

Article

Recommendations for the Model-Based Systems Engineering Modeling Process Based on the SysML Model and Domain Knowledge

Jia Zhang and Shuqun Yang *

College of Electronic and Electrical Engineering, Shanghai University of Engineering Science, Shanghai 201620, China; z_1833476@163.com

* Correspondence: shqyang@sues.edu.cn

Abstract: Model-based systems engineering (MBSE) is a modeling approach used in industry to support the formalization, analysis, design, checking and verification of systems. In MBSE modeling, domain knowledge is the basis of the modeling. However, modeling does not happen overnight; it requires systematic training and a significant investment of resources. Unfortunately, many domain experts lack the expertise required for modeling, even though they know the domain well. The question arises about how to provide system modelers with domain knowledge at the right time to support the efficient completion of modeling. Since MBSE research that integrates AI is just beginning to take off, no public dataset is available. In this paper, aerospace SysML models are constructed based on spacecraft-related domain knowledge to form SysML model data. The validation rules are studied to validate the SysML model data, and combined with the concept of the recommended system, a recommendation method for the MBSE modeling process based on the knowledge and SysML model is proposed. A GLOVE language model is pre-trained by using domain knowledge and general knowledge; the model data are also used to fine-tune the GLOVE language model combined with the pre-training to recommend some domain development processes. The recommendation list is manually quality-verified and fed into the pre-training phase, while new requirement texts are continuously added in the fine-tuning phase, resulting in a more relevant and accurate recommendation list. Experiments show that the incremental recommender system can not only effectively recommend SysML models, but also improve the quality and efficiency of MBSE development.

Keywords: domain modeling; MBSE; NLP; recommendation

Citation: Zhang, J.; Yang, S. Recommendations for the Model-Based Systems Engineering Modeling Process Based on the SysML Model and Domain Knowledge. *Appl. Sci.* **2024**, *14*, 4010. <https://doi.org/10.3390/app14104010>

Academic Editor: Jose Machado

Received: 28 March 2024

Revised: 29 April 2024

Accepted: 5 May 2024

Published: 8 May 2024



Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

The evolution of complex systems in technology and ecology is undergoing profound changes due to the overturning of technological waves and the reshaping of industrial structure [1]. The complexity of these complex engineered systems, as well as the high demands on their performance, are constantly climbing. In recent decades, driven by innovative products in various industries, system development has encountered multiple challenges such as extreme system complexity, short lifecycles and long development cycles [2]. The traditional development model, which overly relies on documentation as the main channel of information transfer and physical testing as the main means of validation, has inherent risks and shortcomings that are gradually surfacing. Guided by the urgent needs of industrial research and the theory of systems engineering [3], the exploration and implementation of model-based systems engineering (MBSE) [4] in the conceptual design phase of a project gradually took off. Nowadays, the specification and integration of digital technologies for MBSE have gained widespread attention and application in the whole-life-cycle R&D process [5,6], thus promoting the realization of innovative development paths,

such as document-centered modeling methodology, whole life-cycle extended application and comprehensive simulation across disciplines.

MBSE modeling is an approach that emphasizes the use of models to describe, analyze, design and verify a system by incorporating domain-specific expertise during the system development process. The MBSE methodology [7–10] provides a systematic set of tools and techniques to enable the process from requirement capture to system design, and verification throughout the development process. The modeling team can translate domain knowledge into executable models to identify potential problems and risks early on and make timely adjustments and optimizations.

MBSE is a widely used systems engineering approach. However, it still has many disadvantages and significant challenges that are difficult to overcome [11,12]. Many enterprises still adhere to traditional systems engineering design and development methods. The adoption of MBSE in the enterprise is also limited by the availability of skilled practitioners [13–15]. Additionally, their informationization ability is insufficient, leading to the formation of a ‘model silo’. The integration of artificial intelligence (AI) with systems engineering is being considered as a means to enhance the level of intelligence in design, optimize decision making, and improve system efficiency and reliability in the face of complexity and uncertainty [16].

In 2019, the Research Committee of the Systems Engineering Research Center (SERC) at the University Affiliated Research Center (UARC), funded by the U.S. Department of Defense, designed a blueprint to guide and lead the direction of AI and autonomy research. The blueprint emphasizes the need to drive novel systems engineering practices to support the development of a new generation of automated, adaptive, and learning systems, collectively referred to as the ‘SE of AI’. In the same year, at the Future of Systems Engineering (FuSE) symposium organized by the International Council on Systems Engineering (INCOSE), Miller et al. [17] introduced the concept of ‘AI for SE’. This two-way shift was characterized as ‘AI for SE’ and ‘SE for AI’.

The SERC Research Committee has created a blueprint to guide research on AI and autonomy. The purpose of this blueprint is to deepen our understanding and focus on the development process. Its aim is to establish connections between the systems engineering discipline and various trends in the field of AI, and to explore the application of these trends to systems automation. The blueprint was elaborated by McDermott, T. et al. [18] and has been presented and discussed in various forums.

Digital engineering forms the basis for all three cross-cutting tasks of SERC and the resulting research roadmap. Systems engineering is in the process of switching between collaboration based on the use of data (authoritative sources of facts) and the use of models (collaborative integrated modeling environments) [19]. As shown in Figure 1, the digital engineering research roadmap developed by the SERC consists of five main goals: (1) model use for decision making; (2) authoritative source of truth (AST); (3) technological innovation; (4) collaborative environment; and (5) digital engineering workforce and cultural evolution. The development of digital engineering begins with data integration in AST, followed by semantic model integration, with the expectation of enabling augmented engineering to bring automation to engineering processes, system quality, and certification through the use of models and big data. Augmented engineering is at the core of AI4SE. The yellow circles in the figure below indicate ongoing development work and the blue circles indicate work in transition areas. Utilizing augmented intelligence in systems engineering and engineering systems is known as augmentation engineering [20].

