Development of an OWL Ontology Based on the Function-Oriented System Architecture to Support Data Synchronization Between SysML and Domain Models

Yizhe Zhang[a], Georg Jacobs, Jia Zhao, Joerg Berroth and Gregor Hoepfner
*Institute for Machine Elements and Systems Engineering, RWTH Aachen University,
Eilfschornsteinstr. 18, 52062 Aachen, Germany*

Keywords: MBSE, OWL Ontology, SysML, Data Synchronization, Function-Oriented System Architecture.

Abstract: A promising approach to systems engineering is called Model-Based Systems Engineering (MBSE), which is increasingly accepted to support the development process of complex systems. In MBSE,engineers use Systems Modeling Language (SysML) for formalized modeling of function-oriented system architecture and solution architecture that enable the integration of various domain models to support a seamless system development process. Domain models simulate the physical behaviors of systems with design parameters so as to realize the quantitative analysis and verification of systems. These design parameters often exist in multiple heterogeneous data sources and often rely on manual importation into the SysML model. However, when systems become complex with a large data volume, manual data exchanges between multiple data sources and SysML models become time-consuming and error-prone. Therefore, this work proposes an ontology based on Web Ontology Language (OWL) for managing data in a standardized way and solving heterogeneous problems. Then, an automatic synchronization mechanism is established between the ontology and the SysML model for easy exchange of data. This work demonstrates and validates the proposed approach with a case study of a technical system (i.e., wind turbine system). The contribution of this work is the creation of a standardized OWL ontology that supports an automatic synchronization between the data from multiple domain models and the SysML models, thus reducing the manual effort of dealing with heterogeneous data sources and the risk of data inconsistency occurring in manual data transmission.

1 INTRODUCTION

According to the International Council on Systems Engineering (INCOSE) Vision 2035 for Systems Engineering, Model-Based Systems Engineering (MBSE) plays a key role in supporting product development under ever the increased system complexity of the products (INCOSE, 2022). Over the past few years, many approaches have been proposed to implement MBSE in the industry (Estefan, 2007; Friedenthal, Moore, & Steiner, 2015; Pietrusewicz, 2019; Weilkiens, 2014).

Among these approaches, a practical approach focuses on the function-oriented architecture of system modeling (Drave et al., 10162020; Jacobs et al., 2022). The function-oriented architecture provides a detailed top-down modeling concept that requires traceable links between different SysML model layers of abstraction (i.e., requirement layer, function layer, and solution layer) during the system modeling process. System Modeling Language (SysML) is a general-purpose graphical modeling language used in MBSE, which can not only support the establishment of various abstract layers of the system architecture but also link with external behavioral domain models through integrated interfaces. Therefore, it can be used to create executable central SysML models at the parametric level (Zhang, Hoepfner, Berroth, Pasch, & Jacobs, 2021). The Institute for Machine Elements and Systems Engineering (MSE) at RWTH University developed an extension of the SysML metamodel named motego profile, which assists engineers in developing a SysML model with the function-oriented system architecture for a technical system (Spütz, Jacobs, Konrad, & Wyrwich, 2021).In order to design, analyze and verify the technical system at the

[a] https://orcid.org/0000-0002-5116-350X

Zhang, Y., Jacobs, G., Zhao, J., Berroth, J. and Hoepfner, G.
Development of an OWL Ontology Based on the Function-Oriented System Architecture to Support Data Synchronization Between SysML and Domain Models.
DOI: 10.5220/0012155900003598
In *Proceedings of the 15th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2023) - Volume 2: KEOD*, pages 143-154
ISBN: 978-989-758-671-2; ISSN: 2184-3228

parameter level, it is necessary to set model parameters for the SysML model. SysML provides instance elements to store the physical and geometric parameters of different domain models which come from heterogeneous data sources (e.g., Excel files, Matlab files) (Zerwas et al., 2021). In order to utilize heterogeneous data in the system validation processes within the SysML model, engineers must manually input this data into the corresponding instances within the model. However, when the system becomes complex and contains a large amount of data, this often leads to a lot of manual work, which reduces the efficiency of the system validation process. Further, it is difficult to avoid the risk of data inconsistency caused by manual errors of engineers when transferring data.To solve this problem, this work develops an ontology and a synchronization mechanism to solve the problem of heterogeneous structural data between the domain models and the SysML model. Ontologies based on Web Ontology Language (OWL) can integrate heterogeneous data (Agustina Buccell, Alejandra Cechich, & Nieves R. Brisaboa, 2005) and support the design of semantic-based standardized database schemas (I.T., 2017). Therefore, the following two requirements are placed on the ontology and synchronization mechanism:

- The OWL ontology developed in this work needs to be standardized in order to have the ability to integrate heterogeneous data.
- The ontology structure needs to correspond to the system architecture with the motego profile.

The realization of the synchronization mechanism requires the establishment of a standardized mapping between the system architecture and the ontology structure. According to the mapping relationship, the ontology can be extended with the change in the system architecture. By using the proposed ontology, the data in the SysML model can be updated in time according to the data from domain models.

In order to achieve these objectives, the paper is structured as follows. Section 2 introduces the foundation of the paper and related works. Section 3 describes how an OWL ontology can be created and how it can be synchronized with the motego profile of a function-oriented SysML model. Section 4 demonstrates the proposed approach through case studies of a wind turbine system. Finally, section 5 discusses the superiority and challenges of this work, as well as the outlook for future research. Section 6 concludes this work.

2 BACKGROUND

2.1 Function-Oriented System Development

In order to improve the transparent traceability of the SysML elements during modeling processes, the Institute for Machine Elements and Systems Engineering (MSE) at RWTH Aachen University proposes a more comprehensive MBSE method for function-oriented system development (Jacobs et al., 2022).

