PRACTICAL APPLICATION

systems

MDPI

Article

Collaborative Model-Based Systems Engineering Using Dataspaces and SysML v2

Zirui Li, Faizan Faheem and Stephan Husung *

Product and Systems Engineering Group, Technische Universität Ilmenau, 98693 Ilmenau, Germany; zirui.li@tu-ilmenau.de (Z.L.); faizan.faheem@tu-ilmenau.de (F.F.)
* Correspondence: stephan.husung@tu-ilmenau.de; Tel.: +49-3677-69-2472

Abstract: Collaborative Model-based Systems Engineering between companies is becoming increasingly important. The utilization of the modeling possibilities of the standard language SysML v2 and the multilateral data exchange via Dataspaces open new possibilities for efficient collaboration. Based on systemic approaches, a modeling concept for decomposing the system into sub-systems is developed as a basis for the exchange. In addition, based on the analysis of collaboration processes in the context of Systems Engineering, an architectural approach with a SysML editor and Dataspace for the exchange is elaborated. The architecture is implemented on the basis of open-source solutions. The investigations are based on an application example from precision engineering. The potential and challenges are discussed.

Keywords: SysML v2; dataspace; collaboration; Systems Engineering; MBSE; requirements; specification

Practical application and need for modeling guidelines

1. Introduction and Motivation

Product development faces many challenges, such as increasing individualized stakeholder demands, rapid technological evolution, regulatory compliance, etc. An important challenge is the increased and growing focus on the realization of needs-oriented solutions [1,2]. The dynamic nature of these challenges leads to a further increase in product complexity (e.g., transition from primary mechanical to mechatronic to cyber-physical systems [3,4]), as well as demands on quality, reliability, safety, etc. [5]. In order to fulfill these requirements, valid product information is needed from all phases of the product life cycle [6,7].

Due to different boundary conditions (e.g., knowledge in the companies, specializations, capacity utilization, global sales, etc.), many products are developed and finally also produced with partners across companies [8–12]. One company assumes overall responsibility and distributes the entire product. This company is usually referred to as the OEM [13]. The OEM defines the scope of what suppliers need to deliver. The share of suppliers in the products varies depending on the industry sector. Surveys show a share of around 50 to 78% [14]. The suppliers are given different requirements and boundary conditions for the development of the scope of supply. To enable the OEM to evaluate the supplier's development results in the context of the overall product and make any necessary modifications to the overall product or to the requirements of the scope of supply, it is increasingly necessary to hand over not only the product specific to the scope of supply, including the necessary verification results, but also its specification, among other things [15,16].

The development of complex mechatronic products is increasingly supported by model-based approaches [17]. The approaches range from semi-formal models, including the associated tools, to formal models for the detailed specification and the simulation of products to determine specific properties [18]. Semi-formal models play a decisive role in the early development phases in particular, as essential information for the specification is not yet available with a sufficient degree of maturity, several variants have to be

check for updates

Citation: Li, Z.; Faheem, F.; Husung, S. Collaborative Model-Based Systems Engineering Using Dataspaces and SysML v2. *Systems* **2024**, *12*, 18. https://doi.org/10.3390/systems12010018

Academic Editor: Ed Pohl

Received: 20 December 2023
Revised: 4 January 2024
Accepted: 6 January 2024
Published: 9 January 2024

described and analyzed, and numerous changes and iteration loops are necessary due to increasing knowledge during development [19]. In this context, the term Model-Based Systems Engineering (MBSE) is used. Complex mechatronic products can be described for development and specification using systems theory approaches [20,21]. One of the approaches to systems theory is that systems (hereafter referred to as the overall system) can be decomposed into several sub-systems (see Figure 1), and the overall system itself can be a sub-system of superordinate systems. The decomposition of the overall system into sub-systems is basically arbitrary [22]. The decomposition is often based on functional or organizational aspects [23], for example, in combination with the distribution of development to different suppliers. In the case of complex mechatronic products, decomposition often results in several system levels with mechatronic sub-systems. The SysML modeling language has become established for the semi-formal modeling of the overall system and the mechatronic sub-systems in MBSE [7,24,25].

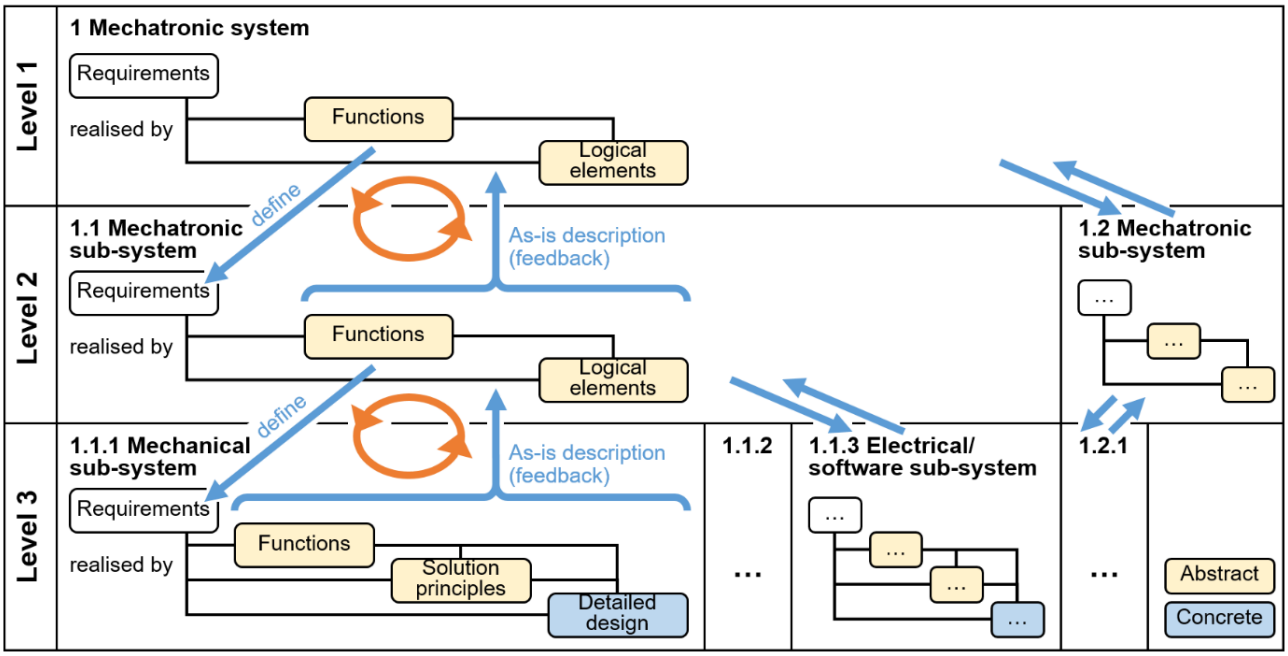

Figure 1. Systems Engineering at different system levels [19]. (blue arrows refer to information transfer, orange arrows refer to the development loops).

During development, it is important to note that SysML tools alone are not sufficient to comprehensively specify and analyze the product. Against this background, several solutions already exist today for linking SysML models with requirements, detailed formal specification models (e.g., CAD models [26] or software models [27]), or simulation models [28,29] in order to create a consistent and traceable overall description [30]. In this contribution, the focus is on SysML models.

During product development, numerous decisions must be made to detail the product. These decisions require information on the current product, including its requirements or specifications, as well as existing knowledge [31] and findings from verification and validation [32]. Product information is often distributed across partners due to collaboration with suppliers [15]. In order for the individual partners to be able to carry out their development in a target-oriented manner, the relevant information and its relationships must be available to the partners. Various aspects must be considered for Collaborative Model-Based Systems Engineering (CMBSE); some relevant aspects are as follows:

- OEMs and suppliers need means to exchange their model-based descriptions of the requirements in connection with the defined division of tasks and responsibilities as well as solution descriptions in a version-consistent form [15].

- Product development, especially in a network with partners, is highly dynamic, as new findings, e.g., from verification and validation [32], can lead to new decisions. This requires the information and its relationships to be updated, which leads to new versions of the models. For efficient CMBSE, the new versions of the models must be exchanged with the partners.
- The information in product development is highly sensitive [33]. Against this background, the partners must decide selectively and comprehensibly who receives which information and for what purpose.

Against the background of the needs discussed above, this contribution discusses an approach to supporting CMBSE. The following two technologies are used:

- Modeling in the context of MBSE is performed using the standardized language SysML version v2 [34].
- Dynamic data exchange between the partners are discussed with the utilization of a Dataspace [35].

Using an example collaboration process (based on the Prostep ivip recommendation [15]) and a prototypical implementation of the CMBSE with SysML v2 and a Dataspace, the current possibilities and open challenges will be discussed in order to overcome the limitations of the current CMBSE (see Section 2). The authors of the contribution are aware that, despite the desire for consistent Model-Based Systems Engineering, not all information is currently available in a model-based form. For the objectives of this contribution, however, it is assumed that the information is validly represented in the models to a sufficient extent. In addition to the approaches presented in this contribution, further information must be transferred via additional documents or something similar.

The following research questions are relevant to this contribution:

1. How can CMBSE be implemented efficiently using SysML v2 and Dataspaces? Details of the main research questions are provided in the following sub-questions.
    a. How do development partners need to structure the system models in SysML v2 for CMBSE?
    b. How can the necessary SysML v2 models be exchanged using Dataspaces?
    c. How should a collaborative development process be designed?
2. What future developments in the context of SysML v2 and Dataspaces are necessary to support the CMBSE in a targeted manner?

2. State of the Art

2.1. Systems Modeling Using SysML

The extension of Systems Engineering methods and processes with models is called Model-Based Systems Engineering (MBSE) [36]. The most widely used modeling language in MBSE is the Systems Modeling Language (SysML) [24,37]. SysML is a semi-formal graphical modeling language [38] for modeling the product at the mechatronic system levels as well as the system context (where the requirements, use cases, and scenarios of the stakeholders and the surrounding systems, including the required interfaces, are described) [2,39,40] or the System of Systems [25,41]. Diagrams (views of the system model elements) are used for the definition of model elements as well as for the use of model elements in further development steps, allowing context- and task-specific further use of model elements [24,42].

Semi-formal modeling offers a certain degree of freedom in modeling. For data exchange with binding requirements, however, sufficient formalization via at least modeling guidelines are necessary [39,43–45].

SysML v2 is the latest version of the System Modeling Language, which addresses the limitations of SysML v1.x [46]. In contrast to SysML v1.x, SysML v2 includes not only the graphical notation but also textual notation, which is a different presentation of the same underlying model syntax. SysML v2 provides a more formal specification of its abstract

syntax, concrete syntax and semantics, and mappings between them, which improve the language's precision and integrity [34].

In system modeling with SysML (both v1.x and v2), the package is used as a container for other elements to organize the model. To realize the reusability of modeling elements in SysML v2, the model elements are divided into two important parts: definition and usage. In general, a definition package contains different definition elements that classify different types of elements, while usage elements can be stored in different packages to describe the usage of a definition in a specific context [47]. Moreover, existing knowledge, such as standard, generic context, or specific [48] terminology, can be stored in packages and imported by other packages for reuse.

SysML v2 contains the modeling language and standard API, but in this contribution only the modeling language will be discussed.

2.2. Collaboration in Engineering

In complex development projects, a targeted division of work is an essential approach for achieving goals efficiently and effectively utilizing resources and complementary skills [11]. Among other things, economic, geographical, and technical boundary conditions increasingly require collaboration between companies. The process of collaboration is very complex, as not only the tasks and responsibilities but also the information relevant to the collaboration must be exchanged between companies [10]. In addition to the organizational and technical challenges, there are also human, legal, cultural, and other challenges that cannot be discussed in this article [49].