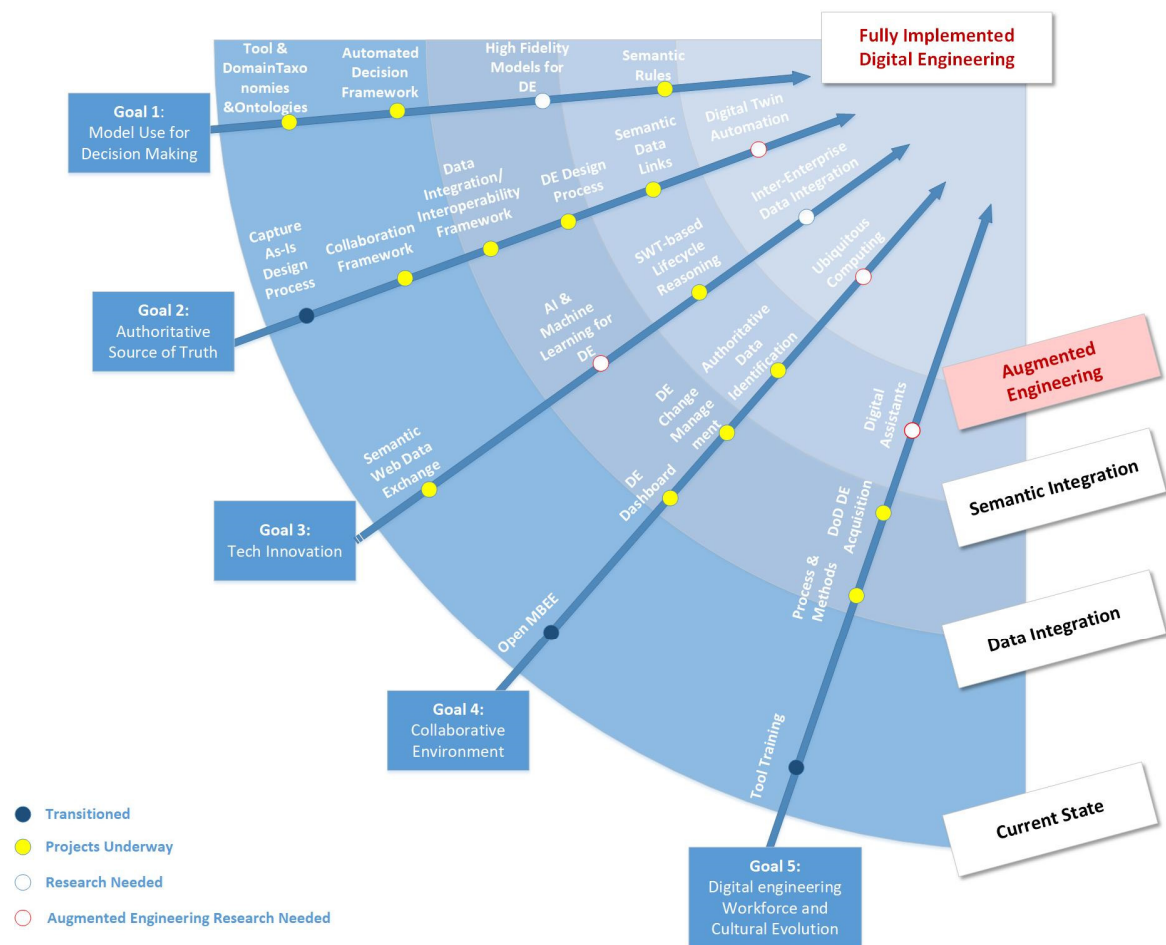


Figure 1. SERC digital engineering roadmap [19].

In this context, McDermott et al. [16] defined AI4SE and SE4AI. AI4SE can be understood as the application of enhanced artificial intelligence and machine learning techniques to systems engineering practices aimed at large-scale model building and spatial design efforts. SE4AI, on the other hand, refers to the application of systems engineering methodologies to the design and operation of learning-based systems, and its core research focus includes the development of learning-based system design principles, lifecycle evolution models, and model management strategies.

In addition, Hagedorn et al. [21] have outlined the current areas of research and development in the engineering disciplines that are evolving in the course of digital transformation. McDermott [19], on the other hand, further summarizes the developments in the field of digital engineering that are closely related to AI4SE, including the following:

1. The use of classification combined with AI techniques for the underlying data. Interoperability between engineering data and program data based on domain-specific ontologies.
2. Authoritative and authentic data sources as the basis for techniques of program management activities in different engineering and complex engineering systems. In order to share data between programs and engineering designs, companies build large datasets to facilitate knowledge transfer and reuse between different programs and projects.
3. Based on knowledge representations such as ontologies, semantic rules provide a fundamental guarantee of integrity and consistency for AI-based reasoning engineering models.
4. Based on the real system model to realize automated digital twin, construct a fully dynamic copy of the virtual system and run in parallel with the physical system.

System design and construction data will be updated from a unified data source corresponding to the real one.

In MBSE modeling, domain knowledge serves as the foundation. This knowledge includes an understanding of the concepts, principles, rules, constraints and best practices of the domain where the system is located. Acquiring domain knowledge involves working with experts, reviewing literature and conducting field research. This deeper understanding helps modelers comprehend the system's nature and requirements, leading to more accurate and credible models. However, modeling requires systematic training and a significant investment of resources. Unfortunately, many domain experts lack the expertise required for modeling, even though they possess a deep understanding of their domain [22]. Providing system modelers with timely domain knowledge to support efficient modeling completion becomes a challenge.

Aiming at the above problems, this paper constructs a SysML model dataset, combines natural language processing (NLP) techniques and proposes a recommendation method for the MBSE modeling process based on domain knowledge and SysML historical model, which recommends the domain expertise for system modelers and assists in modeling to improve the modeling efficiency, accuracy and reliability.

The sections of this paper are organized as follows. Section 1 introduces the concept of MBSE and the background of its development, illustrating the inevitable trend of the development of the combination of systems engineering and artificial intelligence. Section 2 describes the current state of research on assisted modeling in the context of AI4SE and introduces the methodology of this paper. In order to better train the data, Section 3 proposes a dataset construction technique, taking the aerospace domain as an example, constructing an aerospace SysML model, studying the validation rules of data validity, validating the SysML model data and forming a SysML model dataset. Section 4 formally proposes a recommendation method for the MBSE modeling process based on the domain knowledge and SysML historical model, describes the various steps of the method, based on which experiments are carried out in Section 5 to construct the evaluation indexes for the new recommendation, and briefly analyzes the experimental results. Section 6 summarizes the paper's methodology and proposes future directions for the methodology.

2. Related Work

In the context of MBSE, NLP plays an important role in generating SysML models from natural language strategies. Mohammad et al. [23] train named entity recognition models based on multiple textual utterances to identify actors and use cases, which helps to accelerate the early MBSE adoption phase such as the creation of a model repository. Chen et al. [24] proposed an NLP-based framework for information extraction under general conditions that automatically detects actors and their associated behaviors. Furthermore, *aeroBERT-NER* was developed specifically for named entity recognition in the aerospace industry, emphasizing the need for specialized NER models in different domains [25]. Furthermore, *aeroBERT-Classifier* was introduced to classify aerospace requirements using NLP and language models (LMs) in the context of MBSE [26].

In order to enhance the interaction between the modeler and the system for more accurate recommendations, there is an increasing amount of corresponding research. Sun-ke, S. et al. [27] proposed to utilize a semi-supervised system to interact with domain experts to identify rules and to inform the active learner about feature representations based on domain-specific entities and relationships in order to efficiently construct a domain model of the rules. Elkamel et al. [28] proposed to recommend UML classes in the design phase of UML class diagrams to recommend UML classes. Cerqueira et al. [29] proposed a content-based recommendation system for sequence diagrams. Agt-Rickauer et al. [30] introduced a recommendation system based on mediated information retrieval of vocabulary information integrated with a number of existing knowledge bases to automatically suggest model elements for a domain model—DoMoRe. Afterward, Stephan et al. [31] describe the vision of implementing a cognitive virtual software modeling as-

sistant that uses synergies between machine learning and software cloning to improve software design and MDE. Segura et al. [32] present an extensible meta-modeling assistant that allows for providing heterogeneous data description sources and unifying queries that are graphically visualized and integrated into the (meta-)model being constructed. Pérez-Soler et al. [33,34] developed a modeling robot using NLP techniques to interpret user inputs in natural language in order to construct meta-models step-by-step. In addition, Mussbacher et al. [35] proposed a conceptual framework to build intelligent modeling assistants in a more systematic way. Saini et al. [36] introduced DoMoBOT, a web-based BOT for domain modeling, and demonstrated how DoMoBOT's architecture and implementation facilitate robot modeler interaction. Saini et al. [37] later extended the previously proposed architecture by adding a recommendation component and a query answering component to the previously proposed architecture. Burgueño, L. et al [38] in order to facilitate the definition and improve the quality of the domain model, an NLP based approach will provide automatic completion suggestions for the part of the model that is being constructed based on the automated analysis of the project's contextual knowledge and generic knowledge. The process will also take into account the feedback gathered by the designers when interacting with the assistant.

Although different modeling techniques, modeling theories and modeling tools are used in various industries, most system designers indicated that lack of understanding of the value of MBSE and lack of relevant skills and competencies are the two main challenges hindering the transformation of MBSE.