2.1.1 System Architecture

The method applies a function-oriented system architecture for the system modeling of technical engineering systems (e.g., WT systems) (Zhang et al., 2021; Zhang, Roeder, Jacobs, Berroth, & Hoepfner, 2022). The functional architecture is in the form of a hierarchy. The top function can be decomposed into sub-functions which serve as the parts of one higher-level function. A system function, or a part of it, can be delimited by a boundary through which physical quantities can enter and leave the function through functional flows. These flows can be energy flows, material flows, or signal flows. The function of the delimited system transforms the physical quantities of the incoming flows into other physical quantities of the outgoing flows. Functions are referred to as elementary functions if the transformation of the flows they represent does not physically decompose further. The solution element describes a general effect or set of general effects that fulfills functions physically. The functions and the solutions are linked by generalization relationships. Therefore, the solutions will inherit the functional flows from the functions they fulfill. This generalization relationship bridges the gap between functional architecture and physical architecture.

2.1.2 Motego Profile

MSE has extended and customized the SysML metamodel based on the functional architecture and named it motego profile, so that engineers can more easily implement functional-oriented system development.

In order to enable the automatic exchange of data in the SysML model based on the profile, a code-based analysis is required. Extensible Markup Language Metadata Interchange (XMI) is an interchange format for SysML models, which is used to help programmers by using SysML to exchange model data with different languages and development tools (Grose, Brodsky, & Doney, 2002).

The SysML model described by XMI is as follows:

- The XMI element describing the SysML model is the <uml:Model> node.
- Each SysML model element is described in a <packagedElement> node or <nestedClassifier> node under the <uml:Model> node.
- The XMI element describes SysML model properties in the <ownedAttribute> node under the <packagedElement> node.

Table 1 summarizes the critical metamodel elements of the motego profile extended from SysML in XMI format.

2.2 Ontology

To integrate heterogeneous data sources, the concept "Ontology" is widely used in computer science and information science. The construction of an ontology requires the application of a systematic and structured methodology to ensure that it serves as an effective tool for standardizing and formalizing concepts. A well-defined ontology construction methodology contributes to the development of high-quality, consistent, and reusable ontologies, which ultimately facilitate communication, collaboration, and integration of knowledge across various domains (Fernández-López, Gómez-Pérez, & Juristo, 1997).

2.2.1 Web Ontology Language

OWL is a knowledge representation language designed to formulate, exchange, and reason with knowledge about a domain of interest. The OWL provides a way to add a human-readable label on classes, properties, and individuals using annotations (P. Hitzler, M. Krötzsch, B. Parsia, P. Patel-Schneider, & S. Rudolph, 2012). Class, individual, and property are fundamental building units of OWL ontologies.

The class describes concepts in a domain that are used to group data with similar characteristics. The individual is also called the instance, which represents a concrete occurrence of any concept in a domain of interest. The property is further divided into two categories: data property and object property. Data properties are relations between instances and values or classes and datatypes. Object properties are relations between two individuals or two classes.

2.2.2 Ontology Definition Metamodel

In order to realize the interoperability between OWL ontologies and SysML models, OMG developed a specification and defined it as Ontology Definition Metamodel (ODM). ODM makes the metamodel architecture of the SysML model applicable to the modeling of ontologies based on OWL (Colomb et al., 2006). The ODM enables the use of a variety of SysML models as starting points for ontology development. Table 2 shows that ODM has a layered correspondence with SysML metamodel architecture (Agustina Buccell et al., 2005):

- Metamodel layer is the language specification layer. Metamodels describe modular units or models of models in the model layer.

Table 1: Metamodel elements of motego profile based on the SysML.

SysML Metamodel elements	Motego profile elements	Motego profile elements in XMI format
«requirement»	«FunctionalRequirement»	<motegoProfile:FunctionalRequirement/>
«block»	«FucntionalArchitecture» «ElementaryFucntion» «SystemSolution» «SolutionElement» «EnergyFlow»	<motegoProfile:FucntionalArchitecture/> <motegoProfile:ElementaryFucntion/> <motegoProfile:SystemSolution/> <motegoProfile:SolutionElement/> <motegoProfile:EnergyFlow/>
Port property	EnergyFlowPort property	<motegoProfile:EnergyFlowPort/>
Part property	Architecture property Element property	<motegoProfile:ArchitectureProperty/> <motegoProfile:ElementProperty/>
Flow property		<sysml:FlowProperty/>
Satisfy relationship		<sysml:Satisfy/>
Generalization relationship		<generalization xmi:type='uml:Generalization'/>
Association relationship		<packagedElement xmi:type='uml:Association' ></packagedElement>

- contains an entity of the metamodel that describes a particular object of a system, e.g., a SysML model and an OWL ontology.
- Instance layer contains the real-world individuals or instances which are used to describe the objects by user data.

Table 2: The three layers of Ontology Definition Metamodel. Adapted from (Agustina Buccell et al., 2005).

Layer	SysML Metamodel Architecture	Ontology Definition Metamodel
Metamodel layer	SysML metamodel	OWL metamodel
Model layer	SysML model	OWL ontology
Instance layer	SysML instance	OWL individual

2.3 Related Works

Various studies have addressed the integration of SysML models and OWL ontologies to enhance the expressiveness, semantic understanding, and efficiency of system design. This section reviews three related works that deal with similar challenges.

State Analysis is a methodology for designing complex control systems. (Wagner et al., 2012) presents ontological definitions of State Analysis concepts and relations, as well as a practical SysML extension to provide greater flexibility. The ontology offers a formal basis for verifying compliance with State Analysis semantics, while the SysML extension enables the application of State Analysis methodology with SysML tools.

(Graves, 2009) focuses on constructing a Knowledge Base (KB) using OWL to represent detailed system design information, such as part occurrences and interconnections between parts. This work proposes translating suitably restricted SysML block diagrams into OWL, preserving their model-theoretic semantics. The resulting design KBs can be developed using engineering design tools and exported to OWL tools for analysis. This approach provides a partial unification of SysML and OWL, which is sufficient for modeling complex systems.

SysML Requirement Diagrams model non-functional requirements that can't be accommodated in the Unified Modeling Language (UML). However, SysML lacks the capability to represent semantic contexts within the design. (Wardhana, Ashar, & Sari, 2020) proposes a model that automatically transforms SysML Requirement Diagrams into OWL files, capturing the semantic context of system design. The transformation process uses a transformation rule and an algorithm to change SysML Requirement Diagrams into OWL ontology files. This approach reduces errors and time-consuming efforts in a manual transformation process.