A major organizational challenge is the synchronization of development activities [50]. While documents in particular have played a role in collaboration for a long time, collaboration processes using models are becoming increasingly important in the context of digitalization. The use of models creates the potential for collaborative development processes to be increasingly synchronized as relevant, up-to-date information can be exchanged. However, synchronization based on models or model artifacts requires coordinated configuration management [51,52].

Since models always involve a degree of formalization, content and formal modeling definitions for the models must always be coordinated in addition to the product or project information [53,54]. The definitions can include, for example, the specification of units, the use of specific model elements, or terminology.

Product Data Management (PDM), Product Lifecycle Management (PLM) [55,56], or System Lifecycle Management tools [57] have already been established for managing heterogeneous models, including the necessary configuration management within companies [58]. PLM tools support the processes within companies that are represented in the tools with sufficient formalization. Depending on the company, these are highly formalized (especially in larger companies) or less formalized (in smaller and medium-sized companies) [59]. However, it can be seen that the processes are developed differently in several companies. This is a major challenge when using PLM tools for cross-company collaboration. The potential and challenges of cross-company collaboration using PLM tools are discussed in the work of Messaadia [60] and Tilioua [61], among others. Software-as-a-Service (SaaS) PLM approaches are discussed as a new technology [62].

In recent years, the concept of Dataspaces [35] have been developed for targeted, secure, and trustworthy data exchange, which will be examined for the CMBSE in this contribution and is therefore presented in the next section.

2.3. Dataspaces

Nowadays, Dataspaces are instrumental in fostering collaborative data exchange among multiple organizations [63]. The concept of Dataspace was initially introduced as a framework for managing diverse yet interconnected data sources, aimed at facilitating the coexistence of heterogeneous data while avoiding complete data control [64]. Numerous research efforts have further supported the services within Dataspace, such as data model-

ing [65], data integration [66], data dependences [67], and querying in context [68]. There are different definitions of Dataspaces [69–71]. According to the initiatives of the European Data Strategy [72], key objectives addressed by Dataspaces include data exchange, data interoperability, and data sovereignty. Data exchange refers to the use and reuse of data among different entities. Interoperability primarily focuses on applying standards and sharing compatible formats or protocols for data from diverse sources. Data sovereignty involves ensuring entities have control over data, including the rights to collect, store, share, and allow others to use the data [70,73]. The European Commission further elaborated on the key features of the Common European Data Spaces, such as data protection and governance, in a staff working document on Dataspace [74]. This article will adopt the definition provided by the Data Spaces Support Centre (DSSC) [72], as follows:

"A distributed system defined by a governance framework that enables secure and trustworthy data transactions between participants while supporting trust and data sovereignty. A data space is implemented by one or more infrastructures and enables one or more use cases".

Many Dataspaces are currently under development, and associated use cases are being implemented in various fields with different levels of maturity. This includes sectors such as agriculture, automotive, manufacturing, energy, logistics, smart cities, supply chains, and more [75]. For example, Catena-X aims to establish unified standards for data exchange across the entire automotive value chain based on a Dataspace, ensuring transparency in data sharing among stakeholders while adhering to supply chain laws [76]. The Smart Connected Supplier Network (SCSN) [77] provides open communication standards for exchanging order-related data between enterprises or organizations. It includes the definition of a common language to facilitate the exchange of data such as orders, dispatch advice, Technical Product Data, Bills of Materials (BoM), and more.

Furthermore, some research has explored the exchange of Digital Twins within a Dataspace. For instance, Volz et al. [78] proposed the modeling process of a Digital Twin using the Asset Administration Shell (AAS) standardized template and integrated AAS into a Dataspace by extending the Dataspace technical components (Eclipse Dataspace Connector). Usländer et al. [79] analyzed the close relationship between Digital Twins and Dataspaces, proposing an assigned reference model for the development of collaborative Digital Twins.

Considering the latest advancements in the field as discussed in the state of the art, it is crucial to address the open research question mentioned in the introduction section of this paper that enables collaboration in the Systems Engineering process by integrating MBSE with Dataspaces.

3. Approach

3.1. General Approach

As explained in the introduction, many products are often developed and produced by several companies. The OEM is responsible for the overall product. In order for the development tasks to be distributed across the suppliers, the OEM must make defined specifications for the sub-systems in the context of the overall system and the expected scenarios in the product life cycle phases (scenarios during usage of the product, but also during production, transport, maintenance, etc.) [2,80]. These specifications are in the form of requirements [81]. Using model-based approaches, the requirements are at least partially described using models [44,82]. Model-based requirements include, among other things, descriptions of scenarios for the use cases, the associated system contexts, and the black box of the sub-system with the required interfaces [2]. The supplier must analyze the requirements, develop a solution for the sub-system for the OEM and describe it in terms of the requirements (usually also requirements beyond the OEM, e.g., for regulatory purposes) [83]. The sub-system must be described so that the OEM can verify that the requirements are satisfied and that the solution can be integrated into the overall product. Figure 2 shows one scenario of collaboration between OEMs and suppliers.

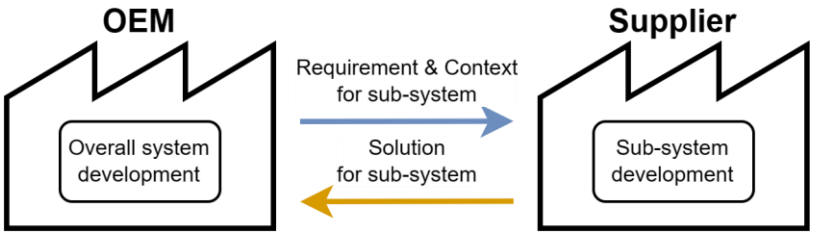

Figure 2. General approach for system development collaboration.

Based on the general approach, several related requirements for the CMBSE approach must be addressed:

- OEM needs means to transfer the requirements specification to the supplier (in the context of MBSE, at least partly model-based, even if many textual or graphical requirements are knowingly still necessary today in parallel to the models [84]). The known requirements regarding consistency of information, versioning, etc. apply to the transfer [51].
- The supplier aims to develop the sub-system based on the black-box description provided by the OEM and subsequently sends the developed description back to the OEM for comparison. Additionally, it is also necessary to ensure that configuration management rules [51] are applied for the return transfer so that the versions of the sub-system models can be assigned to the versions of the requirements.
- Product development includes highly iterative processes, as there are always uncertainties and new findings lead to adapted decisions. It is therefore important that the exchange of information between OEMs and suppliers can also take place dynamically and that partners can use the respective model versions consistently.
- During the collaboration, relevant information on the requirements and the solution description is exchanged. For such sensitive information, it must be determined specifically and selectively who receives which information in order to guarantee information sovereignty and security [33]. This means that the OEM and the suppliers must extract and provide relevant parts of the respective models for the overall system and the required sub-systems (including the requirements for the required functions, interfaces, and other properties; possibly further information on the context).
- To ensure consistency between models from suppliers and OEMs, it is essential that the modeling methodology and other definitions, including units, are aligned [85]. The agreed-upon definitions must be accessible to all partners. For this article, it is assumed that the OEM provides the definition, the supplier can make additions if necessary, and then provides these as a supplemented description.

The outlined requirements are addressed for the approach in the following sections.

3.2. System Modeling with SysML v2 in the Context of CMBSE

In the context of CMBSE, the bidirectional exchange of selected model parts is necessary. If the system models are not modeled exclusively for the partner(s) on the OEM and supplier sides, the selected model part must always be extracted from a more comprehensive system model. Utilizing models in SysML v2 for CMBSE presents significant advantages. One of them is the consistent model description between graphical and textual notations; therefore, it is feasible to derive the graphical notations from the textual notations. For data exchange during collaboration, this means that only the textual description needs to be transferred, and the graphical representation can be rendered unambiguously in the target SysML v2 software tool. The authors are aware that the implementation of SysML v2 is currently still under development and only available in a few software tools, and that software-specific changes may occur. However, this cannot be estimated at the present time. The statements in this paper refer to the possibilities of the standard and the already available pilot implementation of SysML v2 ("SysML v2 Pilot Implementation" [86] with

Eclipse plugin version 0.37.0). By transferring a purely textual description, the approach ensures the interoperability of the data and reduces the file size.

The exchange of model parts requires that the corresponding model parts can be extracted from the overall model. In the SysML v2 specification, this is supported by the package structure (see Figure 3). Structuring via packages was already possible with SysML 1.x. In combination with the textual description, this structure supports the exchange even more with SysML v2. It is important that the individual model parts (e.g., the required sub-system for the transition from OEM to supplier or the specification of the solution for the sub-system for the transition from supplier to OEM) are each described within a package. Further packages can be imported within the package, e.g., the definition package. In addition, it is important that there are no references to SysML model elements from the package to be extracted if they are not considered during the exchange. These references would become unresolvable when imported by the partner. (e.g., model B has some references to elements in model A; A and B are representative here for any models.) If model B is exchanged without model A, the partner will obtain an unresolvable error about these references in model B. At this point, there is a need for concrete modeling guidelines.

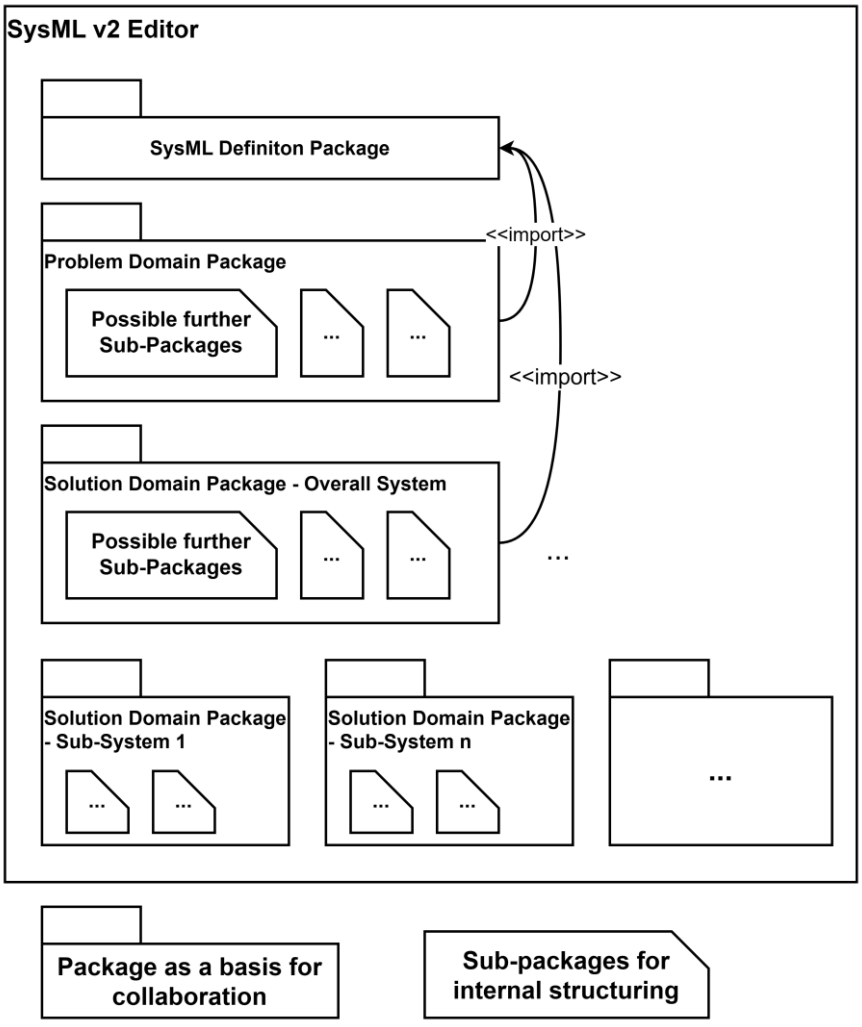

Figure 3. Package structure of a SysML model.