In the context of digital transformation and considering the trend of increasing integration of artificial intelligence and systems engineering, this paper focuses on the research direction of AI4SE. By deeply applying AI techniques, we aim to significantly improve the modeling efficiency of system modelers.

In this paper, inspired by the approach of Burgueño et al. [38], we propose a recommendation method that fuses domain knowledge and SysML models. The domain model is described by constructing a collection of SysML models and a pre-trained model based on general knowledge is fine-tuned using domain-specific text and systems engineering domain model data, aiming at recommending appropriate classes and attributes for a specific development domain. The recommended generated lists are subject to manual quality review and the feedback results are used for optimization in the pre-training phase. On the other hand, during the fine-tuning process, new requirement texts can be continuously introduced to improve the relevance and accuracy of the recommended lists.

3. Dataset Construction

The field of aerospace research has strict requirements for confidentiality because it involves national security, high technology and major economic interests. The information generated during the design, manufacture and operation of spacecraft, including technical details, performance parameters, material science, propulsion systems, etc., is often highly sensitive.

Due to these special characteristics of the spaceflight domain, it is almost impossible to find a complete and detailed set of spacecraft-related SysML models in the open channel. Most detailed system models and associated documentation are protected by strict access controls and confidentiality agreements. These measures ensure that critical information is available only to authorized personnel to maintain the security and integrity of the project. Therefore, when conducting research related to spacecraft systems engineering, this paper chooses to work on the basis of constructing a dataset. There are two sources of datasets: one is a manually constructed set of spaceflight-related SysML models based on corporate internship experience, and the other is publicly available domain textual information.

Before model training, dataset construction and data processing are required to ensure the performance and accuracy of the model. The research of dataset construction mainly includes the following parts: construction of the SysML model, SysML model format transformation as well as parsing, construction of training dataset and construction of

candidate word set. Through the construction of datasets, the data quality can be effectively improved, the model learning process can be optimized and the generalization ability of the model can be ultimately enhanced. The construction of the dataset allows for further validation of its effectiveness, thereby supporting subsequent research and analysis in an effective manner.

It should be noted that the purpose of this section is not to construct the complete aerospace dataset, but to prepare for and exemplify the training of the newly proposed method, and is not limited to a specific domain.

3.1. Constructing the SysML Model

Using the MagicDraw tool, RFLP (requirement, functional, logical and physical) model [39] is applied to perform rough SysML manual modeling of a generic spacecraft system based on the SysML language. The study focuses on domain modeling; therefore, the model content is based on structural models.

As shown in Figure 2, the model references spacecraft system design and uses the block definition diagram (BDD) as a bearer for mission environment analysis, with each block describing one environmental influencing factor. Mission environment analysis is a comprehensive study of the environment in which a particular mission or project is located. It focuses on external and internal factors that may have a direct impact on the execution of the task. The primary purpose of mission environment analysis is to identify, assess and manage the risks and opportunities that may affect the successful completion of the mission. For the mission environment of a spacecraft, the main influencing factors are the ground, the spacecraft itself, space, launchers, users, ground stations and GPS. The ground influences include the natural environment, and the space influences include electromagnetic waves, satellites, the sun and other planets. The * in the diagram indicates that there can be more than one, e.g. the Mission Environment can contain one or more users. The word next to the relationship link is the name of the instance, which can be modified according to the actual situation, and the diagram uses the same name as block by default.

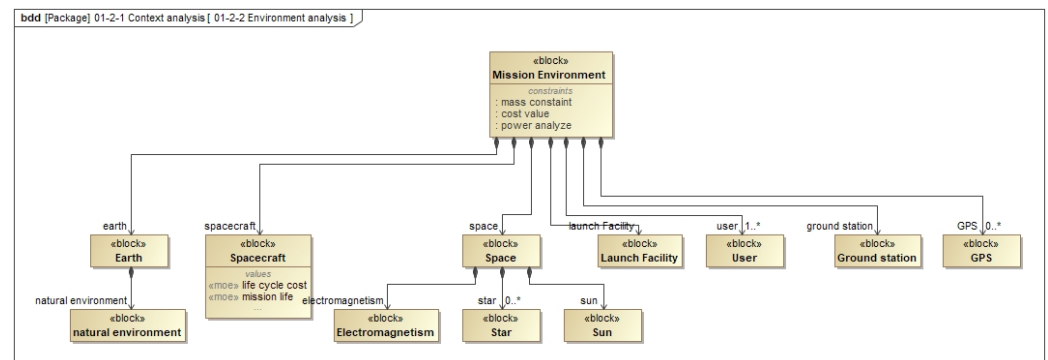


Figure 2. SysML model example.

When analyzing the overall mission environment, in addition to the internal and external environment analysis, it is also necessary to identify the constraints that may be encountered during mission execution (e.g., time, budget, resource limitations, etc.) as well as the assumptions and prerequisites that may affect mission execution. As an example, the representative constraints of the spacecraft mission environment are mass constraints, cost constraints and energy analysis constraints.

The rest of the model instances are represented in the block representation of entities or components in the system. The block can represent physical entities (e.g., hardware devices, sensors, etc.) or logical entities (e.g., software modules, algorithms, etc.). Each block has its own attributes, operations and behaviors, as well as relationships with other blocks. The relationships between the blocks

can be represented by different lines and arrows, such as associations, generalizations and dependencies. The simple semantics based on BDD facilitate the subsequent extraction of entities, attributes and relationships to support the subsequent work of analyzing the data.

3.2. SysML Model Format Transformation

In this paper, the MagicDraw modeling tool is used to construct SysML model sets for the spacecraft-related domains. Since the graphical content of the models is difficult to process directly for data processing, the graphical models are transformed into XML format by the MBSE modeling tool. The advantages of XML data are in terms of its readability, scalability, cross-platform, self-descriptiveness, support for multiple data formats and simplification of data processing and improvement of data availability. In XML documents, the data structure is clear. Moreover, XML documents are text-based, so XML data can be easily exchanged and shared among different devices and systems. This cross-platform nature greatly simplifies the complexity of data exchange and processing. These advantages make XML ideal for handling scenarios such as network data transfer, configuration files and storing structured data.

3.3. SysML Model Data Parsing

When parsing XML model data, namespaces are used to distinguish between elements and attributes from different sources or contexts to avoid naming conflicts. namespace_map is a dictionary that maps short namespace prefixes (e.g., 'uml' and 'xmi') to full namespace URIs to accurately identify namespace elements in XML during parsing. Then, using the namespace mappings, the XML data are loaded and parsed by the XML parsing library and converted into an actionable tree structure. During the parsing process, XPath expressions and namespace_map are used to pinpoint and access specific namespace elements to read or modify their attributes, content and sub-elements. This process ensures an accurate and efficient parsing of XML data. The concept of the tree is used in the process of parsing XML data, through which NLP tools and data preprocessing techniques can be better utilized to support further analysis and application of textual data.

Table 1 shows the structure of some of the XML data stored after parsing. Each node can be constructed as a dictionary containing information about the name, parent, child and arguments, and is stored in a file.

Table 1. XML data parsing storage example.

Name	Parent	Child	Argument
'Mission Environment'	None	'Earth', 'Space', 'GPS', 'Ground Station', 'User', 'Launch Facility', 'Spacecraft'	
'Earth'	'Mission Environment'	'Natural Environment'	
'Spacecraft'	'Mission Environment'		'lifecycle cost', 'mission life', 'performance parameters'

3.4. Constructing the Training Dataset

Building a dataset is a crucial step in machine learning and deep learning projects. Building a dataset in this study consists of two main steps: data collection and data cleaning. Collecting data is the first step in building a dataset. The data can come from a variety of sources such as public datasets, website crawls, sensor data, API calls, and so on. Cleaning the data is to remove noise, handle missing values and outliers, and ensure data quality.