3 APPROACH

While these studies have made significant strides, there is still a need for an approach that focuses on automating synchronization and management of heterogeneous data in SysML models. Building an ontology and synchronizing data manually is time-consuming, and existing research has not yet implemented the synchronization between the SysML model and the OWL ontology (Jenkins & Rouquette, 2012; Wardhana et al., 2020). In addition, although the ODM realizes the creation of an OWL ontology based on the SysML metamodel architecture, the semantics of the created ontologies are not based on the function-oriented system architecture. Therefore, the heterogeneous data sources from domain models are difficult to cooperate with the function-oriented SysML models. To fill this research gap, this section will further develop the ODM to support the automated creation of OWL ontologies corresponding to the function-oriented system architecture.

3.1 Development of the ODM Based on Function-Oriented System Architecture

3.1.1 Definition of the ODM Profile Layer

The motego profile is no longer a standard SysML but a specific language to support the modeling of engineering systems (Drave et al., 10162020; Spütz et al., 2021). Therefore, the motego profile does not belong to the metamodel layer, nor does it belong to the model layer. It is a conceptual framework for specific standardized system modeling between the SysML metamodel and the SysML model. Therefore, this work proposes an additional layer named the "profile layer" between the metamodel layer and the model layer (see Table 3).

The necessity of the profile layer can be explained by the following points:

- Representation of domain-specific concepts: The profile layer allows the incorporation of domain-specific concepts, which are not covered by the standard SysML metamodel, into the modeling process. This facilitates the creation of more accurate and comprehensive system models by capturing the unique aspects of the engineering systems being modeled.

- Customization and extensibility: The profile layer provides a means for users to customize and extend the SysML metamodel to suit their specific modeling needs. This allows for the development of tailored modeling languages that can better address the requirements of individual engineering domains.
- Bridging the gap between metamodel and model layers: The profile layer serves as an intermediate layer between the metamodel and model layers, enabling a smooth transition between the generic SysML metamodel and the concrete system models. This helps to ensure that the models are consistent with the underlying metamodel, while still allowing for the incorporation of domain-specific concepts.

In the profile layer, an additional OWL profile should also be provided between OWL metamodel and OWL ontology, which corresponds to the SysML profile. This work considers the OWL profile as the basic ontology for generating the OWL ontology. By introducing an additional OWL profile between the OWL metamodel and OWL ontology, it is possible to maintain alignment between the SysML and OWL representations of the engineering systems. This ensures that the ontological structures used to represent the systems are consistent and compatible.

Table 3: The Ontology Definition Metamodel with the profile layer.

Layer	SysML Metamodel Architecture	Ontology Definition Metamodel
Metamodel layer	SysML metamodel	OWL metamodel
Profile layer	SysML profile (Motego profile)	OWL profile
Model layer	SysML model	OWL ontology
Instance layer	SysML instance	OWL individual

3.1.2 Definition of the ODM Profile Based on the Motego

The OWL profile, as a conceptual framework, consists mainly of metamodel elements representing classes and relationships of classes. Based on the critical classes, properties, and relationships in the motego profile, this work designs the corresponding profile-classes and profile-properties of the OWL profile as described in the following sub-sections.

3.1.3 Creating OWL Profile-Classes

The motego profile contains basic modeling class elements that can be used as the stereotypes of system requirements elements, function elements, solution elements, and so on. Therefore, as shown in Table 4,

Table 4: The OWL profile elements corresponding to the motego profile.

Profile layer	Motego profile		OWL profile
Profile-classes	<motegoProfile:FunctionalRequirement/>		Declaration(Class(:FunctionalRequirement))
	<motegoProfile:FucntionalArchitecture/> <motegoProfile:ElementaryFucntion/> <motegoProfile:SystemSolution/> <motegoProfile:SolutionElement/> <motegoProfile:EnergyFlow/>		Declaration(Class(:FucntionalArchitecture)) Declaration(Class(:ElementaryFunction)) Declaration(Class(:SystemSolution)) Declaration(Class(:SolutionElement)) Declaration(Class(:EnergyFlow))
Profile-properties	<packagedElement xmi:type='uml:Association'></packagedElement> (Object profile-properties)	<motegoProfile:ArchitectureProperty/> (Object profile-properties)	Declaration(ObjectProperty (:hasArchtecture))
		<motegoProfile:EnergyFlowPort/> (Object profile-properties)	Declaration(ObjectProperty (:hasEnergyFlowPort))
		<sysml:FlowProperty/> (Data profile-properties)	Declaration(DataProperty (:hasFlowProperty))
	<sysml:Satisfy> (Object profile-properties)		Declaration(ObjectProperty (:Satisfy))
	<generalization xmi:type='uml:Generalization'/> (Object Upper-properties)		SubClassOf(:childClass :parentClass) SubObjectPropertyOf(:childObjectProperty :parentObjectProperty) SubDataPropertyOf(:childDataProperty :parentDataProperty)

the motego profile class elements (e.g., «FunctionalRequirement», «FunctionalArchitecture», «SystemSolution», etc.) are regarded as profile-classes in the OWL profile. These profile-classes have the same names as these motego profile class elements. For example, the motego profile class element «FunctionalRequirement» is defined as the OWL profile-class "FunctionalRequirement".

3.1.4 Creating OWL Profile-Properties

The motego profile contains basic modeling property elements of each class and relationship elements between these classes. Therefore, as shown in Table 4, the critical property (e.g., Architecture property, EngergyFlowPort property) and relationship elements (e.g., Association relationship, Satisfy relationship, etc.) can be regarded as profile-properties in the OWL profile.