In the context of system modeling using SysML v2, leveraging a general definition package (cf. Section 2.1) as a foundational framework for interdisciplinary development is an appropriate approach, especially for definitions that remain stable or independent with different product variants. Sharing general definition packages among different companies contributes to maintaining a unified understanding of the product and its modeling among

different developers throughout the multilateral collaboration process. This can help reduce model compatibility issues resulting from disparate modeling processes.

After importing the general definition package, subsequent definition packages can seamlessly extend existing definitions by introducing additional attributes or specifications. Consequently, emerging usage elements defined under these new definition packages undergo further development, enabling the systematic development and refinement of the system model.

Figure 4 shows one possible collaboration process based on the system model using SysML v2. For the approach in this contribution, it is recommended that the OEM create a general definition package that it uses for its modeling. This package can include definitions of parts, interfaces, etc. (see Figure 5). Specific definitions can be derived from the general definitions of collaboration. By refining specialized definition packages, the OEM can define elements of the overall system model based on the refined definitions.

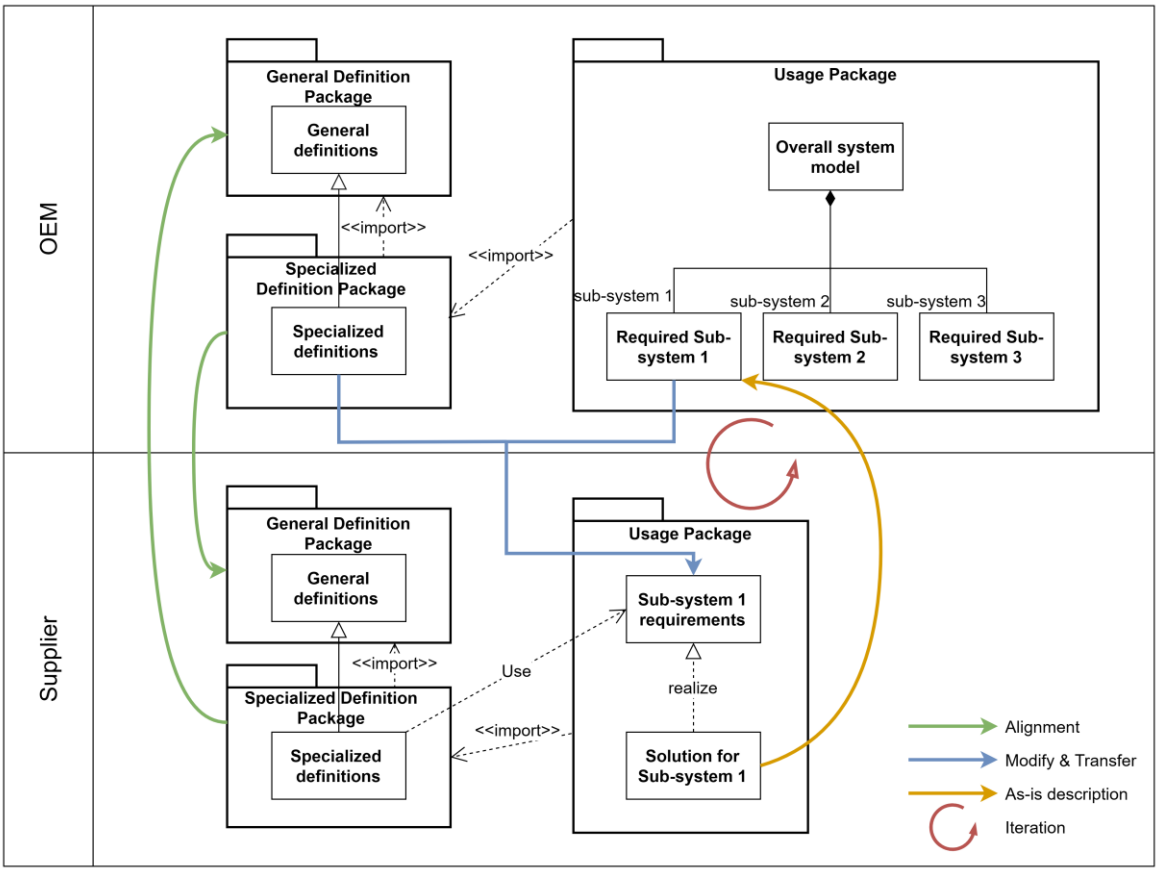

Figure 4. Collaboration process between OEM and suppliers using SysML v2 model.

According to systems theory, sub-systems are decomposed from the overall system. For collaborative development, the OEM needs to transfer the definitions and relevant descriptions of the requirements and context associated with the sub-systems to the suppliers. The transmitted data are depicted as "Sub-system 1 Requirements" in Figure 4.

On the supplier's side, they generate corresponding specialized definitions using not only their general definitions but also the requirements provided by the OEM. Subsequently, suppliers develop sub-systems for the solutions required by the OEM. In this process, suppliers should further align the system model and definitions with the general definitions shared by OEM, thereby avoiding potential model semantic conflicts due to divergent definitions for the same concept. Suppliers provide the OEM with the as-is specification of the sub-system, aligning it with the properties required by the OEM to facilitate the implementation of the sub-system within the overall system. On the supplier

side, the definition package may need to be supplemented, e.g., with additional interface specifications. This must be transferred to the OEM to supplement its definition.

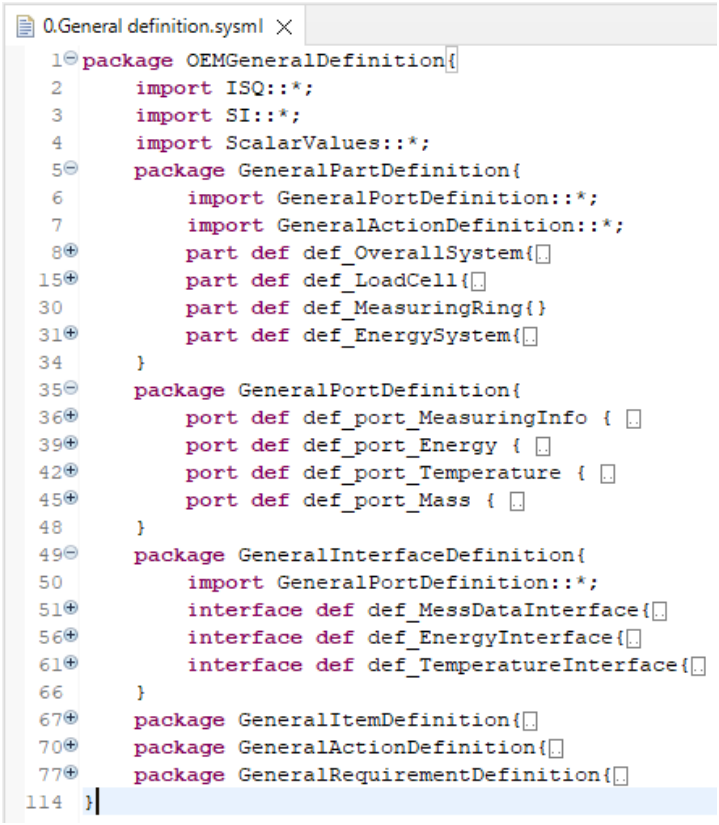

Figure 5. Excerpt from a General Definition Package in SysML v2 (see Section 5). (Symbol "*" in the Figure is used in SysML to import the package and its underlying sub-packages).

Due to continuous updates in requirements and the gaps between the as-is description and the required properties, the modeling and implementation of the system in the collaborative process undergo iterative cycles. Consequently, there is a demand for an efficient and secure exchange approach for the SysML models to effectively facilitate collaborative development.

3.3. Data Exchange via Dataspace

Before exchanging data through the Dataspace, it is imperative for both OEMs and suppliers to institute comprehensive data usage policies. Furthermore, access and use rights should be granted to this data within the appropriate contractual period to ensure data sovereignty. To facilitate collaboration with SysML models, model data needs to be systematically stored in Data Management Tools (see Figure 6) and annotated with detailed descriptions, including data types, version information, etc. This model data is categorized and organized by corresponding metadata, which includes titles, descriptions, license information, contract lists, etc. Additionally, the corresponding models have references to different contracts or agreements. In the Dataspace, links are created for the model data, establishing mutual connections with the relevant metadata, contracts, and usage policies.

The following describes a data model for the Dataspace Connector (the International Data Spaces (IDS) Dataspace Connector [87] is one of the core technical components for communication between IDS Dataspace participants), illustrating the fundamental structure of data provided by data providers within a Dataspace.

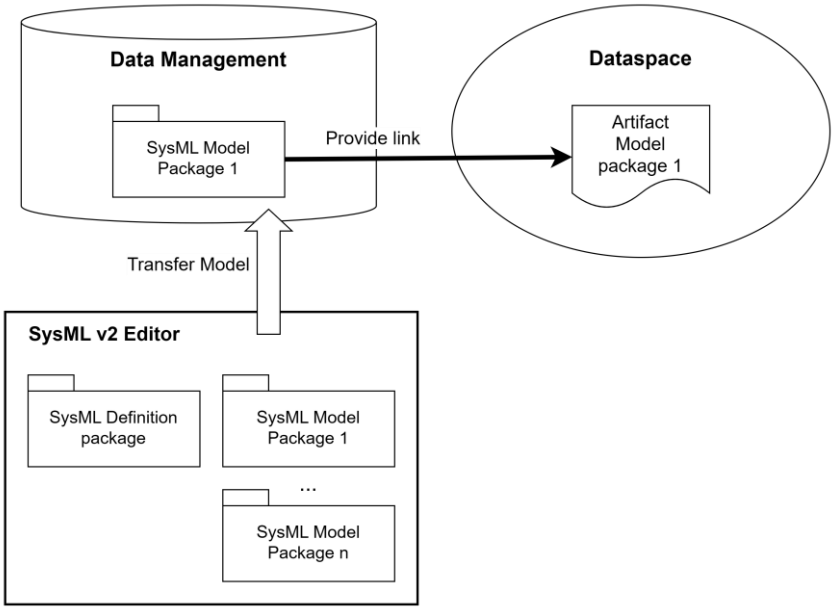

Figure 6. Transfer of the SysML model via the Data Management tool to the Dataspace.

In Figure 7, every artifact represents a singular set of data and describes the information, such as a modified date. A specific contract, which describes the usages agreed upon between the data provider and consumer, are then referred to the artifacts during the data exchange. Representations present a detailed description (e.g., data type, standard) of the artifacts. Additionally, the metadata of a data object (representation) are referred to as the resource, which includes information about the title, license information, and a list of contracts. These resources are organized within catalogs. Catalogs are regarded as the top level of navigation for data.

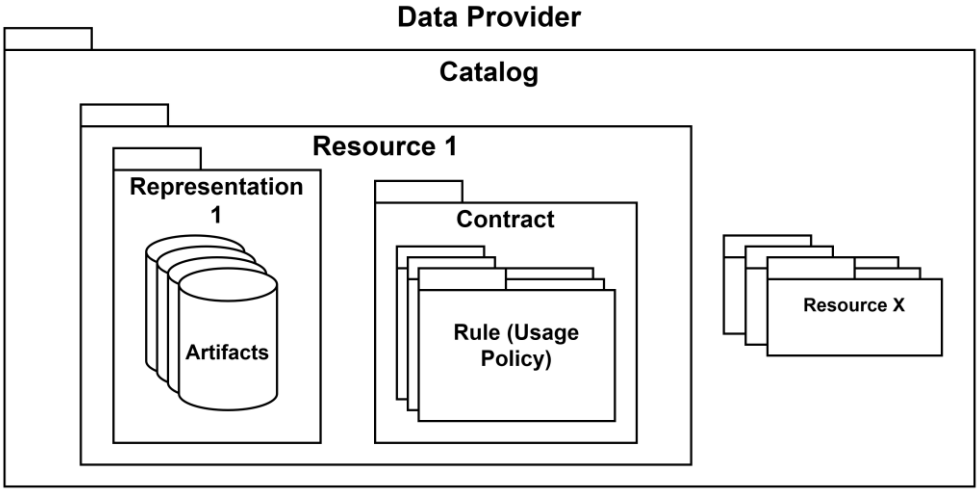

Figure 7. Data model from IDS Dataspace Connector [inspired from [88]].