This includes operations such as removing duplicates, dealing with missing values and standardizing the data format.

The aerospace domain was chosen for modeling in this study. Due to the sensitive and proprietary nature of aerospace technology, real aerospace datasets are often difficult to obtain or subject to strict access restrictions. The National Aeronautics and Space Administration (NASA), the European Space Agency (ESA) and some spacecraft manufacturers or suppliers may release some spacecraft-related open-source datasets for research, education and innovation. Initial construction of datasets is performed through research surveys that search for publicly available spacecraft-related domain texts.

In this paper, the domain text is analyzed using the ChatGPT tool and the results are extracted to generate training data for the NLP model, cleaned of punctuation and plain numbers, and deactivated. At the same time, the XML data are parsed to be used to fine-tune the NLP model again and supplement the dataset. More specifically, the concept of tree is used in the process of parsing the XML data, where the parent nodes and child nodes are combined to save the data, which is supplemented by the contextual data of the nodes. Through this approach, NLP tools and data preprocessing techniques can be better utilized to provide support for further analysis and application of text data.

3.5. Construct Candidate Word Sets

Obtaining a set of candidate words is a critical step in NLP, especially in tasks such as keyword extraction and text categorization. Rule-based approaches usually utilize linguistic knowledge to define a set of rules to extract candidate words. For example, the results of lexical annotation can be utilized to take some specific lexical words (e.g., nouns, verbs) as candidate words. However, rule-based methods cover all cases and require manual definition and maintenance of rules. Statistical-based methods usually utilize a large amount of text data to count the frequency of words, co-occurrence rate and other indicators, and take high-frequency words or words with high co-occurrence rate as candidate words. Machine learning-based methods usually treat candidate word extraction as a classification problem and utilize a large amount of labeled data to improve the accuracy of the extraction results. However, both methods require a certain amount of data.

In this study, due to the lack of publicly available datasets, we chose to utilize the ChatGPT tool to analyze domain-specific text and obtain a set of candidate words. Aerospace-specific vocabulary usually consists of single-word terms, multi-word phrases and acronyms, with multi-word phrases usually having a word count of two to four [40]. When processing proper nouns, we imposed a maximum length limit of five characters and performed word segmentation. After initially obtaining the candidate word set, we filtered the words based on their part of speech, removing adjectives, adverbs, prepositions, and so on. On the other hand, we supplemented the candidate word set with information from various nodes and attributes obtained from the model's XML data to enrich the corpus content.

3.6. Validation of Dataset Validity

When manually constructing a dataset, verifying the integrity and consistency of the data is an essential step to ensure that no critical information is missing from the dataset. At the same time, it is ensured that the data remain logically consistent among the various parts, thus providing a solid foundation for subsequent data processing and analysis, and effectively preventing the distortion of results due to missing or self-contradictory data.

This results in the need for data integrity and consistency validation [41]. Data integrity validation is the process of checking whether the data are comprehensive and free of omissions, whether the data are unique or redundant, and ensuring that the data have not been tampered with or corrupted during transmission or storage. Data integrity validation helps identify and correct incomplete or redundant parts of the data in a timely manner, thus ensuring that data-based decision making and analysis are reliable and effective. Data

consistency validation is the process of primarily checking for discrepancies or conflicts between data from different data sources or different points in time and ensuring that they remain consistent across systems.

As described in Section 4.1, for text-based contextual datasets and candidate word sets, the steps of preprocessing and cleaning the data have already been experienced during their construction. Therefore, the focus of this section is on validating the SysML dataset.

In order to ensure that the constructed model accurately reflects the actual requirements and design of the system, and to prevent model bias due to missing or incorrect data, the integrity and accuracy of the data are verified after manually constructing the SysML model dataset, which in turn ensures the validity and reliability of the analysis and decision making based on this dataset.

Since the models are completed in different project contexts, Project1 and Project2 each define the model elements therein, the following possible contradictions need to be considered when merging the model data:

1. Whether there is a concept of duplication of model elements across projects;
2. Whether the model elements across projects have different definitions for the same concepts;
3. Whether the model elements across projects have a circular inheritance relationship;
4. Whether model elements across projects have the same combination of relationships and pairs of relationships.

Based on the result of parsing the XML file, data validation is performed based on each node and its contextual information. For the above problems, different methods are used in turn to solve them:

1. Iterate over the node names in all projects and query whether there is any duplication;
2. Iterate over the node names in all projects, and if there are nodes with the same name, compare their contextual information, including the father node, child nodes and attribute values;
3. Traverses all parent–child pairs in the project, performs depth-first search based on the concept of stack [42], and detects circular dependencies by maintaining access lists and path lists. The access list is used to record the nodes that have been checked and have no problems, while the path list is used to record the nodes in the current search path so that the specific path can be pointed out when a circular dependency is detected;
4. Iterate through all the parent–child pairs in the project and query for the existence of duplicate parent–child pairs.

By using the above method, the model data were examined and it was found that there were classes with the same name in both projects, but not classes with the same context. This situation occurs because the class was designed in a different context, so the design perspectives are different. Iterating through all the parent–child pairs in the projects, no circular dependency issues were found, but duplicate parent–child pairs were found. This is because even in the same project, there may be data duplication with different perspectives on system design and different levels of analysis.

For contextual datasets of models and texts, there may be duplication of data pairs. When the two are merged, the above problem can be avoided by removing the duplicates from the updates.

4. Methodology

The technical route of the text-based single recommendation method is shown in Figure 3. For purposes of my research, this study opts to manually construct a SysML model as the historical model dataset. Based on the constructed SysML model instances, the MBSE modeling tool is used to export the SysML model into an XML format. By reading the personalized XML tag content, critical information such as classes and attributes can be efficiently extracted from the model.

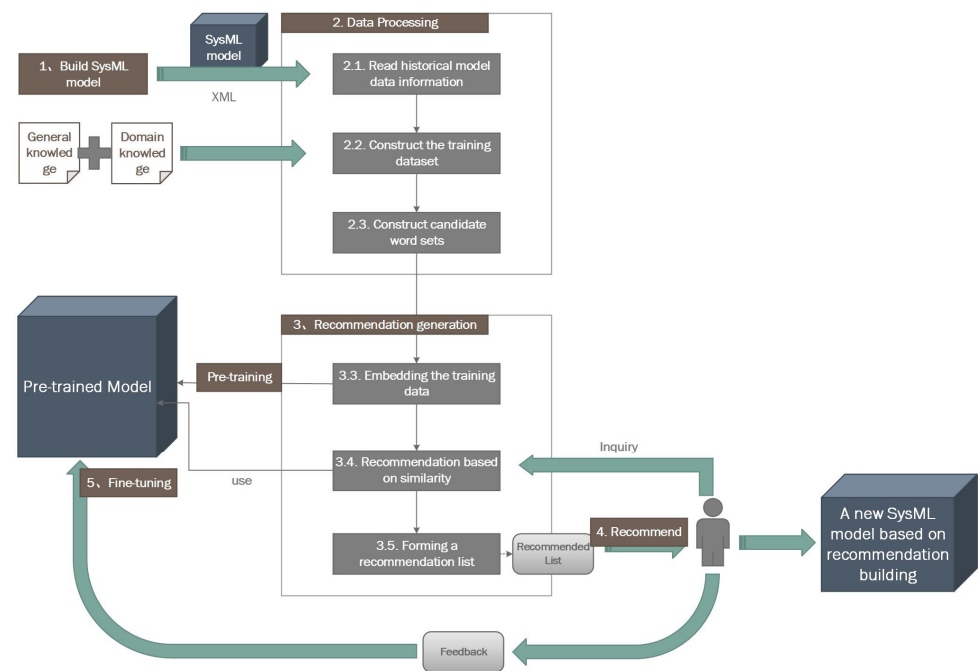


Figure 3. A technology roadmap for text and model-based one-time recommendation methods.

