and how it is computed (e.g., it copies `sensor` and sets the timestamps to the current time). The last gives an idea, composed of the id of the weaving machine and the sensor. This describes how this connector has to be implemented, what data access it needs. The connection itself, the circle in the orange box, refers to the Data Space specification, concretely the Data Product specification[5] [3] for SM4RTENANCE, or the vocabulary hub[6] if following the IDSA architecture, where semantic and federated approaches are available for both vocabulary hubs and semantic asset management [4, 5]. The outputs `sensor status` and `sensor measurement` must occur there. The model is simplified as the full model is not publicly available, and does not contain the cloud infrastructure and the actual operational part of the weaving machine for this reason.

Semantics play a two-fold role. First, the IMF model serializes to RDF/OWL knowledge graphs, allowing it to be checked for consistency w.r.t. logical and other constraints. For example, a SHACL is used to ensure that the pattern for dataspace-connectors is used correctly. Second, the data exchange is described in terms of semantic constraints itself by referring to the vocabulary of the Data Space. Thus, one can query the IMF model for these constraints.

The system model addresses the identified challenges as follows. Challenge **C1** is addressed by the explicit references to the constraints in the system model. The semantics of data is, thus, a first-class citizen of the system design, and must not be added after deployment. Early validation can be done on this level: For example, the semantic constraints require that the ID of the machine is transmitted, which in turn requires already in the design of the weaving machine, it must be considered which element's ID is used and how to transmit it to the data connector. Challenge **C2** is addressed by giving the developer of the data connector a system model that can either be refined into executable code, or used as a requirement for manual development. This system model, as described above, can also be queried to automatically retrieve information needed to develop and deploy a data connector.

5. Discussion

The Itema use case demonstrates the feasibility of using a system model to include Data Space connectors together with semantic technologies, such as vocabulary hubs, in system design to enable engineers to provide reusable blueprints that can serve as a description of the workflow and pipeline to be implemented by software developers. Through our ongoing work, we also notice that the proposed method introduces new methodological and technical challenges.

Our framework requires system model to be mapped to external semantic specifications (e.g., specific ontologies). The first challenge is that it shifts the semantic integration burden onto mechanical or systems engineers, who may lack expertise in formal knowledge representation. Existing tools and languages use different approaches to connect a system model with an ontology and corresponding knowledge graphs, but there are numerous companies and domains that are currently combining systems engineering and semantic technology. Thus, there is no clear best practice right now, bridging the gap between systems engineering tool and semantic web standards is a current challenge beyond Data Spaces.

In addition, as we scale from single machines (like the weaving machines) to entire factory floors or *System-of-Systems* design, the resulting knowledge graphs will become complex. It remains unclear whether the computational performance of graph-reasoning engines when verifying extensive semantic contracts scales across millions of interconnected nodes.

Despite these methodological and technical challenges, the additional effort required for semantic

[5]https://www.datamesh-architecture.com/data-product-canvas
[6]https://docs.internationaldataspaces.org/ids-knowledgebase/ids-ram-4/layers-of-the-reference-architecture-model/3-layers-of-the-reference-architecture-model/3_5_0_system_layer/3_5_6_vocabulary_hub

enrichment during design is outweighed by the potentially substantial reduction in manual and error-prone data integration during deployment. While our approach does not eliminate the need for mapping over time, embedding constraints natively in the system model provides traceability. In this sense, the proposed approach shifts effort from post-hoc integration to system-first and early-stage formalisation, resulting in a more robust and maintainable Data Space architecture. As this work is preliminary, future empirical validation and tooling are needed to fully assess its value.

6. Conclusion

We have presented our vision and ongoing work on combining system engineering and semantic technologies to make semantic constraints a formally grounded component in system models and use these system models then as structured and queriable requirements for software developers. We are currently investigating further uses for dataspace-aware system models, for example, to provide a Digital Thread or as an enabler for self-adaptation [28], and are developing a methodology, including patterns, to enable practitioners to use our approach.

Acknowledgments

This work is partially supported by the EU Commission funded projects SM4RTENANCE (101123490) and Tec4MaaSEs (101138517). The authors thank the support from Itema and Intellimech.

Declaration on Generative AI

The authors have used ChatGPT to assist with the polishing of human-authored text. The authors take full responsibility for the publication's content.

References

[1] S. Steinbuss, Semantic Interoperability in Data Spaces, Technical Report, International Data Spaces Association, 2024. URL: https://doi.org/10.5281/zenodo.10964377, accessed 22 Feb 2026.
[2] L. Vogt, P. Strömert, N. Matentzoglu, N. Karam, M. Konrad, M. Prinz, R. Baum, Suggestions for extending the fair principles based on a linguistic perspective on semantic interoperability, Scientific Data 12 (2025) 688. doi:10.1038/s41597-025-05011-x.
[3] The Linux Foundation, Open Data Product Specification 4.1, Technical Report, 2025. URL: https://opendataproducts.org/v4.1, accessed 22 Feb 2026.
[4] R. David, V. Alexiev, P. Ivanov, W. van den Berg, J. P. Wijbenga, M. Stornebrink, Federated vocabulary hubs as a foundation for semantic layers in data spaces, in: J. Theissen-Lipp, P. Colpaert, A. Pomp, E. Curry, S. Decker (Eds.), SDS, volume 4007, CEUR-WS.org, 2025. URL: https://ceur-ws.org/Vol-4007/06short.pdf.
[5] L. Sánchez-González, A. Iglesias-Molina, Ó. Corcho, M. Poveda-Villalón, On the governance of semantic artefacts in dataspaces, in: SDS, volume 3705 of *CEUR Workshop Proceedings*, CEUR-WS.org, 2024. URL: https://ceur-ws.org/Vol-3705/short08.pdf.
[6] D. D. Walden, T. Shortell, G. Roedler, B. Delicado, O. Mornas, Y. Yew-Seng, D. Endler, INCOSE Systems Engineering Handbook, 5th Edition, Wiley, 2023.
[7] P. De Saqui-Sannes, R. A. Vingerhoeds, C. Garion, X. Thirioux, A taxonomy of MBSE approaches by languages, tools and methods, IEEE Access 10 (2022) 120936–120950. doi:10.1109/ACCESS.2022.3222387.
[8] V. Singh, K. E. Willcox, Engineering design with digital thread, AIAA Journal 56 (2018) 4515–4528. doi:10.2514/6.2018-0569.