In the Dataspace, the catalogs are contained in the self-description of the connector (data provider). The other connectors (as the data consumers) can request the resource metadata in the catalog only using the specific Dynamic Attribute Token (DAT) [89] within the valid "DescriptionRequestMessage". Further agreements about the data usage policy and contract must be negotiated between the data provider and consumer for the retrieval and access of each specific artifact in the metadata.

Figure 8 outlines a simplified process of data exchange (based on the preconfiguration collection file of IDS-testbed [90]) between data consumers and providers, omitting the role

of data brokers and the specific details of data transmission. Initially, the data consumer requests and reviews the self-description of the data provider through a data broker to locate the desired resources. Subsequently, the consumer requests metadata for specific resources, extracting information about representations and artifacts, along with their corresponding usage rules. Next, concerning specific artifacts, the data consumer engages in contract negotiations with the data provider. At this stage, the consumer may modify the contract. It signifies the consumer's acceptance of the contract when consumers add received contract rules or modified contracts to the request sent to the data provider. The provider can then check the contract and respond with either approval or refusal of data access. Upon receiving approval, the consumer gains unrestricted access to the data within the specified contract period.

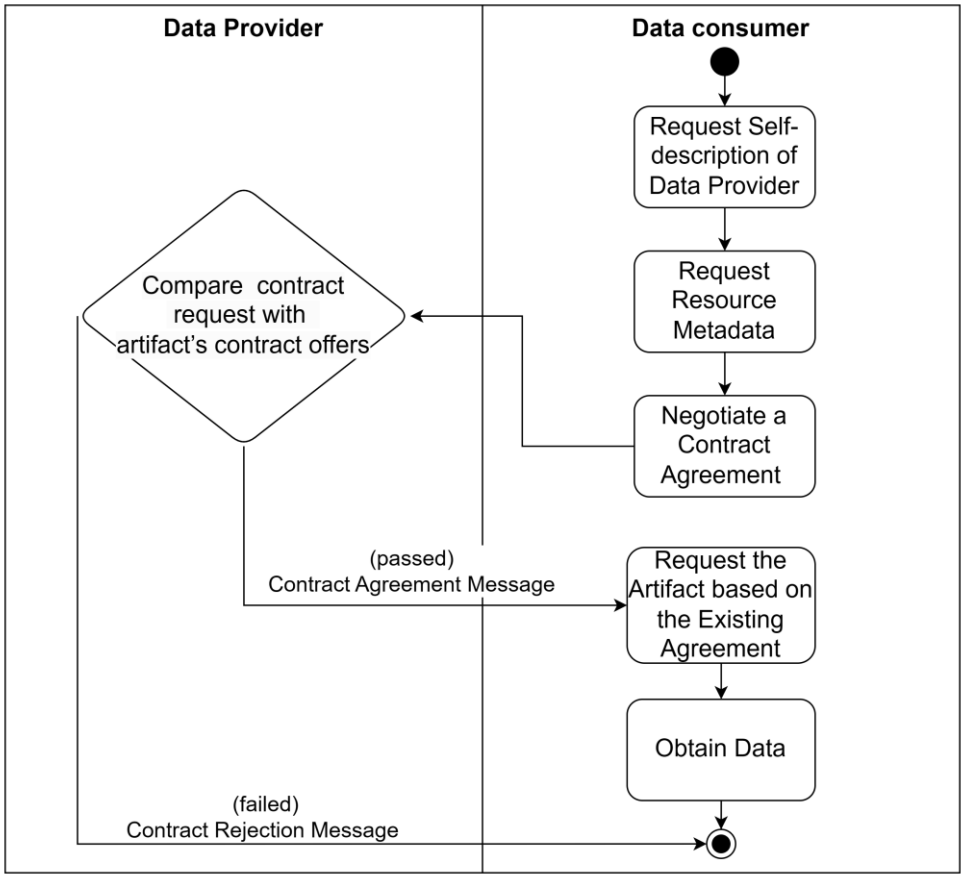

Figure 8. Simplified process of data exchange between data consumers and providers (inspired by the preconfiguration collection file of IDS-testbed [90]).

Based on the agreements reached through negotiations on contracts and usage policies, OEMs and suppliers can adopt appropriate approaches for data exchange. Leveraging the Dataspace allows engineers to share the well-classified metadata (resources and representations) first, followed by the dynamic implementation of specific model data updates. In the CMBSE process, the OEM may need to update requirements based on the as-is properties of sub-systems received from suppliers. With relevant contracts in place, suppliers have the ability to access the latest requirements documentation at any time or refer to previous versions. This flexibility also extends to the data provided by suppliers. Therefore, this implies that iterative development of the system through the Dataspace is feasible during collaboration. Furthermore, different versions of the same model can be accessed and utilized through the Dataspace based on corresponding (maybe the same) contracts and usage policies. Access to several versions of the models (including older versions) can be important for configuration management [52].

Due to the Dataspace's functionality in achieving targeted control over the utilization and release of data, packages (in the context of exchanging SysML v2 models) are suitable to be provided and exchanged as individual items in the form of artifacts within the Dataspace. As an advantage of the SysML v2 model, different exchanged model data can be integrated by importing them into the same file. Additionally, these model data can be extended to incorporate new features or definitions for addressing emerging requirements. Consequently, updates to the model, in certain contexts, can be accomplished by exchanging concise updates based on the existing data. Dataspace serves as a highly suitable platform for facilitating these data exchanges in a secure manner, promoting collaboration in the development of complex mechatronic products.

3.4. Collaboration Approach

Figure 9 shows a possible approach for model-based collaboration in product development between OEM and supplier using a dataspace. On the OEM side, the general definition package is specialized in the black box under the concrete context of a new product, which is the basis for the development of the overall system ("Definition Package" is explained in detail in Section 2.1). The OEM decomposes the system model for sub-system requirements during the development of the overall system. One possible solution for the decomposition is the top-down process [85]. These requirements are subsequently transmitted to the supplier side via the Dataspace, along with the OEM General Definition Package.

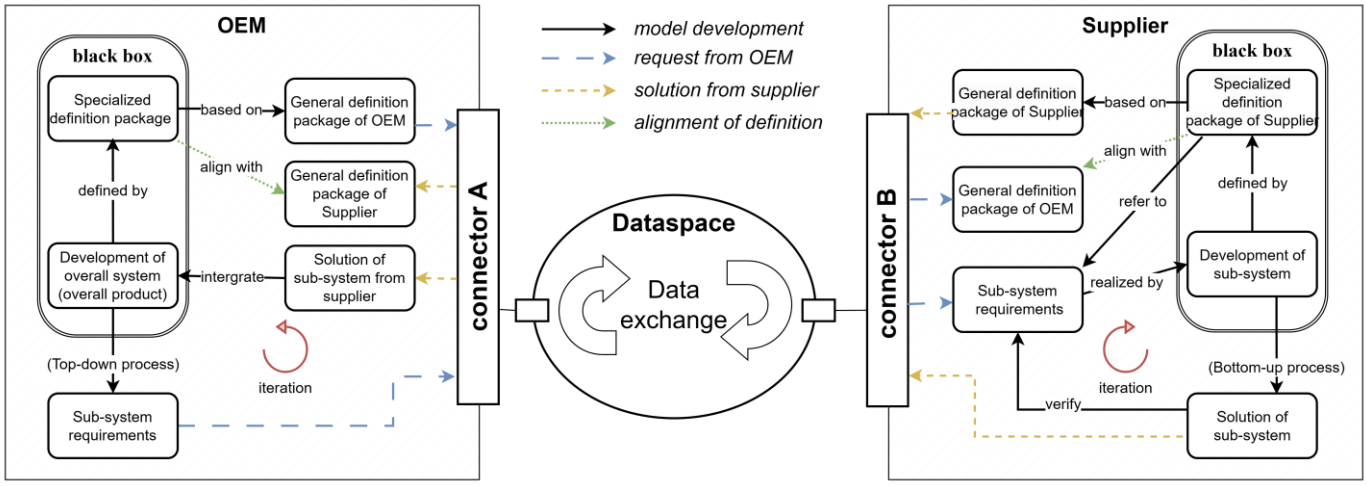

Figure 9. Overall approach to model-based collaboration using Dataspace.

On the supplier side, the sub-system is developed according to the OEMs requirements. The sub-system must be verified according to the supplier's requirements, i.e., a comparison is made between the required and as-is properties [85]. When creating the model-based specification of the sub-system, the supplier should use the definition package provided by the OEM for compatibility, which may need to be specifically extended.

The solution from the supplier is provided to the OEM via the Dataspace, enabling the integration of the supplier's solution into the overall system [15]. With each successful integration, potential new requirements may emerge, necessitating further solutions from the supplier. Consequently, this collaborative approach results in the continuous enhancement and refinement of the overall product through a collaborative development process.

In the practical process, there may be multiple levels of suppliers and several suppliers on each sub-system level (possibly even multilateral collaboration), but due to space limitations in this contribution, this is not discussed for now.

4. Implementation

In this paper, the implementation of the approach outlined in the preceding section involves the applications of SysML v2 modeling and Dataspace. This implementation is realized through the utilization of two open-source repositories available on GitHub.

- System modeling with SysML v2 in Eclipse

In accordance with the installation instructions for Eclipse outlined in the SysML-v2-Release repository (version 2023-11) [48], engineers can install the corresponding plugins and tools within Eclipse. This enables the creation of SysML files and modeling using SysML v2. Eclipse can automatically render corresponding diagrams based on the textual models. Furthermore, the model library provided by the SysML-v2-Release repository includes units or standards for various domains, facilitating engineers in model development.

- Data exchange with Dataspace in the IDS-testbed

The IDS-testbed repository [90] offers a composition of components that, in accordance with the International Dataspaces Association (IDSA) [91] regulations, can establish a Dataspace capable of secure and sovereign data exchange. Serving as a Minimum Viable Dataspace (MVDS), it incorporates sufficient features to serve as a starting point for experimenters and engineers to create a functional Dataspace, that can be customized and extended as needed to meet specific requirements [92].

Upon configuring Eclipse and IDS-testbed, the implementation of the approach can be achieved through the following steps:

1. Modeling by Data Provider in Eclipse:

The data provider initiates the process by performing system modeling in Eclipse. Leveraging SysML v2 plugins and tools (e.g., PlantUML for graphical visualization), the data provider creates a SysML file with <.sysml> filename extensions to define the content of the system. Here is a simple example (all the used example models are as Supplementary Materials provided on GitHub [93]):

Figure 10 illustrates a simple model using SysML v2 within Eclipse. On the left is the text notation encoded by the author. The example definition package imports the International System of Quantities (ISQ) and the International System of Units (SI) stored in the model library, defining the names and characteristics of the parts. The temperature and mass properties of the parts refine the temperature and mass properties specified in the ISQ standard. The example usage package utilizes the imported example definition package to describe the system and its constituent structures, corresponding to the composition relationship in SysML 1.x. Parts inherit properties from their definitions and can redefine them at any level of nesting. Eclipse automatically generates the graphical representation on the right through textual notation. Different packages in the example can be saved in separate <.sysml> files and imported into other files, similar to how ISQ and SI packages are handled.