The training dataset is constructed according to Section 3, based on which the GLOVE model, which has already been trained based on general knowledge, is pretrained using information from two sources so that the model learns rich domain knowledge. By adding domain knowledge GLOVE models can be better adapted to domain-specific data and tasks. Domain knowledge helps the model understand the terminology and concepts within the domain and thus improves the performance within that domain. When the model is queried, a recommendation list is generated based on the similarity calculation. Based on the content of the recommendation list, the user makes a selection. We use user feedback as a valuable data source to fine-tune and optimize the model so as to continuously improve the performance of the recommendation system.

4.1. Model Pre-Training

Pre-training of NLP models [43] is a common technique in the field of deep learning, especially for NLP tasks. In model pre-training, a neural network model is usually trained using large-scale unlabeled textual data to learn a generalized language representation or schema. Such pre-trained models are usually fine-tuned on specific tasks to fit specific application scenarios.

Common pre-training methods based on language models include Word2Vec [44], GLOVE [45] and FastText. These methods capture the semantic relationships between words by training a large corpus and representing each word as a vector. However, these methods cannot effectively solve the problems of multiple meanings of words and missing contextual information.

To solve the above problems, researchers have proposed more complex pre-trained language models such as ELMo, GPT and BERT. These models use a deep bidirectional neural network structure to better capture contextual information. Among them, BERT [46] uses a bi-directional transformer structure and is pre-trained by the masked language model (MLM) and next-sentence prediction (NSP) tasks, which has achieved significant results in various NLP tasks.

Given that NLP models trained on large general-purpose text corpora such as Twitter, Wikipedia, and Google News have been shown to have excellent generalization ability and utility, this paper fine-tunes (fine-tuning) the GLOVE model, which has already been trained based on general knowledge, with the input of new domain knowledge text.

The advantage of model pre-training is that large-scale unlabeled data can be utilized for learning, thus improving the model's performance and generalization ability on specific tasks. Pre-trained models can be migrated and fine-tuned to various NLP tasks, such as text categorization [47,48], named entity recognition [49,50], syntactic analysis [51], etc., thus speeding up model training and improving results.

4.2. Recommendation Generation

Recommendation generation typically involves one or more recommendation algorithms that predict what a user might be interested in based on the user's historical behavior, preferences, and other relevant information.

4.2.1. Embedding the Training Data

Generating word vectors is an important step in NLP, which involves converting words or phrases in a text into a numerical vector representation so that they can be processed and understood by machine learning models. Word vectors typically capture semantic and syntactic information about words, allowing semantically similar words to be located closer together in the vector space.

4.2.2. Recommendation Based on Similarity

Similarity-based recommendation is a popular method for providing personalized recommendations by analyzing the similarity between users or items. The core idea of this approach is that if a user likes an item, they may also like other similar items.

The computation of similarity is the key step in similarity-based recommendation. The features of a user or item are represented as vectors and their similarity is calculated using methods such as cosine similarity, Pearson's correlation coefficient, or Euclidean distance. The selection of the method depends on the specific application scenario and data characteristics.

This paper calculates cosine similarity between entities based on their attributes or features and recommends entities with high similarity to users. Cosine similarity is a common recommendation algorithm used to measure the angle between two vectors to determine their similarity. The value of cosine similarity ranges from -1 to 1 , with a higher value indicating greater similarity. The list of recommendations is sorted according to similarity.

4.2.3. Forming a Recommendation List

The list of generated suggestions is presented to the designer for reference and decision making. For each recommendation, the designer has the option to adopt, reject, or put it on hold. Once the system modeler decides to adopt or reject a suggestion, we categorize it as a positive or negative test case, respectively. However, if the designer chooses to ignore a recommendation, no action will be taken, and the ignored recommendation may be re-presented to the designer for consideration in the future.

4.3. User Feedback

Feedback mechanism is a concept that is widely used in several fields. Feedback can be categorized into two types: positive feedback and negative feedback. Positive feedback refers to the controlled part of the feedback information, the direction of the control information with the same, can promote or strengthen the activities of the control part of the system tends to be unstable, which may lead to the system more and more deviation from the equilibrium position. Negative feedback, on the other hand, means that the controlled part sends out feedback information whose direction is opposite to that of the control information, which can inhibit or weaken the activity of the controlling part and help the system to maintain a steady state.

In this paper, the operation of the console is used to simulate the click confirmation of the user interface. The user is prompted at the console to enter relevant data as input to

the recommendation algorithm, and the recommendation system returns a set of recommended items based on the user's input, after processing. The console prints out all the recommendations and their related information for the user to view. Based on the obtained recommendations, we can manually determine one or more positive samples in the console and record the results. The program log identifies the recommendations for each "click" and labels the candidates as positive feedback, which helps the recommender system to more accurately adjust the user profile and recommendation algorithms to provide recommendations that better meet the user's needs.

In this step, the system designer's acceptance and approval of a recommendation output, i.e., "click", are regarded as positive feedback that is sent back to the pre-trained model to improve the quality and performance of the model. The designer's feedback not only helps the model to optimize the existing concepts, but also encourages the model to be closer to the actual needs in future recommendations. At the same time, whenever a designer chooses to abandon a recommendation, i.e., does not "click" on it, we record it as a negative example. These negative examples are used as negative feedback to fine-tune the pre-trained model, so that the pre-trained model can avoid similar inaccuracies in the subsequent recommendations, and guide the model towards a more accurate and desirable direction. This combination of positive and negative feedback allows the recommendation results to be continuously optimized, gradually approaching the ideal state.

4.4. Building SysML Models Based on Recommendations

Through text-based incremental recommendation techniques, entity-based inputs can generate a list of recommendations. Then, based on the project background and personal experience, the system designer can select these recommendations to ensure that each of them fits the requirements. Next, the system designer analyzes the entities, determines the relationship between different entities, and selects appropriate model elements to serve as the bearers of these recommendations.

For word A, based on the recommendation list, concept B that meets the individual's needs is selected. The criteria for constructing a model based on recommendations in this paper are as follows:

1. If concept B is a class, construct block based on SysML. On this basis, judge the relationship between the new concept B and A. If it is clear that the two are composition or inheritance, add the corresponding model elements. If it is not possible to judge for the time being, then choose the "association" relationship connection.
2. If concept B is a class and already exists, it proves that in the previous recommendation, class B and another class C have a very close relationship. At this point, you can re-judge the relationship between class B, class A and class C, and consider whether to change.
3. If concept B is an attribute, determine whether it is an attribute of A; if so, add attributes for A.
4. If concept B is an attribute, class A inherits from class C. If class C also has the attribute, drop the addition of attribute B.

5. Discussion

Regarding the specific implementation of the recommendation process, this study utilized the GLOVE implementation written in Python by Stanford. The generality of the approach is more clearly demonstrated by the fact that the following three domain texts were randomly selected to be obtained as training data by searching for spacecraft-related domains. The domain knowledge documents used here are "Spacecraft dynamics and control: an introduction" [52], "Spacecraft thermal control" [53] and "Standard materials and processes requirements for spacecraft" [54]. These domain knowledge documents are readily accessible on Google and the data are publicly available.

The parameter settings for the GLOVE model are shown in Table 2 below.

Table 2. GLOVE model parameterization.