In the motego profile, these properties and the association relationship are defined separately, while in the OWL, they are defined jointly as the OWL profile-properties, and are standardized named "has" combined with the name of the property. For example, the architecture property with the association relationship in the motego profile can be used to represent the composition relationship between two «FunctionalArchitecture» classes. While in the OWL profile, "hasArchitecture" property is defined to represent this composition relationship. This choice facilitates the modeling of nested structures, reflecting the inherent containment relationships in the SysML model, and supports our goals of data synchronization and consistency. Most of the relationships of the motego profile can also be regarded as the OWL profile-property, which is named the same as the name of the relationships. For example, a satisfy relationship can be named "Satisfy" property in the OWL upper-ontology. It is worth noting that the generalization relationship does not need to be customized in OWL, and it can be represented directly using the built-in ontology relations (e.g., "SubClassOf"). As mentioned in section 2.2.1, the OWL profile-property can be divided into two types, which are object profile-property and data profile-property. Data profile-properties associate classes with data (e.g., strings, numbers, etc.), while object profile-properties associate classes with other classes. For example, "hasFlow" is the data property, and "hasArchitecture" is an object property, since the parts of the super-class are usually other sub-classes.

3.2 Development of Mapping Relationship for Data Synchronization

The development of the profile layer enables OWL ontology to be created and structured in a standardized way according to the function-oriented system architecture. In order to automatically synchronize the ontology with the SysML model, the specific mapping rules need to be designed at the model and instance layer.

3.2.1 Mapping Between OWL Ontologies and SysML Models

At the model level, engineers create SysML model elements based on the motego profile (e.g., requirement elements, function elements). In order to establish the synchronization mechanism between the SysML model and the OWL ontology, the following steps need to be considered:

- Creating the corresponding ontology elements based on the SysML model element ID & Name.
- Defining the relationship between the OWL ontology elements and the OWL profile elements based on the classifier relationships between the SysML model elements and the SysML profile elements.

Table 5 shows the mapping relationship between the SysML model and the OWL ontology with the function-oriented system architecture. A class needs to be declared in the OWL ontology at first. This class is defined according to the element ID of the model element, so as to ensure that the ontology can correspond to the unique element in the SysML model. In addition, the names of the model element can be stored in ontology annotations. Secondly, the stereotype of the model element is classified as the SysML profile element. To present this classifier relationship, the "SubClassOf" association can be established in the OWL ontology between the ontology elements and the profile.

3.2.2 Mapping Between OWL Individuals and SysML Instances

At the instance layer, engineers create instances with specific parameter values for SysML model elements. In order to establish a synchronization mechanism between a SysML instance and an OWL individual, the following steps need to be considered:

Table 5: The OWL ontology elements corresponding to the SysML model elements.

Model layer		SysML model	OWL ontology
Model elements	Creating OWL ontology elements	<packagedElement xmi:type='uml:Class' xmi:id=ElementID name=ElementName></packagedElement>	Declaration(Class(:ElementID)) AnnotationAssertion(rdfs:label : ElementID ElementName)
	Defining the relationship between ontology elements and profile	<motegoProfile:ProfileElement base_Class=ElementID/>	SubClassOf(:ElementID :ProfileElement)

Table 6: The OWL individuals corresponding to the SysML instances.

Instance layer		SysML instance	OWL individual
Instances	Creating OWL individuals	<packagedElement xmi:type='uml:InstanceSpecification' xmi:id=InstanceID name=InstanceName></packagedElement>	Declaration(NamedIndi vidual (:IndividualName))
	Defining the relationship between individuals and ontology elements	<classifier xmi:idref=ElementID/>	ClassAssertion(:Elemen tID :IndividualName)
	Setting the data mapping	<slot definingFeature=FeatureID> <value value=DataValue> </value> </slot>	DataPropertyAssertion(: FeatureID :IndividualName DataValue)

- Creating OWL individuals for the OWL ontology element.
- Defining the relationship between the OWL ontology element and the OWL individual based on the corresponding relationship between the SysML model element and the SysML instance.
- Setting the mapping between the property values of the OWL individual and the feature values of the SysML instance.

Table 6 shows the mapping relationship between SysML instances and OWL individuals with the function-oriented system architecture. An individual needs to be created and named in OWL ontology first. Next, an OWL ontology element as the classifier for the individual need to be defined. Then, based on the mapping relationships in the model layer, the corresponding instance can be found in SysML. Finally, a mapping relationship between individual property values and instance feature values is created based on the corresponding feature's ID.

3.2.3 Synchronization Mechanism Based on Mapping Relationships

As shown in Figure 1, in order to establish the synchronization mechanism, the ontology architecture needs to be defined first. In this work, the definition of the architecture takes place at the profile layer, and designers need to create the relevant OWL profile according to the SysML profile. Specifically, the motego profile elements with XMI format are parsed through the object-oriented programming language (i.e., Java). These parsed elements include the customized classes, properties, and relationships, which can be temporarily saved in a Java container. These elements are then converted and saved as structured OWL profile elements according to the corresponding transformation rules given in Table 4.

Secondly, the original OWL ontology can be easily generated based on the SysML model according to the mapping relationships given in Table 5. The synchronization of the model occurs after the SysML model has been changed. As shown in Figure 1, the SysML model elements and the OWL ontology elements are parsed separately by the programming language, including the ID and name information of these elements. Then, the information is saved in the Java container for the profile. By comparing the difference between the information in the Java container of SysML and the Java container of OWL, the Java container for the model can be updated and used to generate the updated OWL ontology. Specifically, the mechanism requires comparing the differences (e.g., name changes) between model elements and ontologies with the same ID. In addition, when SysML model elements are added or deleted, the corresponding elements in the OWL ontology can also be added or deleted along with them.