[9] A. M. Madni, M. Sievers, Model-based systems engineering: Motivation, current status, and research opportunities, Systems Engineering 21 (2018) 172–190. doi:10.1002/SYS.21438.

[10] E. Karabulut, S. F. Pileggi, P. Groth, V. Degeler, Ontologies in digital twins: A systematic literature review, Future Gener. Comput. Syst. 153 (2024) 442–456. doi:10.1016/J.FUTURE.2023.12.013.

[11] J. Gregory, A. Salado, Towards a systems engineering ontology stack, INCOSE International Symposium 34 (2024) 1304–1318. doi:https://doi.org/10.1002/iis2.13210.

[12] M. Elaasar, N. Rouquette, D. Wagner, B. J. Oakes, A. Hamou-Lhadj, M. Hamdaqa, opencaesar: Balancing agility and rigor in model-based systems engineering, in: 2023 ACM/IEEE International Conference on Model Driven Engineering Languages and Systems Companion (MODELS-C), 2023, pp. 221–230. doi:10.1109/MODELS-C59198.2023.00051.

[13] International Data Spaces Association, nternational data spaces reference architecture model, 2025. https://docs.internationaldataspaces.org/ids-knowledgebase/ids-ram-4.

[14] S. Bader, J. Pullmann, C. Mader, S. Tramp, C. Quix, A. W. Müller, H. Akyürek, M. Böckmann, B. T. Imbusch, J. Lipp, et al., The international data spaces information model–an ontology for sovereign exchange of digital content, in: ISWC, Springer, 2020, pp. 176–192. doi:10.1007/978-3-030-62466-8_12.

[15] Z. Boukhers, C. Lange, O. Beyan, Enhancing data space semantic interoperability through machine learning: a visionary perspective, in: ACM Web Conference 2023, 2023, pp. 1462–1467. doi:10.1145/3543873.3587658.

[16] M. Stäbler, P. Moosmann, P. Dittmer, D. Wang, C. Lange, F. Köster, Semantic data link: Bridging domain-specific needs with universal and interoperable semantic models, in: SDS, Springer, 2024. URL: https://ceur-ws.org/Vol-3705/short13.pdf.

[17] M. Stäbler, M. Lange, S. Kipper, C. Langdon, F. Köster, Gc-dam: Graph and contextual embeddings for heterogeneous data asset matching, in: SDS, CEUR-WS, 2025. URL: https://ceur-ws.org/Vol-4007/07regular.pdf.

[18] T. Ranathunga, S. Bharti, A. McGibney, A semantic interoperability toolkit for sharing energy data and models in a manufacturing data space, in: Workshop on Semantic Interoperability in Data Spaces, Budapest, Hungry, 2024.

[19] M. Bacco, A. Kocian, S. Chessa, A. Crivello, P. Barsocchi, What are data spaces? systematic survey and future outlook, Data in Brief 57 (2024) 110969. doi:10.1016/j.dib.2024.110969.

[20] International Data Spaces Association, What tekniker offers to data spaces users, 2025. https://internationaldataspaces.org/what-tekniker-offers-to-data-spaces-users/.

[21] European Commission's Simpl Programme, Simpl-open, 2025. https://simpl-programme.ec.europa.eu/dashboard/simpl-open.

[22] J. Klîmek, P. Koupil, P. Škoda, J. Bártîk, Š. Stenchlák, M. Nečaskỳ, I. Holubová, Atlas: A toolset for efficient model-driven data exchange in data spaces, in: 2023 ACM/IEEE International Conference on Model Driven Engineering Languages and Systems Companion (MODELS-C), IEEE, 2023, pp. 4–8. doi:10.1109/MODELS-C59198.2023.00009.

[23] INCOSE, Systems engineering 2035 vision, 2022. URL: https://incose.org/publications/se-vision-2035.

[24] O. M. Group, Systems Modeling Language (SysML) Version 1.6, Technical Report, Object Management Group, 2024. URL: https://sysml.org/, accessed 30 Oct 2025.

[25] M. Elaasar, N. Rouquette, Ontological Modeling Language v2, Technical Report, OpenCaesar, 2025. URL: https://www.opencaesar.io/oml/, accessed 22 Oct 2025.

[26] DNV, DNV-RP-0670 Asset information modelling framework, Technical Report, Det Norske Veritas, 2024. URL: https://www.dnv.com/digital-trust/recommended-practices/asset-information-modelling-dnv-rp-0670/, accessed 02 Feb 2026.

[27] J. Cederbladh, A. Cicchetti, J. Suryadevara, Early validation and verification of system behaviour in model-based systems engineering: A systematic literature review, ACM Trans. Softw. Eng. Methodol. 33 (2024) 81:1–81:67. doi:10.1145/3631976.

[28] E. Kamburjan, N. Bencomo, S. L. T. Tarifa, E. B. Johnsen, Declarative lifecycle management in digital twins, in: EDTConf, ACM, 2024, pp. 353–363. doi:10.1145/3652620.3688248.