Initialization Parameter	Value
no_components	100
learning_rate	0.05
Window_size	10

no_components represents the number of latent dimensions used to express the underlying features of the data, and learning_rate is the learning rate for stochastic gradient descent (SGD) estimation, controlling the update speed of model parameters during training. The remaining parameters are set to the default values of the GLOVE model. Because the task complexity in this study is moderate, 100-dimensional word vectors are sufficient to capture the key semantic relationships between words without the need for higher dimensions. Moreover, the amount of training data is small and using higher word vector dimensions may lead to overfitting.

After inputting a word or phrase from the console, a corresponding recommendation list is obtained, and the candidate items are sorted and displayed according to the similarity of the word vectors, and the similarity of the corresponding candidate words is output. From this, we can judge whether it belongs to an entity or an attribute based on the recommended concepts, and select entities and attributes in turn according to the criteria of constructive models so as to update the model content and construct a new SysML model to assist modelers in modeling.

We set the number of recommended items to 30, with the first 10 being the most similar phrases. Based on the fine-tuned GLOVE model word embeddings, the input “spacecraft” generates the following recommended output list, as shown in Table 3.

Table 3. 15 Recommendations based on “spacecraft”.

No.	Candidate Term	Similarity
1	propulsion subsystem	0.700057
2	power subsystem	0.649774
3	power mgmt sw	0.388500
4	tank	0.385118
5	avionics subsystem	0.377041
6	mission context	0.318995
7	launch vehicles	0.287502
8	software subsystem	0.277133
9	brazed layer	0.269542
10	flight crew	0.231632
11	stellar	0.229464
12	space physics	0.229332
13	heat exchange	0.220312
14	combustion gases	0.219325
15	gn&c subsystem	0.215557

From this, we can select entities and attributes, and based on the selection, obtain the recommended relationships between entities. As shown in Figure 4a,b, spacecraft was taken as an example and selected based on the recommended list.

Based on this approach, a new SysML model is constructed sequentially based on the contents of the recommendation list.

For example, Figure 5 shows the model constructed based on recommendations, combined with personnel selection. The selection of this model element is based on personal judgment and actually simulates the process of selecting based on recommendations, thus constructing a new SysML model. Subsequently, the SysML model constructed based on recommendations can be used as historical model data to enrich the training dataset, making recommendations more accurate.

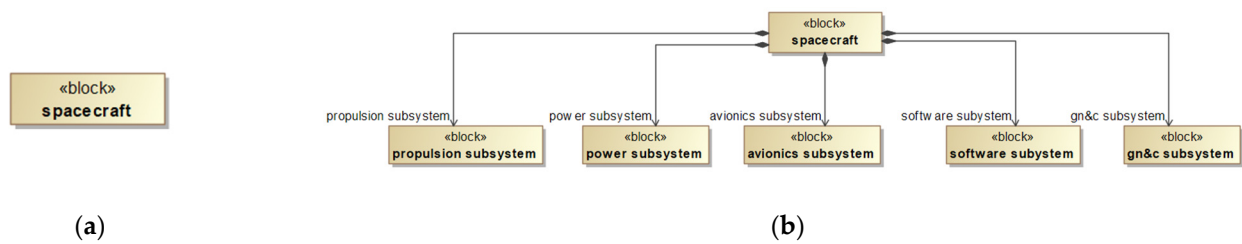


Figure 4. Testing with spacecraft as an example. (a) Creating the spacecraft block. (b) On this basis, the recommended items were selected according to Table 2 and added to the model.

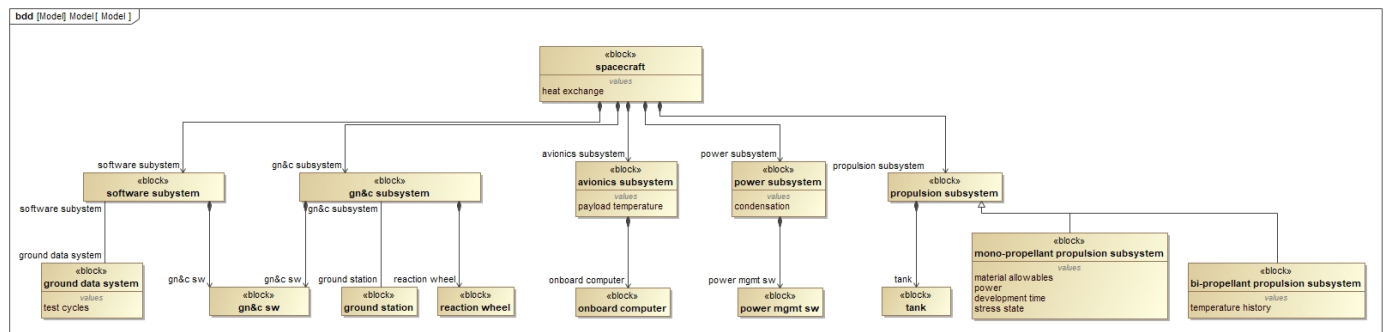


Figure 5. SysML model constructed based on recommendations.

Take the list of candidate words as a test set and recommend them separately to get the candidate list of the top 30 items of similarity of each keyword. Use the model as a source to construct the true set of recommendation data. For one of the classes block1, the classes associated with it and its own attributes are used as its recommendation items. More specifically, using class block1 as a node, select the two levels of abstraction upward and the two levels of detail downward, and include the “parent”, “child” directly associated with block1, and the indirectly associated “sibling” classes, as well as “parent” of “parent” classes, and “child” of “child” classes as the true set of recommended items. As an example, the recommendations for spacecraft include all attributes, the subsystem classes and all classes contained in the subsystem classes.

Based on this, 100 inputs are randomly tested to judge the number of times out of 100 recommendations are valid. The validity is judged by evaluating a recommendation as valid if there is a truth value in the result based on one of the inputs. This section uses the GLOVE pre-trained model and evaluates the effectiveness of recommendations by comparing the pre-feedback and post-feedback results, respectively, using the user’s initial feedback as a cutoff. A comparison was attempted using different pre-training models Word2Vec and bert. The results of the three pre-trained models before and after feedback are evaluated as shown in Table 4 below. Recommendations were made using Minkowski similarity as well as cosine similarity respectively, and it can be seen that the effectiveness of recommendations using cosine similarity is higher. And after one feedback, the effectiveness of recommendation is improved accordingly.

Table 4. Evaluation of the results.

Similarity Metric	Recommendation Effectiveness (before Feedback)	Recommendation Effectiveness (after Feedback)
Cosine similarity	69.1%	70.2%
Minkowski similarity	65.5%	67.8%

This method integrates two sources of information, domain knowledge and model data, and better utilizes SysML models to make recommendations more authentic and

reliable. The manually constructed models are used in the study as the historical model set for example only. During the actual application of the method, the SysML models constructed based on the guidance of domain experts can be used as the training data source and combined with the specialized domain knowledge as the input to make authentic and reliable recommendations. Subsequently, the SysML models constructed based on the recommendations can be used as historical model data with new and increasing domain knowledge or specifications to continuously fine-tune the GLOVE model, gradually enrich the training dataset, and utilize the database to record the model evolution process in the full lifecycle.

6. Conclusions

In order to better assist modeling and improve modeling efficiency, this paper, inspired by the approach of Burgueño et al. [34], proposes a recommendation method that fuses domain knowledge and SysML models. The method not only considers the influence of general and domain knowledge on the modeling process, but also utilizes the SysML model as training data for recommendation of subsequent modeling. As an example, in the aerospace domain, the manually constructed SysML models are used as a historical model set and their validity is verified to better support the experiments. Using domain-specific text and systems engineering domain model data, the original pre-trained model based on generic text is fine-tuned with the aim of recommending appropriate classes and attributes for a specific development domain. Based on the list generated from the recommendations, the rules for constructing the new SysML model are specified, and from the results, it can be seen that the recommendation list takes both data sources into account in a comprehensive manner, and based on the formulated criteria for judging the validity of the recommendations, the approach proposed in this paper is practical and efficient for SysML modeling.