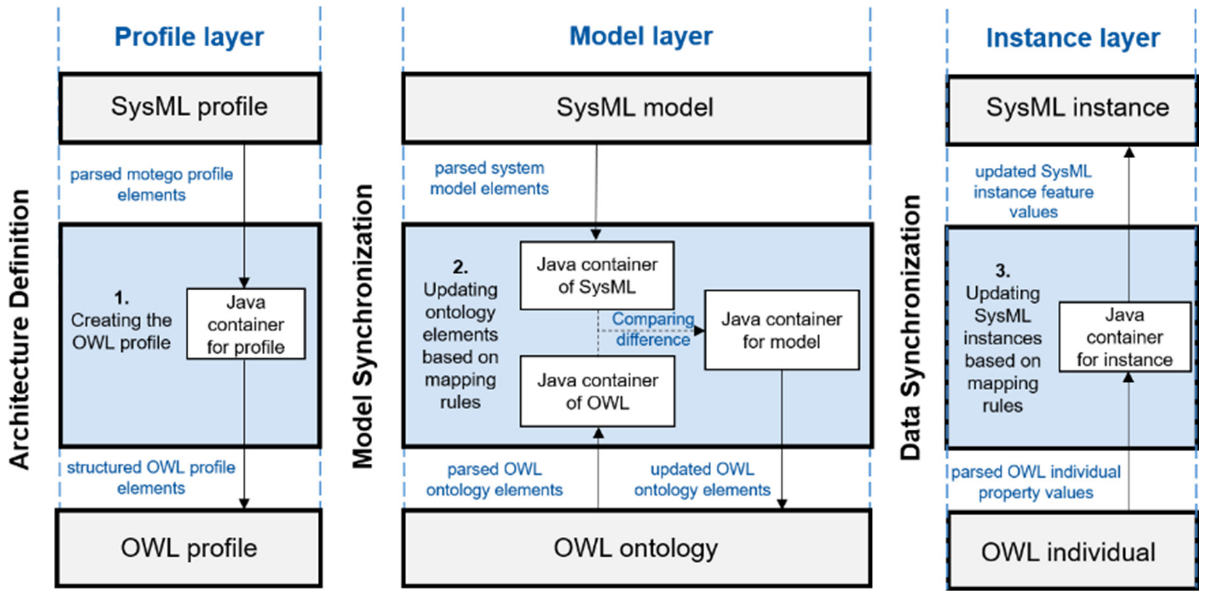

Profile layer
Model layer
Instance layer
Architecture Definition
Model Synchronization
Data Synchronization
SysML profile
parsed motego profile elements
1. Creating the OWL profile
Java container for profile
structured OWL profile elements
OWL profile
SysML model
parsed system model elements
2. Updating ontology elements based on mapping rules
Java container of SysML
Comparing difference
Java container for model
Java container of OWL
parsed OWL ontology elements
updated OWL ontology elements
OWL ontology
SysML instance
updated SysML instance feature values
3. Updating SysML instances based on mapping rules
Java container for instance
parsed OWL individual property values
OWL individual

Figure 1: The synchronization mechanism between SysML & OWL based on mapping relationships.

Finally, in order to synchronize the data in the instance layer, the corresponding SysML instance and OWL individual need to be found according to the mapping relationships given in Table 6. The synchronization of data occurs after OWL individual changes. As shown in Figure 1, the individual for which data properties need to be synchronized is parsed in the programming language, and the property values are then stored in the Java container for the SysML instance. These property values are then synchronized to the corresponding feature values of the corresponding SysML instance.

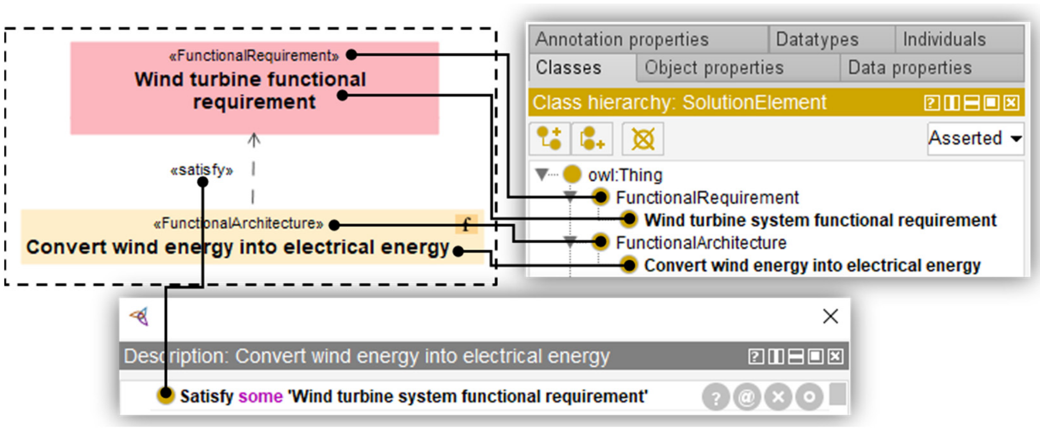
4 CASE STUDIES

This section demonstrates an OWL ontology based on the function-oriented system architecture, and the application of the designed OWL ontology to support the data synchronization between a SysML instance and the data source (i.e., Excel) from domain models. In this work, the SysML model is built in a system modeler tool (i.e., Cameo Systems Modeler), and OWL ontology is created by using an open-source ontology editor (i.e., Protégé).

4.1 Demonstration of an OWL Ontology Based on Function-Oriented System Architecture

In this case study, a simple wind turbine OWL ontology is demonstrated, which is automatically created based on the wind turbine SysML with the function-oriented system architecture. In this case study, we present a comprehensive wind turbine OWL ontology, automatically created based on the wind turbine SysML with a function-oriented system architecture. While this example is intentionally simplified for ease of understanding, it’s worth noting that the mapping method provided in this paper theoretically allows for the addition of more objects and parameters to make the model more complex. Additionally, we have considered the implementation of a physics-based model of the wind turbine in domain model to demonstrate synchronization with that model, showcasing the robustness and adaptability of our approach.

For the wind turbine SysML model, the requirements need to be modeled first. As shown in Figure 2, the SysML model stereotype «FunctionalRequirement» is a motego profile element, which is automatically created as a profile-class “FunctionalRequirement” in the OWL profile according to the mapping relationships. A SysML requirement model element named “Wind turbine system functional requirement” has the classifier «FunctionalRequirement». According to Table 5, this requirement model element is represented in OWL as a sub-class of the profile ontology “FunctionalRequirement”. Protégé describes the ontology model as a tree model by structuring profile elements and ontology elements in parent-nodes and children nodes.


«FunctionalRequirement»
Wind turbine functional requirement
«satisfy»
«FunctionalArchitecture»
Convert wind energy into electrical energy
Annotation properties
Datatypes
Individuals
Classes
Object properties
Data properties
Class hierarchy: SolutionElement
Asserted
owl:Thing
FunctionalRequirement
Wind turbine system functional requirement
FunctionalArchitecture
Convert wind energy into electrical energy
Description: Convert wind energy into electrical energy

Satisfy some 'Wind turbine system functional requirement'

Figure 2: Demonstration of the SysML model architecture between the requirement & function layer in the OWL ontology.