2. Data Provider: import the model into Dataspace.

After exporting model files from Eclipse, the subsequent step involves uploading these files to a server (see Figure 6). This facilitates the Dataspace connector by utilizing an API to remotely import the files as artifacts. The following outlines a straightforward operational example:

a. Upload the model to a server (e.g., a PDM/PLM tool for the companies, but using a simple web server in the investigation from the author's side also works);
b. Create a post request for the model address to the Dataspace connector address in Postman (one of the most famous API platforms for building and using APIs).
c. Send the request.

3. Data Provider: configures model data provision.

In accordance with the previous section, Dataspace Connector's data model necessitates that data providers establish metadata descriptions pertaining to the model data.

Additionally, data providers are required to define usage policies and contracts associated with the exchanged data. Subsequently, the data provider is expected to register and add resources, including representations, artifacts, and the corresponding contracts, within the relevant catalog.

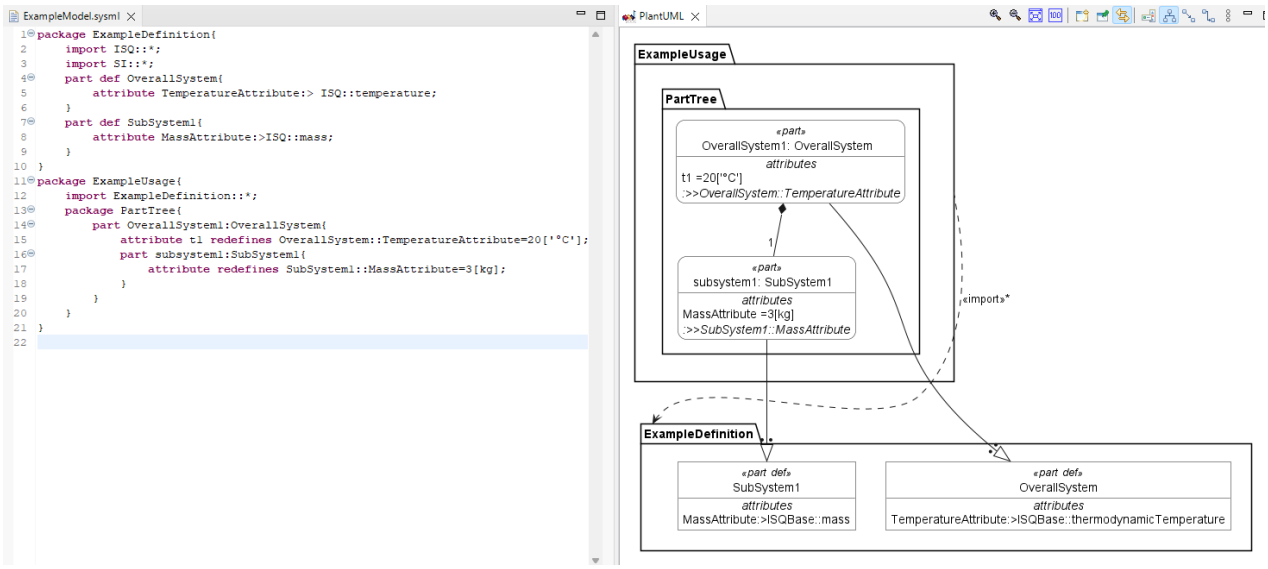

Figure 10. Simple example of SysML v2 model. (Symbol "*" in the Figure is used in SysML to import the package and its underlying sub-packages).

4. Data Consumers request and obtain model data.

By requesting descriptions of resources and relevant expressions from data providers, data consumers can determine the necessary artifacts corresponding to specific model data. If consumers submit contracts with the required artifact ID to data providers, either drafted by the providers or modified by the consumers, it signifies the consumer's agreement to all terms, including stipulated timeframes and usage policies outlined in the contract. Through contract comparison, data providers furnish feedback to consumers regarding data access permissions. Once granted permission, consumers can consistently access data within the specified constraints defined by the contract.

5. Data Consumers obtain a model and import it into Eclipse for subsequent modeling.

After acquiring the data, data consumers should convert the obtained data files to the <.sysml> filename extension using an extended API or program (since this open-source Dataspace cannot export files with specific extensions). Subsequently, by importing the SysML model files into Eclipse, engineers can engage in further collaborative development of the model. To import packages into different projects, engineers need to select the appropriate project references in the project's properties.

Through these steps, the entire process, from system modeling to data provision and requests, have been implemented.

5. Example

The approach developed is applied in this contribution using a precision engineering application, specifically for a load cell. For this example, an OEM develops a measurement system that enables precise force measurement capabilities (see also [85]). To address this requirement, the OEM collaborates with a supplier specializing in load cell solutions.

The measurement system is engineered to quantify force accurately, tailored to specific application needs. It integrates various sub-systems (components) and functionalities. The load cell, as one of the key sub-systems provided by the supplier, operates based on fundamental principles such as the piezoresistive effect.

In the scope of this contribution, the collaboration between the OEM and the supplier focuses on aspects of model development and model exchange. On the one hand, the measurement system is modeled based on the requirements for accuracy class, the range of measuring force, and the temperature range. On the other hand, for research and modeling purposes, the parameters of the load cell are settled by the author based on the OIML standard [94]. Furthermore, models from both OEMs and suppliers are simplified and maintained with a significant degree of consistency to avoid ambiguity in description. This, therefore, cannot represent the complete model alignment process in real collaborative scenarios, such as differences in modeling the architecture or selecting terminologies for some ports, features, or units. The full models are provided on GitHub [93].

At the beginning of this CMBSE example approach, the OEM should model the measurement system and provide the requirements and boundary conditions for the development of the scope of supply.

Figure 11 shows the general definition and specific definition for the force measurement system from the OEM side. On the top side of Figure 11, the overall system is defined at a high-level with its parts, ports, requirements, etc. The corresponding graphical notation is shown below, which is generated directly in Eclipse. In the force measurement system definition file, the overall system definition from the general definition package is specified as force measurement system, which not only inherits all the attributes defined already but is also extended with the new attribute "load cell number" and the state machine about the required state.

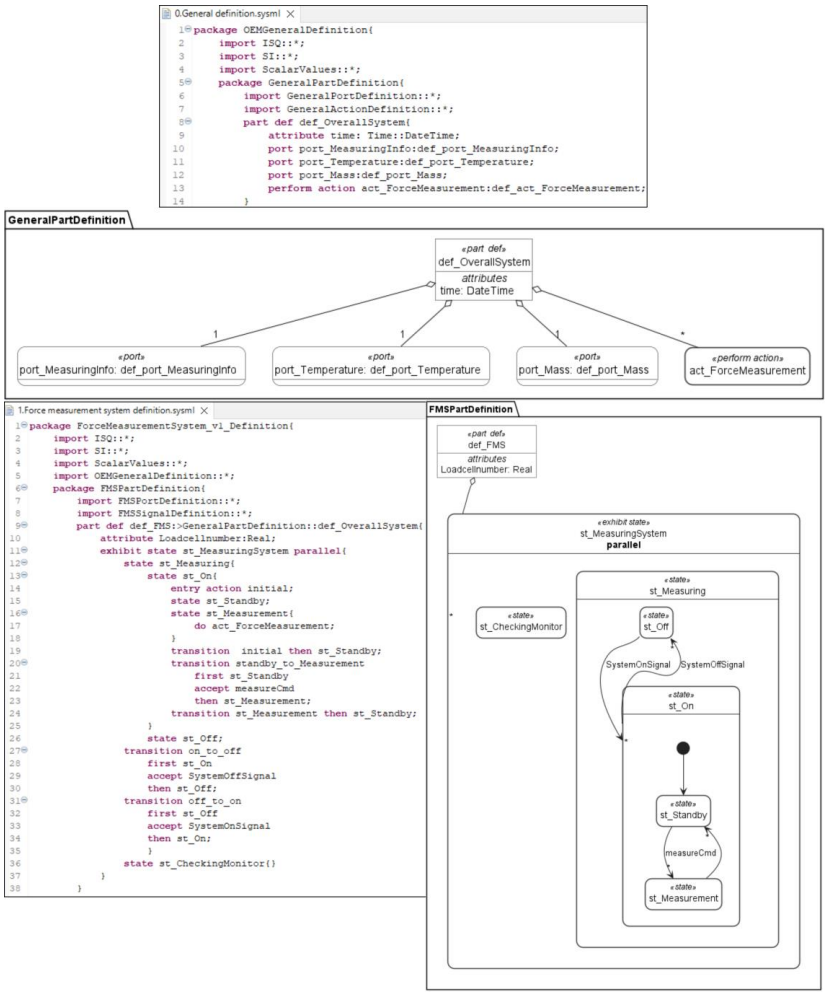

Figure 11. OEM general definition and specific definition for the force measurement system. (Symbol "*" in the Figure is used in SysML to import the package and its underlying sub-packages).

OEMs can model the detailed system structure using the specified definitions. In this contribution, the modeling approach is based on the MagicGrid methodology [39], including the design of the system structure, system behavior, requirements, parameters, etc. Furthermore, the requirements for the load cell are derived based on the referred definitions and usages. These requirements, as black-box descriptions, contain specific information for the supplier to provide the solution for the load cell, and some information such as the state machine of the force measurement system is not included.

After exporting the SysML model with the relevant requirements and definition packages from SysML v2 tools to Data Management Tools, OEMs can upload them as artifacts to their connectors (in the example in Figure 9, Connector A) in the Dataspace. Figure 12 shows the API in Postman and its response. In the response, a link is created for the artifact, which is always aligned with the up-to-date SysML model at this address.

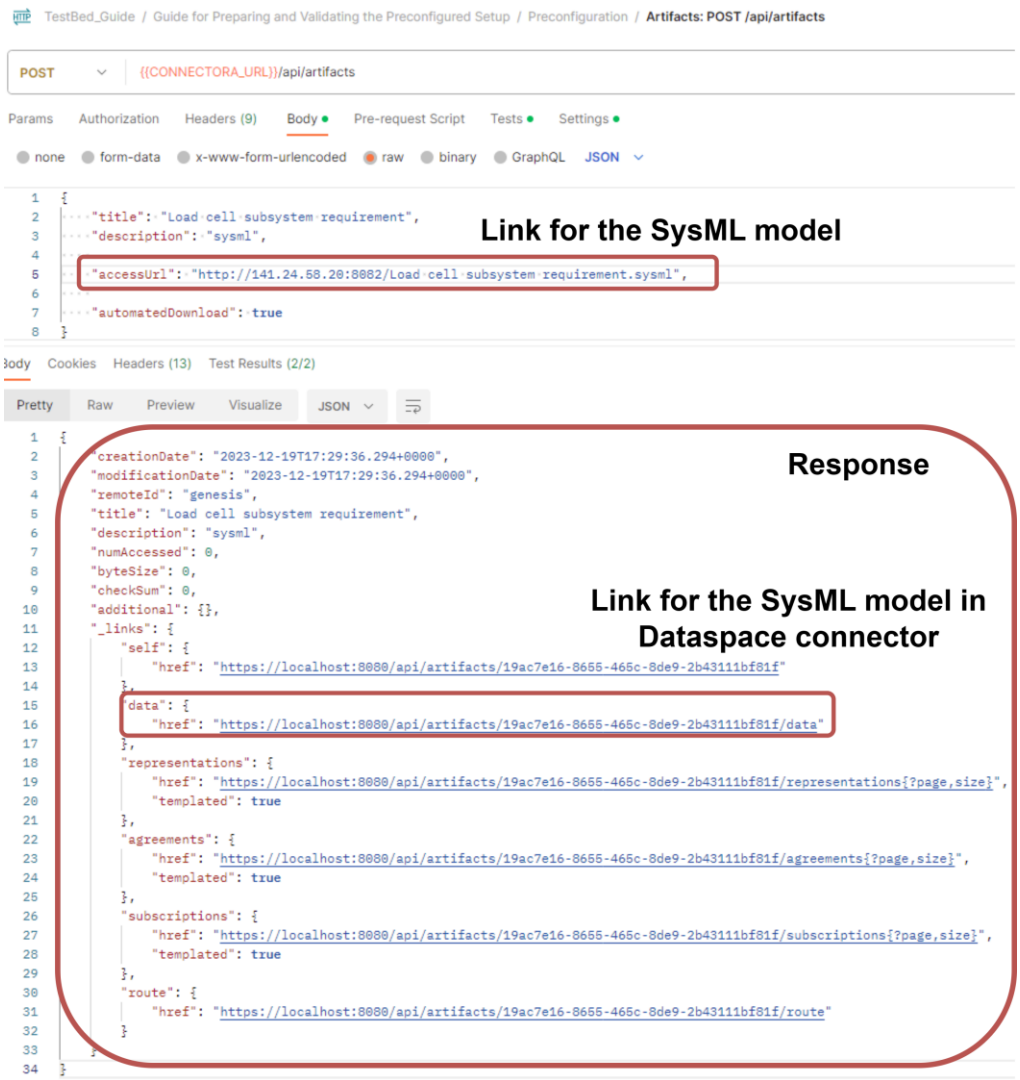

Figure 12. Dataspace API for data upload in Postman.