Looking ahead, there are plans to explore other recommendation methods, such as knowledge graphs or alternative models, for even more precise recommendations. The graphical data structure of knowledge graphs bears similarities to the information structure of SysML models. Leveraging the entities, attributes and relationships from knowledge graphs can enrich descriptions, store more SysML information and ultimately improve recommendation accuracy and user satisfaction.

As engineering projects continue to evolve, there will be an influx of new domain knowledge. A complete recommender system can be designed by adding a front and back end. In the future, the system engineering domain model can be reconstructed based on single-shot recommendation lists. New domain texts can be incorporated for word embedding, and the deep learning model can be further fine-tuned. As the model iterates, texts, recommendations and constructed models from each iteration can be recorded in a database, forming a multi-version evolution lifecycle for models/records. This approach effectively enhances the quality and cost efficiency of model-based systems engineering development.

Author Contributions: Conceptualization, J.Z. and S.Y.; methodology, J.Z. and S.Y.; modeling, J.Z.; software, J.Z.; validation, J.Z.; investigation, J.Z.; writing, J.Z.; tuition, S.Y. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The original contributions presented in the study are included in the article, further inquiries can be directed to the corresponding author.

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

References

1. Törngren, M.; Grogan, P.T. How to deal with the complexity of future cyber-physical systems? *Designs* **2018**, *2*, 40. [CrossRef]
2. Kossiakoff, A.; Biemer, S.M.; Seymour, S.J.; Flanagan, D.A. *Systems Engineering Principles and Practice*; John Wiley & Sons: Hoboken, NJ, USA, 2020.
3. Gregory, J.; Berthoud, L.; Tryfonas, T.; Rossignol, A.; Faure, L. The long and winding road: MBSE adoption for functional avionics of spacecraft. *J. Syst. Softw.* **2020**, *160*, 110453. [CrossRef]
4. Haskins, C.; Forsberg, K.; Krueger, M.; Walden, D.; Hamelin, D. Systems engineering handbook. *INCOSE* **2006**, *9*, 185. Available online: https://en.wikipedia.org/wiki/Model-based_systems_engineering (accessed on 13 July 2023).
5. Gough, K.M.; Phojanamongkolkij, N. Employing Model-Based Systems Engineering (MBSE) on a NASA Aeronautic Research Project: A Case Study. In Proceedings of the 2018 Aviation Technology, Integration, and Operations Conference, Atlanta, GA, USA, 25–29 June 2018; p. 3361. [CrossRef]
6. Lorenzo Alvarez, J.; de Koning, H.P.; Fischer, D.; Wallum, M.; Metselaar, H.; Kretzenbacher, M. Best practices for model based systems engineering in ESA projects. In Proceedings of the 2018 AIAA SPACE and Astronautics Forum and Exposition, Orlando, FL, USA, 17–19 September 2018; p. 5327. [CrossRef]
7. Estefan, J.A. Survey of model-based systems engineering (MBSE) methodologies. *IncoSE MBSE Focus Group* **2007**, *25*, 1–12.
8. Lykins, H.; Friedenthal, S.; Meilich, A. Adapting Uml for an object oriented systems engineering method (Oosem). In Proceedings of the INCOSE International Symposium, Minneapolis, MN, USA, 16–20 July 2000; Volume 10, No. 1. pp. 490–497. [CrossRef]
9. Halle, M.; Vergnaud, J.R. Harmony processes. In *Crossing the Boundaries in Linguistics: Studies Presented to Manfred Bierwisch*; Springer: Dordrecht, The Netherlands, 1981; pp. 1–22. [CrossRef]
10. Kruchten, P. *The Rational Unified Process: An introduction*; Addison-Wesley Professional: Boston, MA, USA, 2004.
11. Albers, A.; Zingel, C. Challenges of model-based systems engineering: A study towards unified term understanding and the state of usage of SysML. In *Proceedings of the Smart Product Engineering: Proceedings of the 23rd CIRP Design Conference, Bochum, Germany, 11–13 March 2013*; pp. 83–92. [CrossRef]
12. Chami, M.; Bruel, J.M. A survey on MBSE adoption challenges. In Proceedings of the INCOSE EMEA Sector Systems Engineering Conference (INCOSE EMEASEC 2018), Berlin, Germany, 5–7 November 2018; pp. 1–16.
13. Broodney, H.; Dotan, D.; Greenberg, L.; Masin, M. 1.6. 2 Generic Approach for Systems Design Optimization in MBSE 1. In *INCOSE International Symposium*; Wiley Online Library: Hoboken, NJ, USA, 2012; Volume 22, pp. 184–200. [CrossRef]
14. Chami, M.; Aleksandraviciene, A.; Morkevicius, A.; Bruel, J.M. Towards solving MBSE adoption challenges: The D3 MBSE adoption toolbox. In Proceedings of the INCOSE International Symposium, Rome, Italy, 9–12 July 2018; Volume 28, No. 1. pp. 1463–1477. [CrossRef]
15. Wang, L.; Izygon, M.; Okon, S.; Wagner, H.; Garner, L. Effort to accelerate MBSE adoption and usage at JSC. In Proceedings of the AIAA SPACE 2016, Long Beach, CA, USA, 13–16 September 2016; p. 5542. [CrossRef]
16. McDermott, T.A.; Blackburn, M.R.; Beling, P.A. Artificial intelligence and future of systems engineering. *Syst. Eng. Artif. Intell.* **2021**, 47–59. [CrossRef] [PubMed]
17. Miller, W. Future of Systems Engineering. In Proceedings of the INCSOE 2019 International Symposium, Torrance, CA, USA, 26–29 January 2019; Volume 28.
18. McDermott, T. Digital engineering and AI—Transformation of systems engineering. *Int. Counc. Syst. Eng.* **2020**, *1*, 78–85.
19. McDermott, T.; DeLaurentis, D.; Beling, P.; Blackburn, M.; Bone, M. AI4SE and SE4AI: A research roadmap. *Insight* **2020**, *23*, 8–14. [CrossRef]
20. Madni, A.M. Exploiting augmented intelligence in systems engineering and engineered systems. *Insight* **2020**, *23*, 31–36. [CrossRef]
21. Hagedorn, T.; Bone, M.; Kruse, B.; Grosse, I.; Blackburn, M. Knowledge representation with ontologies and semantic web technologies to promote augmented and artificial intelligence in systems engineering. *Insight* **2020**, *23*, 15–20. [CrossRef]
22. Doren, A.; Markina-Khusid, A.; Cotter, M.; Dominguez, C. A Practitioner’s Guide to Optimizing the Interactions Between Modelers and Domain Experts. In Proceedings of the 2019 IEEE International Systems Conference (SysCon), Orlando, FL, USA, 8–11 April 2019; pp. 1–8. [CrossRef]
23. Chami, M.; Zoghbi, C.; Bruel, J.M. A first step towards AI for MBSE: Generating a part of SysML models from text using AI. In *INCOSE Artificial Intelligence for Systems Engineering: 2019 Conference Proceedings, Madrid, Spain, 12–13 November 2019*; The REUSE Company: Madrid, Spain, 2019; pp. 123–136.
24. Chen, M.; Bhada, S.V. Converting natural language policy article into MBSE model. In Proceedings of the INCOSE International Symposium, Detroit, MI, USA, 25–30 July 2022; Volume 32, pp. 73–81. [CrossRef]
25. Tikayat Ray, A.; Pinon-Fischer, O.J.; Mavris, D.N.; White, R.T.; Cole, B.F. aeroBERT-NER: Named-entity recognition for aerospace requirements engineering using BERT. In Proceedings of the AIAA SCITECH 2023 Forum, Gaylord National Harbor, Washington, USA, 23–27 January 2023; p. 2583. [CrossRef]
26. Tikayat Ray, A.; Cole, B.F.; Pinon Fischer, O.J.; White, R.T.; Mavris, D.N. aerobert-classifier: Classification of aerospace requirements using bert. *Aerospace* **2023**, *10*, 279. [CrossRef]
27. Sunkle, S.; Kholkar, D.; Kulkarni, V. Informed active learning to aid domain experts in modeling compliance. In Proceedings of the 2016 IEEE 20th International Enterprise Distributed Object Computing Conference (EDOC), Vienna, Austria, 5–9 September 2016; pp. 1–10. [CrossRef]