By analogy, a function model element named “Convert wind energy into electrical energy” is represented in OWL as a sub-class of the profile ontology “FunctionalArchitecture”. The function model element has a satisfy relationship with the «FunctionalRequirement» “Wind turbine functional requirement” in the SysML model. This relationship is defined as the object profile-property “Satisfy”, which is used to represent that the function ontology element “Convert wind energy into electrical energy” satisfies the requirement ontology element “Wind turbine functional requirement”. In the wind turbine SysML model, the function model element “Convert wind energy into electrical energy” is classified as a «FunctionalArchitecture» which can be broken down into two «ElementaryFunction» (i.e., “Convert wind energy into mechanical energy” and “Convert mechanical energy into electrical energy”). These function model elements can be transformed and represented in the OWL ontology with a hierarchical model tree (see Figure 2). The relationship between the super-function and its sub-functions is transformed into an object profile-property “hasArchitecture” of the OWL super-function ontology element. In addition, each function model element has the Port property. Still taking the «FunctionalArchitecture» “Convert wind energy into electrical energy” as an example, the “Port 2” typed by “ElectricalEngergy_Output” is an EnergyFlowPort property of the function element in the SysML model. In the OWL ontology, this property is transformed into an object profile-property “hasEnergyFlowPort” of the function element (see Figure 3).

In the same way, solution model elements can also be transformed and represented in the OWL ontology with a hierarchical model tree (see Figure 3). In the wind turbine SysML model, function and solution elements are linked by generalization relationships. This relationship is directly represented as “SubClassOf” in the OWL ontology.

4.2 Demonstration of the Data Synchronization Between SysML and OWL

In this section, the case study demonstrates when engineers change the property values of the individual or choose other individuals, the feature values of the instance in the SysML model also change synchronically.

As shown in Figure, assuming the «EnergyFlow» “ElectricalEnergy_Output” has two flow properties, “Voltage” and “Current”. These two flow properties can be specified in an instance. For example, the instance named “Instance1” has the flow properties “Voltage” and “Current” with the original feature values 220.0V and 5.0A. The corresponding individual can be created in OWL according to the mapping relationships given in Table 6, which is named “Individual_A”.

Protégé provides the functionality of synchronizing ontology individuals with external data sources, such as the “csv.” file from the Excel model. When the data from the data source is imported into the OWL ontology, a new individual (i.e., “Individual_B”) can be generated. This individual has the latest parameter value (i.e., Voltage = 220.0V & Current = 10.0A). These parameter values in the new ontology individual can be synchronized with the “Instance1” in the SysML model. During the synchronization procedure, the corresponding parameter values (i.e., “Voltage” and “Current”) of the SysML instance “Instance1” will be updated accordingly.

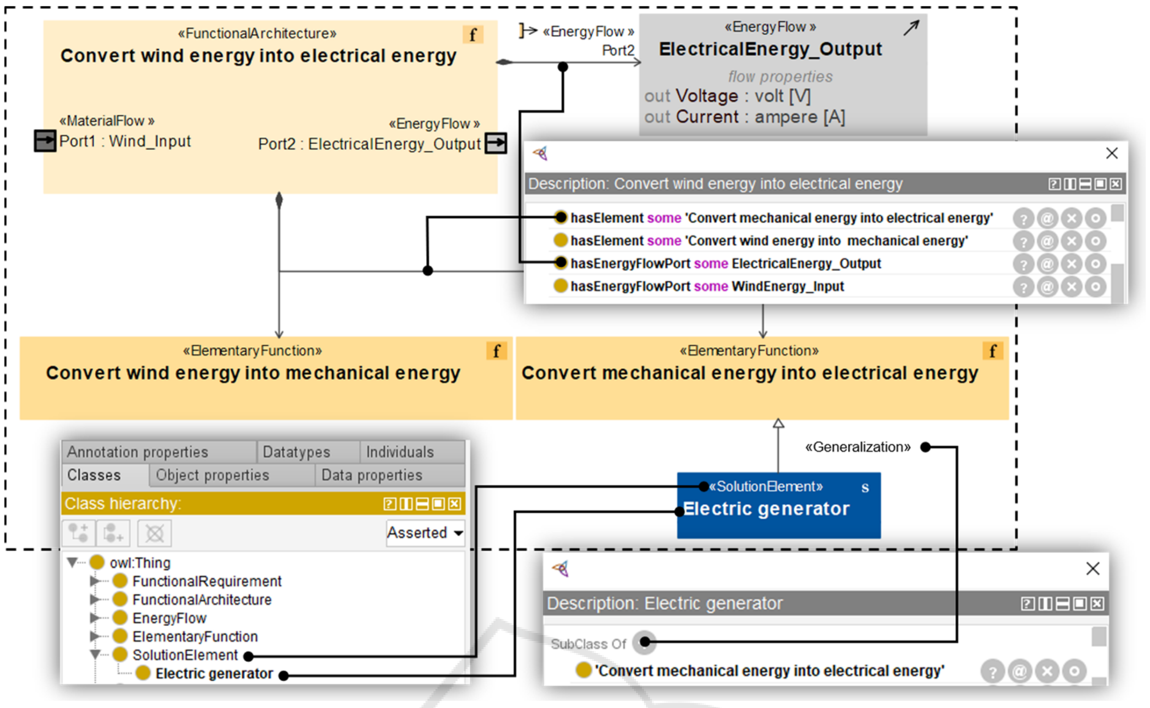

«FunctionalArchitecture»
Convert wind energy into electrical energy
f
«MaterialFlow»
Port1 : Wind_Input
«EnergyFlow»
Port2 : ElectricalEnergy_Output
«EnergyFlow»
Port2
«EnergyFlow»
ElectricalEnergy_Output
flow properties
out Voltage : volt [V]
out Current : ampere [A]
Description: Convert wind energy into electrical energy
hasElement some 'Convert mechanical energy into electrical energy'
hasElement some 'Convert wind energy into mechanical energy'
hasEnergyFlowPort some ElectricalEnergy_Output
hasEnergyFlowPort some WindEnergy_Input
«ElementaryFunction»
Convert wind energy into mechanical energy
«ElementaryFunction»
Convert mechanical energy into electrical energy
«Generalization»
«SolutionElement»
s
Electric generator
Annotation properties
Datatypes
Individuals
Classes
Object properties
Data properties
Class hierarchy:
Asserted
owl:Thing
FunctionalRequirement
FunctionalArchitecture
EnergyFlow
ElementaryFunction
SolutionElement
Electric generator
Description: Electric generator
SubClass Of
'Convert mechanical energy into electrical energy'

Figure 3: Demonstration of the SysML model architecture between function & solution layer in the OWL ontology.