When the artifacts aligned with its usage policy are organized in a specific catalog, the supplier can query them and start negotiations with the OEM. As Figure 13 shows, the supplier must provide the usage policy ("Provider_rule" with red box in Figure 13) as consent to all the contents of the requested artifact link ("Provider_artifact" with red box in Figure 13). With the received agreement ID, the supplier obtained access to the model within the contract period.

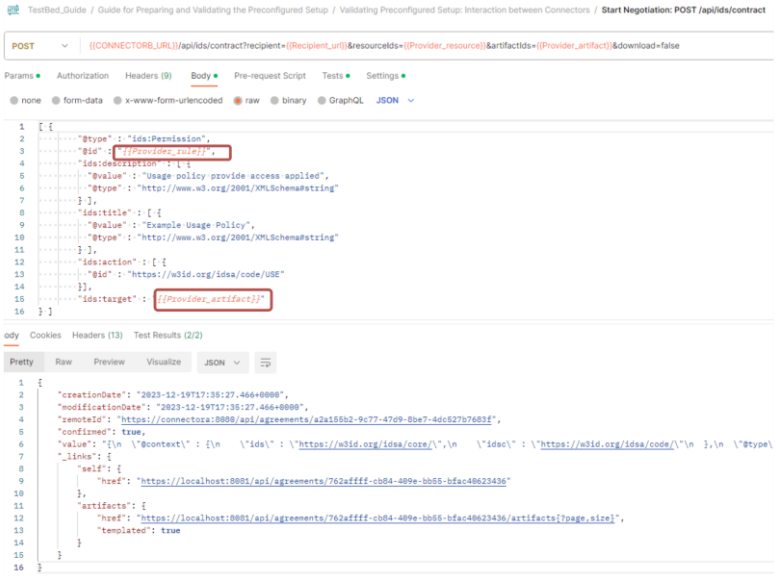

Figure 13. Dataspace API for negotiation between OEM and supplier. ("Provider_rule" and "Provider_artifact" are marked with red box).

On the supplier side, the SysML model with the requirements and definitions can be accessed after the negotiation, and the development process can be carried out. For the solution specifications of the load cell sub-system, the definitions used in the load cell are from both the OEMs and supplier's general definitions and further extended, as shown in Figure 14. Moreover, the as-is specifications in Figure 14, such as the parameters and interfaces of the load cell, are described and connected to satisfy the requirements of the OEM. After development, the supplier makes its specification model as the solution for the load cell sub-system available in the same way (see Figure 9). This is no longer described in detail in the contribution.

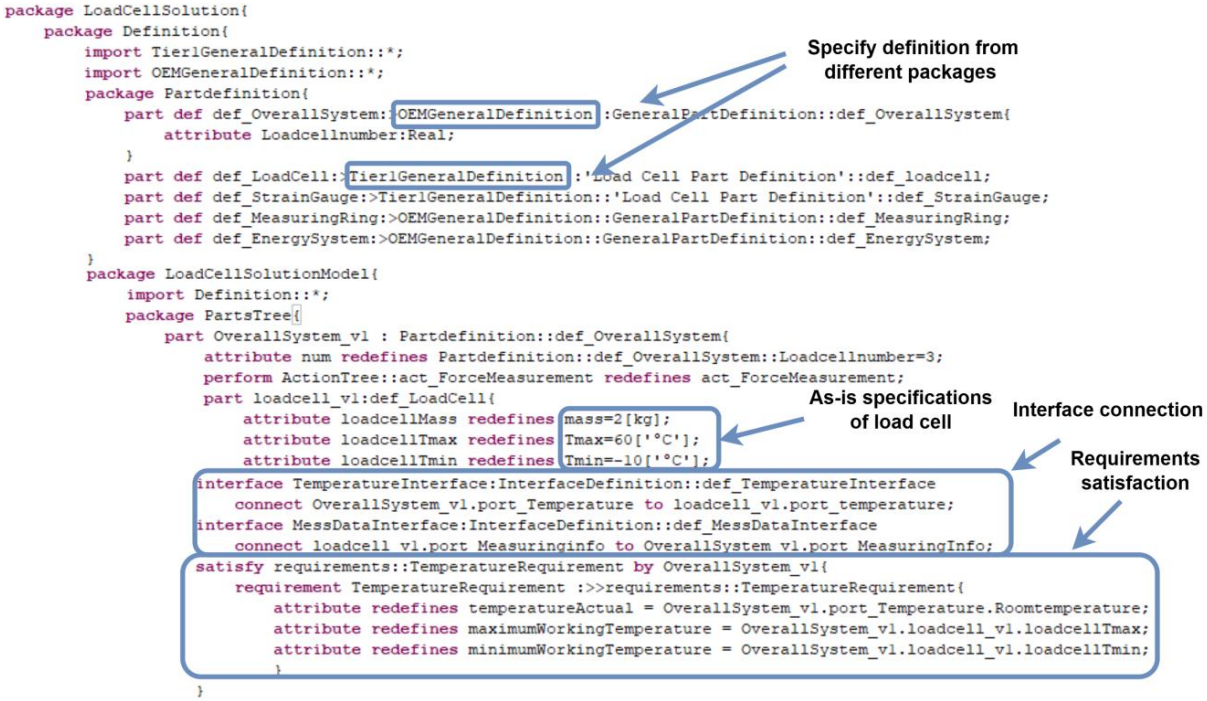

Figure 14. Solution for load cell sub-system from supplier. (Symbol "*" in the Figure is used in SysML to import the package and its underlying sub-packages).

6. Conclusions and Outlook

Collaborative Model-based Systems Engineering between companies is gaining increasing significance. The modeling language SysML v2 serves as an important basis for facilitating the model data exchange in the collaboration. As data exchange becomes more prevalent, the secure and trustworthy sharing of data are progressively gaining importance. Against this background, the Dataspaces approach is coming into the focus of discussions. This contribution aims to investigate the possibilities and challenges of combining modeling with SysML v2 and data exchange via Dataspaces. Based on systemic approaches, a modeling concept for decomposing the system into sub-systems is developed as a basis for the exchange. In addition, based on the analysis of collaboration processes in the context of Systems Engineering, an architectural approach with a SysML editor and Dataspace for the exchange is elaborated. The implementation of the architecture is based on the open-source solutions "SysML-v2-Release" in the Eclipse environment and "IDS-testbed". The investigations are based on an application example from precision engineering. The potential and challenges are discussed.

SysML v2 offers good preconditions for exchange with the new metamodel and the combined textual and graphical notation, among other things. Structuring via packages (already possible in SysML 1.x) is an essential basis for extracting sub-models (see research question 1a). Packages can be exchanged efficiently on the basis of textual modeling and the import options in SysML v2. Modeling guidelines are necessary to enable the extraction of sub-models in a targeted manner. These must make it possible for the packages to be exchanged independently (or only with references to other packages that are also exchanged) without references to the outside being lost during the exchange. This concerns the structuring of the models, the definition of relations (so that the models can be extracted), and the import of packages (possibly entire SysML models). This contribution also shows the potential of definition packages. Concrete modeling guidelines are also necessary for the definition packages and the formalization of requirements so that cross-company collaboration is possible. This also applies to context-specific extensions for the definition packages.

Dataspaces enable the exchange of models, even if the partners' development processes are different. The processes are considered in the partners' data management tools (e.g., in the PDM or PLM system). Decoupling of the individual processes takes place via Dataspaces. Based on the modeling in the editor, the SysML models are saved in the company's data management tool, and their specific links are uploaded and provided in a secure way via the dataspace (see research question 1b).

The contribution discusses a possible cooperation process between the OEM and supplier. Both provide the required models according to the distribution of tasks (see research question 1c). Future work should address the relevant coupling points for synchronization, including the management and alignment of disruptive changes from new versions of sub-systems. As of today, the collaboration process can be linked to milestones. Closer collaboration with short iteration loops is possible and the subject of research. This contribution assumes a state-of-the-art collaboration process between companies. Dataspace's search and access options also offer the potential for supporting open development or innovation processes. This potential must be discussed in further investigations.

The application of the approach described in this contribution (a combination of SysML v2 and Dataspace) shows significant potential with regard to the flexible exchange of sub-models and their integration into the models of the respective partner. With the help of Dataspaces, access to the models can be controlled. The main challenge is the necessary modeling guidelines from the engineers' viewpoint. In addition, the IT data infrastructure must enable access to the models provided in the data management tools across company boundaries. This provision must be supported administratively. In order to make the process between the SysML editor, data management tool, and Dataspace as efficient as possible, workflows are required that must be implemented in the connectors.

Supplementary Materials: The following supporting information can be downloaded at: https://github.com/ziruili-tu-ilmenau/CMBSE (accessed on 20 December 2023).

Author Contributions: Methodology, Z.L. and S.H.; Writing—original draft, Z.L. and S.H.; Writing—review and editing, Z.L., S.H. and F.F. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Data are contained within the article.