28. Elkamel, A.; Gzara, M.; Ben-Abdallah, H. An UML class recommender system for software design. In Proceedings of the 2016 IEEE/ACS 13th International Conference of Computer Systems and Applications (AICCSA), Agadir, Morocco, 29 November–2 December 2016; pp. 1–8. [\[CrossRef\]](#)
29. Cerqueira, T.G.; Ramalho, F.; Marinho, L.B. A Content-Based Approach for Recommending UML Sequence Diagrams. In *SEKE*; Federal University of Campina Grande: Campina Grande, Brazil, 2016; pp. 644–649.
30. Agt-Rickauer, H.; Kutsche, R.D.; Sack, H. Domore—a recommender system for domain modeling. In Proceedings of the 6th International Conference on Model-Driven Engineering and Software Development MODELSWARD, Funchal, Madeira, Portugal, 22–24 January 2018; Volume 1, pp. 71–82. [\[CrossRef\]](#)
31. Stephan, M. Towards a cognizant virtual software modeling assistant using model clones. In Proceedings of the 2019 IEEE/ACM 41st International Conference on Software Engineering: New Ideas and Emerging Results (ICSE-NIER), Montreal, QC, Canada, 25–31 May 2019; pp. 21–24. [\[CrossRef\]](#)
32. Segura, Á.M.; Pescador, A.; de Lara, J.; Wimmer, M. An extensible meta-modelling assistant. In Proceedings of the 2016 IEEE 20th International Enterprise Distributed Object Computing Conference (EDOC), Vienna, Austria, 5–9 September 2016; pp. 1–10. [\[CrossRef\]](#)
33. Pérez-Soler, S.; Guerra, E.; De Lara, J.; Jurado, F. The rise of the (modelling) bots: Towards assisted modelling via social networks. In Proceedings of the 2017 32nd IEEE/ACM International Conference on Automated Software Engineering (ASE), Urbana, IL, USA, 30 October–3 November 2017; pp. 723–728. [\[CrossRef\]](#)
34. Pérez-Soler, S.; Guerra, E.; de Lara, J. Collaborative modeling and group decision making using chatbots in social networks. *IEEE Softw.* **2018**, *35*, 48–54. [\[CrossRef\]](#)
35. Mussbacher, G.; Combemale, B.; Kienzle, J.; Abrahão, S.; Ali, H.; Bencomo, N.; Búr, M.; Burgueño, L.; Engels, G.; Jeanjean, P.; et al. Opportunities in intelligent modeling assistance. *Softw. Syst. Model.* **2020**, *19*, 1045–1053. [\[CrossRef\]](#)
36. Saini, R.; Mussbacher, G.; Guo, J.L.; Kienzle, J. Domobot: An ai-empowered bot for automated and interactive domain modelling. In Proceedings of the 2021 ACM/IEEE International Conference on Model Driven Engineering Languages and Systems Companion (MODELS-C), Fukuoka, Japan, 10–15 October 2021; pp. 595–599. [\[CrossRef\]](#)
37. Saini, R.; Mussbacher, G.; Guo, J.L.; Kienzle, J. Automated, interactive, and traceable domain modelling empowered by artificial intelligence. *Softw. Syst. Model.* **2020**, *21*, 1015–1045. [\[CrossRef\]](#)
38. Burgueño, L.; Clarisó, R.; Gérard, S.; Li, S.; Cabot, J. An NLP-based architecture for the autocompletion of partial domain models. In Proceedings of the International Conference on Advanced Information Systems Engineering, Melbourne, VIC, Australia, 28 June–2 July 2021; pp. 91–106. [\[CrossRef\]](#)
39. Li, T. Aircraft Assembly Process Design for Complex Systems Installation and Test Integration. 2019. Available online: <https://dSPACE.lib.cranfield.ac.uk/handle/1826/20074> (accessed on 27 December 2023).
40. NASA. Glossary of Aerospace Terms. Available online: <https://www.grc.nasa.gov/WWW/k-12/TRC/glossary.htm> (accessed on 27 December 2023).
41. Duggineni, S. Data Integrity and Risk. *Open J. Optim.* **2023**, *12*, 25–33. [\[CrossRef\]](#)
42. Tarjan, R. Depth-first search and linear graph algorithms. *SIAM J. Comput.* **1972**, *1*, 146–160. [\[CrossRef\]](#)
43. Pan, S.J.; Yang, Q. A survey on transfer learning. *IEEE Trans. Knowl. Data Eng.* **2009**, *22*, 1345–1359. [\[CrossRef\]](#)
44. Mikolov, T.; Sutskever, I.; Chen, K.; Corrado, G.S.; Dean, J. Distributed representations of words and phrases and their compositionality. *Adv. Neural Inf. Process. Syst.* **2013**, *26*, 3111–3119.
45. Pennington, J.; Socher, R.; Manning, C.D. Glove: Global vectors for word representation. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), Doha, Qatar, 25–29 October 2014; pp. 1532–1543.
46. Devlin, J.; Chang, M.W.; Lee, K.; Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv* **2018**, arXiv:1810.04805. [\[CrossRef\]](#)
47. Sebastiani, F. Machine learning in automated text categorization. *ACM Comput. Surv. (CSUR)* **2002**, *34*, 1–47. [\[CrossRef\]](#)
48. Sebastiani, F. Text categorization. In *Encyclopedia of Database Technologies and Applications*; IGI Global: Hershey, PA, USA, 2005; pp. 683–687. [\[CrossRef\]](#)
49. Nadeau, D.; Sekine, S. A survey of named entity recognition and classification. *Linguisticae Investig.* **2007**, *30*, 3–26. [\[CrossRef\]](#)
50. Li, J.; Sun, A.; Han, J.; Li, C. A survey on deep learning for named entity recognition. *IEEE Trans. Knowl. Data Eng.* **2020**, *34*, 50–70. [\[CrossRef\]](#)
51. Sportiche, D.; Koopman, H.; Stabler, E. *An Introduction to Syntactic Analysis and Theory*; John Wiley & Sons: Hoboken, NJ, USA, 2013.
52. De Ruiter, A.H.; Damaren, C.; Forbes, J.R. *Spacecraft Dynamics and Control: An Introduction*; John Wiley & Sons: Hoboken, NJ, USA, 2012.
53. Meseguer, J.; Pérez-Grande, I.; Sanz-Andrés, A. *Spacecraft Thermal Control*; Elsevier: Amsterdam, The Netherlands, 2012.
54. Roe, R.R., Jr. *NASA-STD_6016A*; Standard Materials and Processes Requirements for Spacecraft. NASA Technical Standard: Washington, DC, USA, 2016.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.