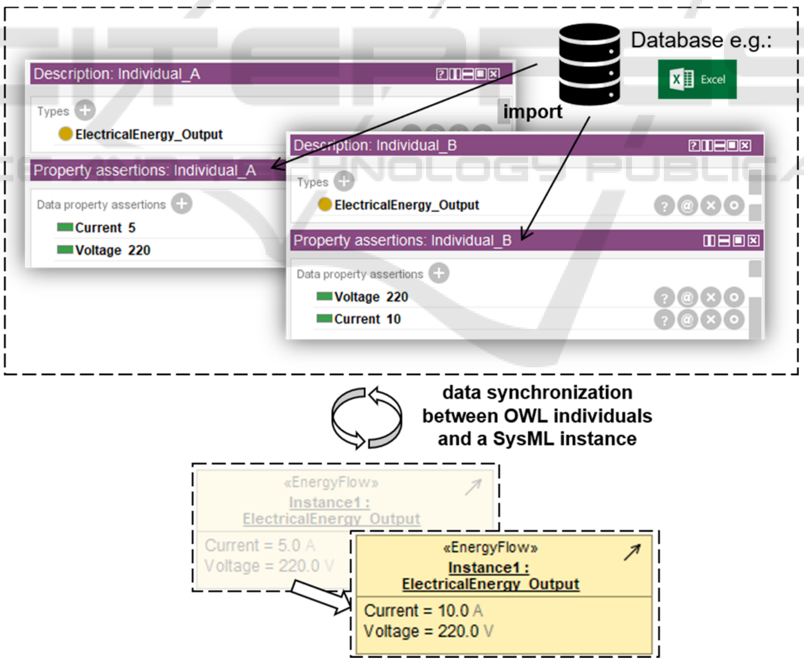


Figure 4: Demonstration of the data synchronization between SysML model and OWL ontology.

5 DISCUSSION

The case studies show that the developed OWL ontology can be used to support the data synchronization between heterogeneous data sources from domain models with the SysML model. Our method offers a mapping mechanism that, in theory, can be expanded by adding more objects and parameters to increase the complexity of the model. Although the wind turbine system case study presented here is simplified for clarity, the approach can be extended to more complex systems.

The proposed approach has significant advantages over manually setting up the data in the SysML model. First, the structure of the ontology corresponds to the function-oriented system architecture, and the data semantic with explicit relationships can be automatically generated, thereby efficiently organizing heterogeneous data sources. Second, engineers can automatically synchronize the ontology with the SysML model based on the mapping relationships implemented by executing Java code, thereby reducing manual efforts and the risk of data inconsistency caused by human mistakes. This work takes the wind turbine system as an example to demonstrate how OWL ontology is created and how data from heterogeneous data sources synchronized with instances in the SysML model. Although the cases presented in the work are limited and very specific, the work is considered scalable due to the normalized semantic definition of the ontology and the corresponding mapping relationships that can be extended. Although in the case studies, the heterogeneous data sources are automatically imported through the specific OWL ontology editing software (i.e., Protégé), the OWL (Web Ontology Language) provides a standardized approach for representing, sharing, and reusing domain knowledge. This allows us to integrate information from different data sources seamlessly. Consequently, our method exhibits a high degree of generalizability.

The utilization of OWL as a middle layer in our approach offers distinct advantages that enhance the robustness and efficiency of the mapping process. OWL's rich expressive power allows for the representation of complex relationships and constraints within the data, facilitating consistency checking and validation of the models. This ensures that the synchronized data adheres to predefined rules and semantics, reducing the likelihood of errors and inconsistencies. Moreover, the standardized nature of OWL enables seamless integration and sharing of domain knowledge, promoting reusability and scalability of the approach. While the current work focuses on the synchronization aspect, the potential for leveraging OWL inference capabilities remains an avenue for future exploration, offering possibilities for more advanced reasoning and analysis within the SysML modeling environment.

However, the current automatic synchronization mechanism still has limitations. First, this work designs the architecture and mapping relationships of ontology based on a state-of-the-art SysML profile (i.e., motego profile). With the further development of the motego profile, designers need to continuously improve and supplement the ontology structure to increase the robustness of the synchronization mechanism. Secondly, this work only implements the scenario where multiple ontology individuals are synchronized with a single SysML instance, and the data synchronization mechanism can be further developed to handle the synchronization scenario with multiple SysML instances. Finally, since the OWL ontology is created based on the SysML model architecture, the model element IDs automatically created by the SysML model are used as identity authentication during data synchronization. However, for the possible existence of multiple SysML models, the data coupling between the SysML model and domain model based on IDs is not reliable enough. (Zerwas et al., 2022) proposed the concept of model signatures. With the development of this concept, algorithms for automatic data compatibility checking need to be introduced into the synchronization mechanism. This will greatly improve the data consistency in the synchronization process proposed in this work.