Acknowledgments: We acknowledge support for the publication costs by the Open Access Publication Fund of the Technische Universität Ilmenau.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Albers, A.; Haug, F.; Fahl, J.; Hirschter, T.; Reinemann, J.; Rapp, S. Customer-Oriented Product Development: Supporting the Development of the Complete Vehicle through the Systematic Use of Engineering Generations. In Proceedings of the 2018 IEEE International Systems Engineering Symposium (ISSE), Rome, Italy, 1–3 October 2018; pp. 1–8.
2. Husung, S.; Weber, C.; Mahboob, A. Model-Based Systems Engineering—A New Way for Function-Driven Product Development. In *Design Methodology for Future Products—Data Driven, Agile and Flexible*, 1st ed.; Krause, D., Heyden, E., Eds.; Springer International Publishing: Cham, Switzerland, 2022; ISBN 978-3-030-78367-9.
3. acatech. *Cyber-Physical Systems—Driving Force for Innovation in Mobility, Health, Energy and Production*; acatech: Munich Germany, 2021.
4. Eigner, M.; Dickopf, T.; Schneider, M.; Schulte, T. mecPro2-A holistic concept for the model-based development of cybertronic systems. In Proceedings of the 21st International Conference on Engineering Design (ICED 17), Vancouver, BC, Canada, 21–25 August 2017; pp. 378–388.
5. Waschle, M.; Wolter, K.; Bause, K.; Behrendt, M.; Albers, A. Considering Functional Safety—Supporting the development of automated driving vehicles by the use of Model-Based Systems Engineering. In Proceedings of the 2022 17th Annual System of Systems Engineering Conference (SOSE), Rochester, NY, USA, 7–11 June 2022; IEEE: New York, NY, USA, 2022; pp. 275–280, ISBN 978-1-6654-9623-0.
6. Trauer, J.; Schweigert-Recksiek, S.; Engel, C.; Spreitzer, K.; Zimmermann, M. What is a Digital Twin?—Defintions and Insights from an Industrial Case Study in Technical Product Development. In Proceedings of the 16th International Design Conference (DESIGN 2020), Cavtat, Croatia, 26–29 October 2020; pp. 757–766.
7. Mahboob, A. Modelling and Use of SysML Behaviour Models for Achieving Dynamic Use Cases of Technical Products in Different VR-Systems. Ph.D. Thesis, Technische Universität Ilmenau, Ilmenau, Germany, 2021.
8. Duehr, K.; Heimicke, J.; Breitschuh, J.; Spadinger, M.; Kopp, D.; Haertenstein, L.; Albers, A. Understanding Distributed Product Engineering: Dealing with Complexity for Situation- and Demand-Oriented Process Design. *Procedia CIRP* **2019**, *84*, 136–142. [CrossRef]
9. Larsson, A.; Törlind, P.; Karlsson, L.; Mabogunje, A.; Leifer, L.; Larsson, T.; Elfström, B.-O. Distributed Team Innovation—A Framework for Distributed Product Development. In Proceedings of the 14th International Conference on Engineering Design, Stockholm, Sweden, 19–21 August 2003.
10. Borsato, M.; Peruzzini, M. Collaborative Engineering. In *Concurrent Engineering in the 21st Century*; Stjepandić, J., Wognum, N.J.C., Verhagen, W., Eds.; Springer International Publishing: Cham, Switzerland, 2015; pp. 165–196, ISBN 978-3-319-13775-9.
11. Lu, S.-Y.; Elmaraghy, W.; Schuh, G.; Wilhelm, R. A Scientific Foundation of Collaborative Engineering. *CIRP Annals* **2007**, *56*, 605–634. [CrossRef]
12. Steck, M.; Husung, S. Surrogate-Based Calculation Method for Robust Design Optimization Considering the Fatigue Probability for Variable Service Loads of eBike Drive Units. *Designs* **2024**, *8*, 4. [CrossRef]
13. Belkadi, F.; Messaadia, M.; Bernard, A.; Baudry, D. Collaboration management framework for OEM—Suppliers relationships: A trust-based conceptual approach. *Enterp. Inf. Syst.* **2017**, *11*, 1018–1042. [CrossRef]
14. Bogaschewsky, R.; Eßig, M.; Lasch, R.; Stölzle, W. (Eds.) *Supply Management Research*; Gabler: Wiesbaden, Germany, 2010; ISBN 978-3-8349-2057-7.
15. prostep ivip Association. *prostep ivip Recommendation 2023 PSI 28—SysML WF/IF*; prostep ivip Association: Darmstadt, Germany, 2023; Available online: https://www.prostep.org/shop/detail?ai[action]=detail&ai[controller]=Catalog&ai[d_name]=sysml_rec_wfif&ai[d_pos]=7 (accessed on 20 December 2023).
16. Schilli, B.; Dai, F. Collaborative life cycle management between suppliers and OEM. *Comput. Ind.* **2006**, *57*, 725–731. [CrossRef]
17. Schmidt, M.M.; Zimmermann, T.C.; Stark, R. Systematic Literature Review of System Models for Technical System Development. *Appl. Sci.* **2021**, *11*, 3014. [CrossRef]
18. Fritz, J.; Zingel, C.; Kokko, J.; Lenardon, G.; Brier, B. Systems Engineering Methods and Tools. In *Systems Engineering for Automotive Powertrain Development*, 1st ed.; Hick, H., Küpper, K., Sorger, H., Eds.; Springer International Publishing: Cham, Switzerland, 2021; pp. 271–302, ISBN 9783319996295.

19. Husung, S.; Weber, C.; Mahboob, A. Integrating Model-Based Design of Mechatronic Systems with Domain-Specific Design Approaches. In Proceedings of the 17th International Design Conference, Cavtat, Croatia, 23–26 May 2022; pp. 1895–1904.
20. Ropohl, G. *Allgemeine Systemtheorie: Einführung in Transdisziplinäres Denken*; Edition Sigma: Berlin, Germnay, 2012; ISBN 9783845269153.
21. Hubka, V.; Eder, W.E. *Theory of Technical Systems*; Springer: Berlin/Heidelberg, Germany, 1988; ISBN 978-3-642-52123-2.
22. Ariyo, O.O.; Eckert, C.M.; Clarkson, P.J. Hierarchical decompositions for complex product representation. In Proceedings of the 10th International Design Conference, Dubrovnik, Croatia, 19–22 May 2008; pp. 737–744.
23. Browning, T.R. Applying the design structure matrix to system decomposition and integration problems: A review and new directions. *IEEE Trans. Eng. Manag.* **2001**, *48*, 292–306. [CrossRef]
24. Friedenthal, S. *A Practical Guide to SysML: The Systems Modeling Language*, 3rd ed.; Elsevier Science: San Francisco, CA, USA, 2014; ISBN 978-0128002025.
25. Morkevicius, A.; Aleksandraviciene, A.; Krisciuniene, G. From UAF to SysML: Transitioning from System of Systems to Systems Architecture. In Proceedings of the INCOSE International Symposium 2021, Virtual Event, 16–19 November 2021; pp. 585–598.
26. Schumacher, T.; Kaczmarek, D.; Inkermann, D.; Lohrengel, A. Fostering Model Consistency in Interdisciplinary Engineering by Linking SysML and CAD-Models. In Proceedings of the 2022 IEEE International Symposium on Systems Engineering (ISSE), Vienna, Austria, 24–26 October 2022; IEEE: New York, NY, USA, 2022; pp. 1–7, ISBN 978-1-6654-8182-3.
27. Leite, J.; Oquendo, F.; Batista, T. SysADL: A SysML Profile for Software Architecture Description. In *Software Architecture*; Hutchison, D., Kanade, T., Kittler, J., Kleinberg, J.M., Mattern, F., Mitchell, J.C., Naor, M., Nierstrasz, O., Pandu Rangan, C., Steffen, B., et al., Eds.; Springer: Berlin/Heidelberg, Germany, 2013; pp. 106–113, ISBN 978-3-642-39030-2.
28. Schamai, W.; Fritzson, P.; Paredis, C.; Pop, A. Towards Unified System Modeling and Simulation with ModelicaML: Modeling of Executable Behavior Using Graphical Notations. In Proceedings of the 7th International Modelica Conference, Como, Italy, 20–22 September 2009; Linköping University Electronic Press: Linköping, Sweden, 2009; pp. 612–621.
29. Mahboob, A. C. Pulse: An industrial demonstrator for a digital twin powered by MBSE for achieving digital continuity during the complete development process. In Proceedings of the 60th Ilmenau Scientific Colloquium, Ilmenau, Germany, 4–8 September 2023.
30. Hick, H.; Bajzek, M.; Faustmann, C. Definition of a system model for model-based development. *SN Appl. Sci.* **2019**, *1*, 1074. [CrossRef]
31. Trujillo, A.; de Weck, O.L.; Madni, A.M. An MBSE Approach Supporting Technical Inheritance and Design Reuse Decisions. In *ASCEND 2020, Virtual Event*; American Institute of Aeronautics and Astronautics: Reston, Virginia, 2020; ISBN 978-1-62410-608-8.
32. Mandel, C.; Böning, J.; Behrendt, M.; Albers, A. A Model-Based Systems Engineering Approach to Support Continuous Validation in PGE—Product Generation Engineering. In Proceedings of the IEEE ISSE International Symposium on Systems Engineering 2021, Vienna, Austria, 13 September–13 October 2021.
33. Amara, N.; Landry, R.; Traoré, N. Managing the protection of innovations in knowledge-intensive business services. *Res. Policy* **2008**, *37*, 1530–1547. [CrossRef]
34. Friedenthal, S. Requirements for the Next Generation Systems Modeling Language (SysML® v2). *Insight* **2018**, *21*, 21–25. [CrossRef]
35. Curry, E.; Scerri, S.; Tuikka, T. Data Spaces: Design, Deployment, and Future Directions. In *Data Spaces*; Curry, E., Scerri, S., Tuikka, T., Eds.; Springer International Publishing: Cham, Switzerland, 2022; pp. 1–17, ISBN 978-3-030-98635-3.
36. Estefan, J.A. Survey of model-based systems engineering (MBSE) methodologies. *INCOSE MBSE Focus Group* **2007**, *25*, 1–12.
37. Dori, D. *Model-Based Systems Engineering with OPM and SysML*; Springer: New York, NY, USA, 2016; ISBN 978-1-4939-3294-8.
38. SysML.org. SysML FAQ: What Is SysML? Available online: https://sysml.org/sysml-faq/what-is-sysml.html (accessed on 4 January 2024).
39. Aleksandraviciene, A.; Morkevicius, A. (Eds.) *MagicGrid® Book of Knowledge—A Practical Guide to Systems Modeling Using MagicGrid*; Vitae Litera, UAB: Kaunas, Lithuania, 2018.
40. Morkevicius, A.; Aleksandraviciene, A.; Mazeika, D.; Bisikirskiene, L.; Strolia, Z. MBSE Grid: A Simplified SysML-Based Approach for Modeling Complex Systems. In Proceedings of the INCOSE International Symposium, Adelaide, Australia, 15–20 July 2017.
41. Petitdemange, F.; Borne, I.; Buisson, J. Modeling System of Systems configurations. In Proceedings of the 2018 13th Annual Conference on System of Systems Engineering (SoSE), Paris, France, 19–22 June 2018; IEEE: New York, NY, USA, 2018; pp. 392–399, ISBN 978-1-5386-4876-6.
42. Husung, S.; Weber, C.; Mahboob, A.; Kleiner, S. Using Model-Based Systems Engineering for Need-Based and Consistent Support of the Design Process. In Proceedings of the International Conference on Engineering Design (ICED21), Gothenburg, Sweden, 16–20 August 2021; pp. 3369–3378.
43. GfSE/SAF. GfSE/SAF-Specification: The Specification for the System Architecture Framework (SAF). Available online: https://github.com/GfSE/SAF-Specification (accessed on 4 January 2024).
44. Inkermann, D.; Huth, T.; Vietor, T.; Grewe, A.; Knieke, C.; Rausch, A. Model-Based Requirement Engineering to Support Development of Complex Systems. In Proceedings of the Procedia CIRP 2019, Póvoa de Varzim, Portgal, 8–10 May 2019; pp. 239–244.
45. Böhm, W. *Model-Based Engineering of Collaborative Embedded Systems: Extensions of the SPES Methodology*; Springer: Berlin/Heidelberg, Germany, 2021.