6 CONCLUSION

SysML models often require parameters from various domains to design, analyze and verify the system. How to realize the automatic synchronization of these heterogeneous data and SysML models is a challenge. The purpose of this work is to provide an approach to facilitate data import and management from external sources to the SysML model. The focus of this work is to develop an OWL ontology for integrating data and synchronizing with the SysML model. The ontology designed in this work is based on the function-oriented system architecture. Therefore, the association structure of data in the ontology is consistent with the system architecture existing in the SysML model, which is easily extensible and beneficial to data synchronization and management. In order to realize the automatic synchronization mechanism, the mapping rules between the system architecture and the ontology are established. In order to obtain general mapping rules, an analysis of the SysML profile is required. The SysML profile extended by the motego profile is mapped to the OWL profile. In addition, it is demonstrated through the wind turbine SysML model cases that the generated ontology can be used to interoperate with data from external data sources of domain models, and these data in the ontology can be synchronized to the SysML model based on mapping rules and unique model element IDs. This work reduces the manual effort of data import and management in the SysML model and avoids the risk of data inconsistency.

REFERENCES

Agustina Buccell, Alejandra Cechich, & Nieves R. Brisaboa (2005). Ontology-based data integration methods: a framework for comparison. *Revista Colombiana de Computación, 6*(1), 1–24, from http://revistas.unab.edu.co/index.php/rcc/article/view/1068.

Colomb, R., Raymond, K., Hart, L., Emery, P., Welty, C., Xie, G. T., & Kendall, E. (2006). The Object Management Group Ontology Definition Metamodel. In C. Calero, F. Ruiz, & M. Piattini (Eds.), *Ontologies for Software Engineering and Software Technology* (pp. 217–247). Scholars Portal.

Drave, I., Rumpe, B., Wortmann, A., Berroth, J., Hoepfner, G., Jacobs, G., et al. (10162020). Modeling mechanical functional architectures in SysML. In E. Syriani & H. Sahraoui (Eds.), *Proceedings of the 23rd ACM/IEEE International Conference on Model Driven Engineering Languages and Systems* (pp. 79–89). New York, NY, USA: ACM.

Estefan, J. A. (2007). *Survey of model-based systems engineering (MBSE) methodologies.*

Fernández-López, M., Gómez-Pérez, A., & Juristo, N. (1997). *METHONTOLOGY: From Ontological Art Towards Ontological Engineering. Proceedings of the Ontological Engineering AAAI-97 Spring Symposium Series:* American Asociation for Artificial Intelligence; Facultad de Informática (UPM).

Friedenthal, S., Moore, A., & Steiner, R. (2015). *A practical guide to SysML: The systems modeling language* (Third edition). Amsterdam, Boston: Elsevier MK Morgan Kaufmann is an imprint of Elsevier.

Graves, H. (2009). *Integrating sysml and owl.*

Grose, T. J., Brodsky, S., & Doney, G. C. (2002). *Mastering XMI: Java programming with the XMI toolkit, XML, and UML. OMG: v.21.* New York, N.Y.: Wiley.

I.T., D. (2017). *Ontology-based knowledge base design.*

INCOSE (2022). Building the Systems Engineering Workforce of the Future: Education, Training and Development of System Engineers.

Jacobs, G., Konrad, C., Berroth, J., Zerwas, T., Höpfner, G., & Spütz, K. (2022). Function-Oriented Model-Based Product Development. In D. Krause & E. Heyden (Eds.), *Design Methodology for Future Products* (pp. 243–263). Cham: Springer International Publishing.

Jenkins, J. S., & Rouquette, N. F. (2012). *Semantically-Rigorous systems engineering modeling using SysML and OWL:* Pasadena, CA : Jet Propulsion Laboratory, National Aeronautics and Space Administration, 2012.

P. Hitzler, M. Krötzsch, B. Parsia, P. Patel-Schneider, & S. Rudolph (2012). OWL 2 Web Ontology Language Primer (Second Edition). *undefined,* from https://www.semanticscholar.org/paper/OWL-2-Web-Ontology-Language-Primer-(Second-Edition)-Hitzler-Kr%C3%B6tzsch/bce0b752e7f5288d8032ca971f72a3c58e810a7c.

Pietrusewicz, K. (2019). Metamodelling for Design of Mechatronic and Cyber-Physical Systems. *Applied Sciences, 9*(3), 376.

Spütz, K., Jacobs, G., Konrad, C., & Wyrwich, C. (2021). Integration of Production and Cost Models in Model-Based Product Development. *Open Journal of Social Sciences, 09*(12), 53–64.

Wagner, D. A., Bennett, M. B., Karban, R., Rouquette, N., Jenkins, S., & Ingham, M. (2012). An ontology for State Analysis: Formalizing the mapping to SysML. In *2012 IEEE Aerospace Conference. Big Sky, Montana, USA, 3 - 10 March 2012* (pp. 1–16). Piscataway, NJ: IEEE.

Wardhana, H., Ashar, A., & Sari, A. K. (2020). Transformation of sysml requirement diagram into owl ontologies. *International Journal of Advanced Computer Science and Applications*, 106–114, from https://www.researchgate.net/profile/helna-wardhana-3/publication/341153151_transformation_of_sysml_requirement_diagram_into_owl_ontologies/links/5ec5c00b92851c11a87ae2fa/transformation-of-sysml-requirement-diagram-into-owl-ontologies.pdf.

Weilkiens, T. (2014). *Systems Engineering mit SysML/UML: Anforderungen, Analyse, Architektur. Mit einem Geleitwort von Richard Mark Soley:* dpunkt.verlag.

Zerwas, T., Jacobs, G., Kowalski, J., Husung, S., Gerhard, D., Rumpe, B., et al. (2022). Model Signatures for the Integration of Simulation Models into System Models. *Systems, 10*(6), 199.

Zerwas, T., Jacobs, G., Spütz, K., Höpfner, G., Drave, I., Berroth, J., et al. (2021). Mechanical concept development using principle solution models. *IOP Conference Series: Materials Science and Engineering, 1097*(1), 12001, from https://iopscience.iop.org/article/10.1088/1757-899X/1097/1/012001.

Zhang, Y., Hoepfner, G., Berroth, J., Pasch, G., & Jacobs, G. (2021). Towards Holistic System Models Including Domain-Specific Simulation Models Based on SysML. *Systems, 9*(4), 76.

Zhang, Y., Roeder, J., Jacobs, G., Berroth, J., & Hoepfner, G. (2022). Virtual Testing Workflows Based on the Function-Oriented System Architecture in SysML: A Case Study in Wind Turbine Systems. *Wind, 2*(3), 599–616.