46. Bajaj, M.; Friedenthal, S.; Seidewitz, E. Systems Modeling Language (SysML v2) Support for Digital Engineering. *Insight* **2022**, *25*, 19–24. [CrossRef]
47. Systems Modeling Language (SysML): Version 2.0 Beta 1. Part 1: Language Specification. Available online: https://www.omg.org/spec/SysML/2.0/Beta1/Language/PDF (accessed on 11 December 2023).
48. Systems-Modeling/SysML-v2-Release: The Latest Incremental Release of SysML v2. Start Here. Available online: https://github.com/Systems-Modeling/SysML-v2-Release (accessed on 11 December 2023).
49. Gogan, L.M.; Popescu, A.-D.; Duran, V. Misunderstandings between Cross-cultural Members within Collaborative Engineering Teams. *Procedia—Soc. Behav. Sci.* **2014**, *109*, 370–374. [CrossRef]
50. Barbosa, C.E.; Trindade, G.; Epelbaum, V.J.; Chang, J.G.; Oliveira, J.; Rodrigues Neto, J.A.; de Souza, J.M. Challenges on designing a distributed collaborative UML editor. In Proceedings of the 2014 IEEE 18th International Conference on Computer Supported Cooperative Work in Design (CSCWD 2014), Hsinchu, Taiwan, 21–23 May 2014; Hou, J.-L., Ed.; IEEE: Piscataway, NJ, USA, 2014; pp. 59–64, ISBN 978-1-4799-3776-9.
51. *ISO/IEC/IEEE 24748-1*; Systems and Software Engineering—Life Cycle Management: Part 1:Guidelines for Life Cycle Management. IEEE: Piscataway, NJ, USA, 2018.
52. *ISO/IEC/IEEE 15288:2023*; Systems and Software Engineering: System Life Cycle Processes. IEEE: Piscataway, NJ, USA, 2023.
53. Wouters, L.; Creff, S.; Bella, E.E.; Koudri, A. Collaborative systems engineering: Issues & challenges. In Proceedings of the 2017 IEEE 21st International Conference on Computer Supported Cooperative Work in Design (CSCWD); Wellington, New Zealand, 26–28 April 2017, IEEE: Piscataway, NJ, USA, 2017; pp. 486–491, ISBN 978-1-5090-6199-0.
54. Zerwas, T.; Jacobs, G.; Kowalski, J.; Husung, S.; Gerhard, D.; Rumpe, B.; Zeman, K.; Vafaei, S.; König, F.; Höpfner, G. Model Signatures for the Integration of Simulation Models into System Models. *Systems* **2022**, *10*, 199. [CrossRef]
55. Heber, D.T.; Groll, M.W. A Meta-Model to Connect Model-based Systems Engineering with Product Data Management by Dint of the Blockchain. In Proceedings of the 2018 International Conference on Intelligent Systems (IS), Funchal-Madeira, Portugal, 25–27 September 2018; IEEE: Piscataway, NJ, USA, 2018; pp. 280–287, ISBN 978-1-5386-7097-2.
56. Wang, C. MBSE-Compliant Product Lifecycle Model Management. In Proceedings of the 2019 14th Annual Conference System of Systems Engineering (SoSE), Anchorage, AK, USA, 19–22 May 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 248–253, ISBN 978-1-7281-0457-7.
57. Eigner, M.; Gilz, T.; Denger, A.; Fritz, J. Applicability of model-based system lifecycle management for cyber-physical systems. In Proceedings of the 14th Mechatronics Forum International Conference, Karlstad, Sweden, 16–18 June 2014; pp. 294–302.
58. Gerhard, D.; Salas Cordero, S.; Vingerhoeds, R.; Sullivan, B.P.; Rossi, M.; Brovar, Y.; Menshenin, Y.; Fortin, C.; Eynard, B. MBSE-PLM Integration: Initiatives and Future Outlook. In *Product Lifecycle Management. PLM in Transition Times: The Place of Humans and Transformative Technologies*; Noël, F., Nyffenegger, F., Rivest, L., Bouras, A., Eds.; Springer Nature Switzerland: Cham, Switzerland, 2023; pp. 165–175, ISBN 978-3-031-25181-8.
59. Pol, G.; Merlo, C.; Legardeur, J.; Jared, G. Implementation of collaborative design processes into PLM systems. *Int. J. Prod. Lifecycle Manag.* **2008**, *3*, 279. [CrossRef]
60. Messaadia, M.; Belkadi, F.; Eynard, B.; Sahraoui, A.-E.-K. System Engineering and PLM as an integrated approach for industry collaboration management. *IFAC Proc. Vol.* **2012**, *45*, 1135–1140. [CrossRef]
61. Tilioua, N.; Bennouna, F.; Chalh, Z. Using Internet of Things to Increase Efficient Collaboration in PLM. In *Digital Technologies and Applications*; Motahhir, S., Bossoufi, B., Eds.; Springer International Publishing: Cham, Switzerland, 2021; pp. 825–833, ISBN 978-3-030-73881-5.
62. Mas, F.; Arista, R.; Oliva, M.; Hiebert, B.; Gilkerson, I.; Rios, J. A Review of PLM Impact on US and EU Aerospace Industry. *Procedia Eng.* **2015**, *132*, 1053–1060. [CrossRef]
63. Otto, B.; Jarke, M. Designing a multi-sided data platform: Findings from the International Data Spaces case. *Electron. Mark.* **2019**, *29*, 561–580. [CrossRef]
64. Halevy, A.; Franklin, M.; Maier, D. Principles of dataspace systems. In Proceedings of the SIGMOD/PODS06: International Conference on Management of Data and Symposium on Principles Database and Systems, Chicago, IL, USA, 26–28 June 2006; Gottlob, G., van den Bussche, J., Eds.; ACM: New York, NY, USA, 2006; pp. 1–9, ISBN 1595933182.
65. Sarma, A.D.; Dong, X.; Halevy, A.Y. Data Modeling in Dataspace Support Platforms. *Concept. Model. Found. Appl.* **2009**, *5600*, 122–138. [CrossRef]
66. Usmani, A.; Khan, M.J.; Breslin, J.G.; Curry, E. Towards Multimodal Knowledge Graphs for Data Spaces. In *Companion, Proceedings of the ACM Web Conference 2023, WWW '23: The ACM Web Conference 2023, Austin, TX, USA, 20 April–4 May 2023*; Ding, Y., Tang, J., Sequeda, J., Aroyo, L., Castillo, C., Houben, G.-J., Eds.; ACM: New York, NY, USA, 2023; pp. 1494–1499, ISBN 9781450394192.
67. Song, S.; Chen, L.; Yu, P.S. On data dependencies in dataspaces. In *Data Engineering (ICDE), Proceedings of the 2011 IEEE 27th International Conference on Data Engineering (ICDE 2011), Hannover, Germany, 11–16 April 2011*; IEEE: New York, NY, USA, 2011; pp. 470–481, ISBN 978-1-4244-8959-6.
68. Li, Y.; Meng, X. Supporting context-based query in personal DataSpace. In Proceedings of the 18th ACM Conference on Information and Knowledge Management, Hong Kong, China, 2–6 November 2009; ACM: New York, NY, USA, 2009.
69. International Data Spaces Association. Dataspace Protocol—Working Draft—Terminology. Available online: https://docs.internationaldataspaces.org/ids-knowledgebase/v/dataspace-protocol/overview/terminology (accessed on 14 December 2023).

70. Data Spaces Support Centre. Core Concepts—Glossary. Available online: https://dssc.eu/space/Glossary/176554052/2.+Core+Concepts (accessed on 14 December 2023).
71. Abel, R.; Crispin, N.; Peter, K. What Is a Data Space. Available online: https://gaia-x-hub.de/wp-content/uploads/2022/10/white_paper_definition_dataspace_en.pdf (accessed on 20 December 2023).
72. European Commission. Communication from the Commission to the European Parliament, the Council, the European Economic and Social Committee, and the Committee of the Regions: A European Strategy for Data. Available online: https://eur-lex.europa.eu/legal-content/EN/TXT/HTML/?uri=CELEX:52020DC0066&from=EN (accessed on 15 December 2023).
73. Hellmeier, M.; von Scherenberg, F. A Delimitation of Data Sovereignty from Digital and Technological Sovereignty. In Proceedings of the ECIS 2023, Kristiansand, Norway, 11–16 June 2023.
74. European Commission. Staff Working Document on Data Spaces. Available online: https://digital-strategy.ec.europa.eu/en/library/staff-working-document-data-spaces (accessed on 14 December 2023).
75. International Data Spaces. Data Spaces Radar—International Data Spaces. Available online: https://internationaldataspaces.org/adopt/data-spaces-radar/ (accessed on 14 December 2023).
76. Catena-X. Available online: https://www.t-systems.com/de/de/branchen/automotive/loesungen/catena-x (accessed on 14 December 2023).
77. Smart Connected Supplier Network (SCSN) Process Documentation Manual. Available online: https://smart-connected-supplier-network.gitbook.io/processmanual/ (accessed on 14 December 2023).
78. Volz, F.; Sutschet, G.; Stojanovic, L.; Usländer, T. On the Role of Digital Twins in Data Spaces. *Sensors* **2023**, *23*, 7601. [CrossRef]
79. Usländer, T.; Baumann, M.; Boschert, S.; Rosen, R.; Sauer, O.; Stojanovic, L.; Wehrstedt, J.C. Symbiotic Evolution of Digital Twin Systems and Dataspaces. *Automation* **2022**, *3*, 378–399. [CrossRef]
80. Göhlich, D.; Bender, B.; Fay, T.-A.; Gericke, K. Product requirements specification process in product development. In Proceedings of the 23rd International Conference on Engineering Design (ICED21), Gothenburgh, Sweden, 16–20 August 2021.
81. Pohl, K. *Requirements Engineering: Fundamentals, Principles, and Techniques*; Springer: Berlin/Heidelberg, Germany, 2010.
82. Glinz, M. Improving the Quality of Requirements with Scenarios. In Proceedings of the Second World Congress on Software Quality, Yokohama, Japan, 25–29 September 2000.
83. Douglass, B.P. Agile Stakeholder Requirements Engineering. In *Agile Systems Engineering*; Douglass, B.P., Ed.; Morgan Kaufmann an Imprint of Elsevier: Amsterdam, The Netherlands; Boston, MA, USA; Heidelberg, Germany, 2016; pp. 147–188, ISBN 9780128021200.
84. Ebert, C.; Jastram, M. ReqIF: Seamless Requirements Interchange Format between Business Partners. *IEEE Softw.* **2012**, *29*, 82–87. [CrossRef]
85. Li, Z.; Faheem, F.; Husung, S. Systematic use of model-based solution patterns using the example of a load cell. In Proceedings of the 60th Ilmenau Scientific Colloquium, Ilmenau, Germany, 4–8 September 2023.
86. Seidewitz, E. Systems-Modeling/SysML-v2-Pilot-Implementation: Proof-of-Concept Pilot Implementation of the SysML v2 Textual Notation and Visualization. Available online: https://github.com/Systems-Modeling/SysML-v2-Pilot-Implementation (accessed on 10 December 2023).
87. GitHub. International-Data-Spaces-Association/DataspaceConnector: This Is an IDS Connector Reference Implementation. Available online: https://github.com/International-Data-Spaces-Association/DataspaceConnector (accessed on 20 December 2023).
88. Dataspace Connector. Data Model. Available online: https://international-data-spaces-association.github.io/DataspaceConnector/Documentation/v6/DataModel (accessed on 11 December 2023).
89. 3.5.2 IDS Connector. Available online: https://docs.internationaldataspaces.org/ids-knowledgebase/v/ids-ram-4/layers-of-the-reference-architecture-model/3-layers-of-the-reference-architecture-model/3_5_0_system_layer/3_5_1_identity_provider (accessed on 20 December 2023).
90. GitHub. International-Data-Spaces-Association/IDS-Testbed. Available online: https://github.com/International-Data-Spaces-Association/IDS-testbed (accessed on 11 December 2023).
91. International Data Spaces. International Data Spaces. Available online: https://internationaldataspaces.org/ (accessed on 19 December 2023).
92. Minimum Viable Data Space (MVDS). Available online: https://docs.internationaldataspaces.org/knowledge-base/mvds (accessed on 20 December 2023).
93. GitHub. Ziruili-Tu-Ilmenau/CMBSE: Repository for the Model Used in the Paper "Collaborative Model-Based Systems Engineering Using Dataspaces and SysML v2". Available online: https://github.com/ziruili-tu-ilmenau/CMBSE (accessed on 20 December 2023).
94. *International Recommendation OIML R60 for Load Cells*; Bureau International de Métrologie Légale: Paris, France, 2000.