Enhanced Metamodels Approach Supporting Models for Manufacturing (MfM) Methodology

Domingo Morales-Palma [1], Manuel Oliva [2], Rebeca Arista [3], Carpóforo Vallellano [1] and Fernando Mas [1,4]

[1] University of Sevilla, Camino de los Descubrimientos s/n, 41092 Sevilla, Spain
[2] Airbus Spain, Avenida del Aeropuerto s/n, 41020 Sevilla, Spain
[3] Airbus France, 1, Rond Point Maurice Bellonte, 31707 Blagnac Cedex, France
[4] M&M Group, Av. Inventor Pedro Cawley 31, 11500 Cadiz, Spain

Abstract

The authors have proposed a Models for Manufacturing (MfM) methodology to apply Ontology-Based Engineering (OBE) concepts to industrialization and manufacturing activities. OBE acts as an enabler of knowledge capitalization and management by establishing well-defined domain concepts in terms of terminology, definitions, behaviours, and relationships. The MfM methodology is based on a 3-Layer Model (3LM) framework and is supported by user-friendly modeling tools. It is supported by metamodels that aim to ensure independence from commercial software tools. A metamodel is a model that describes a class of models that provides elements for constructing models and helps to establish an integrated and standard modeling system. Following the 3LM and MfM methodology, manufacturing process information can be represented from metamodel to model in consistence with their data relationships and representations. This paper presents the main novelties introduced in the MfM methodology metamodels, discusses the proposed improvements, and outlines the next steps in the development of the MfM methodology.

Keywords

Metamodels, Models for Manufacturing (MfM), Ontology-Based Engineering (OBE), Manufacturing process modelling

1 Introduction

To maintain control and reduce manufacturing and assembly costs, follow environmental support, reduce total carbon footprint, and secure short time-to-market, products, processes, and industrial resources must be designed in a shorter way reusing existing elements. To follow the previous directives and to improve multidisciplinary design and industrial simulation of complex systems, Ontology-Based Engineering (OBE) is currently considered a novel approach thanks to the digital support and technological advances in computer science [1, 2].

Models for Manufacturing (MfM) is an OBE methodology: computer aided graphical modeling authoring tools are used to define and specify data, functions, behaviours, and semantics of industrial systems. Simulation tools are used to replicate the required behaviour of the system for simulation of complex products. MfM is derived from the concept of Model Based Systems Engineering (MBSE), enabling more robust engineering in terms of models and their associated behavioural abstractions [3]. A preliminary definition of the MfM methodology was published by the authors in [4] and will be introduced in the next section.

The need to formalize all concepts of the ontology layer was clear when Model Lifecycle Management (MLM) [5] began to be developed and implemented. It was necessary to formalize the

FOMI 2022: 12th International Workshop on Formal Ontologies meet Industry, September 12-15, 2022, Tarbes, France
EMAIL: dmpalma@us.es (A.1); manuel.oliva@airbus.com (A.2); rebeca.arista@airbus.com (A.3); carpofor@us.es (A.4); fmas@us.es (A.5)
ORCID: 0000-0001-5816-9528 (A.1); 0000-0001-5945-1162 (A.2); 0000-0001-9382-1593 (A.3); 0000-0001-5903-1336 (A.4) ; 0000-0001-7230-9929 (A.5)

concepts and allow them to change and evolve during the lifecycle. According to Sprinkle et al. [6], to be able to manipulate models, their language needs to be specified as the model of these models, i.e., metamodels.

The authors published a preliminary version of the MfM metamodels in [7]. The present definition of metamodels, based on the previous version, has been improved using the Meta-Object Facility (MOF) standard of the Object Management Group [8]. Metamodels developed include all concepts of the ontology layer and four model types (Scope, Data, Behaviour, and Semantic), as well as the relationships between them.

In summary, this work presents the enhancement of metamodels that support MfM methodology with the objective of maintaining independence with commercial software tools. The paper presents the proposed metamodels and relationships highlighting the difference from the previous publication. The rest of the document is structured as follows: Section 2 contains a brief review of the MfM methodology, workflows around the 3LM and references to the industrial and non-industrial use cases developed; Section 3 presents the metamodels, characteristics, and details; Section 4 discloses the discussion and proposes topics for further research.

2 Review of Models for Manufacturing (MfM) Methodology

This section makes a brief review about the MfM methodology and the workflows over the different layers in the 3LM, gives an introduction to metamodels and presents some works on industrial and non-industrial use cases developed by different authors using MfM. Models developed using commercial software tools for use cases are included to highlight the kind of models developed and the need of metamodels.

2.1 The 3-Layers Model (3LM) as a Framework

The MfM methodology is based on the 3-Layers Model framework. The 3LM ensures the independence between the three layers, isolating the Service Layer, Data Layer, and Ontology Layer between each other. The 3LM framework and layers are represented in Figure 1.

Each layer gathers the same sort of items. Data layer collects all the databases, legacy databases, commercial databases, clouds, data lakes, and interfaces. Ontology layer keeps company knowledge such as scope, data, behavior, and semantic models. Service layer has the software services, legacy or commercial, such as authoring tools, simulation tools, visualizers, data analytics services, dashboards, or space design exploration tools.

The MfM methodology follows a set of golden rules:

- MfM methodology defines a set of metamodels to be applied.
- The MfM methodology is agnostic against tools and does not define any preferred one. It promotes tools that allow writing models and reading, understanding, sharing and discussing models very easily by skilled engineers.
- The MfM methodology encourages a mechanism to manage the lifecycle, configuration, and effectiveness of the model using MLM (Model Lifecycle Management) to meet this requirement.

The core of the 3LM framework is the Ontology layer, where the knowledge of the company is collected, stored, managed and used. The Ontology Layer is made up of Scope models, Data models, Behaviour models, and Semantic models. The Scope model defines the limits of the ontology and holds all main Data model objects and functions which will define the Behaviour models. It is the relevant model for the discussion between engineers and should be kept as simple as possible.

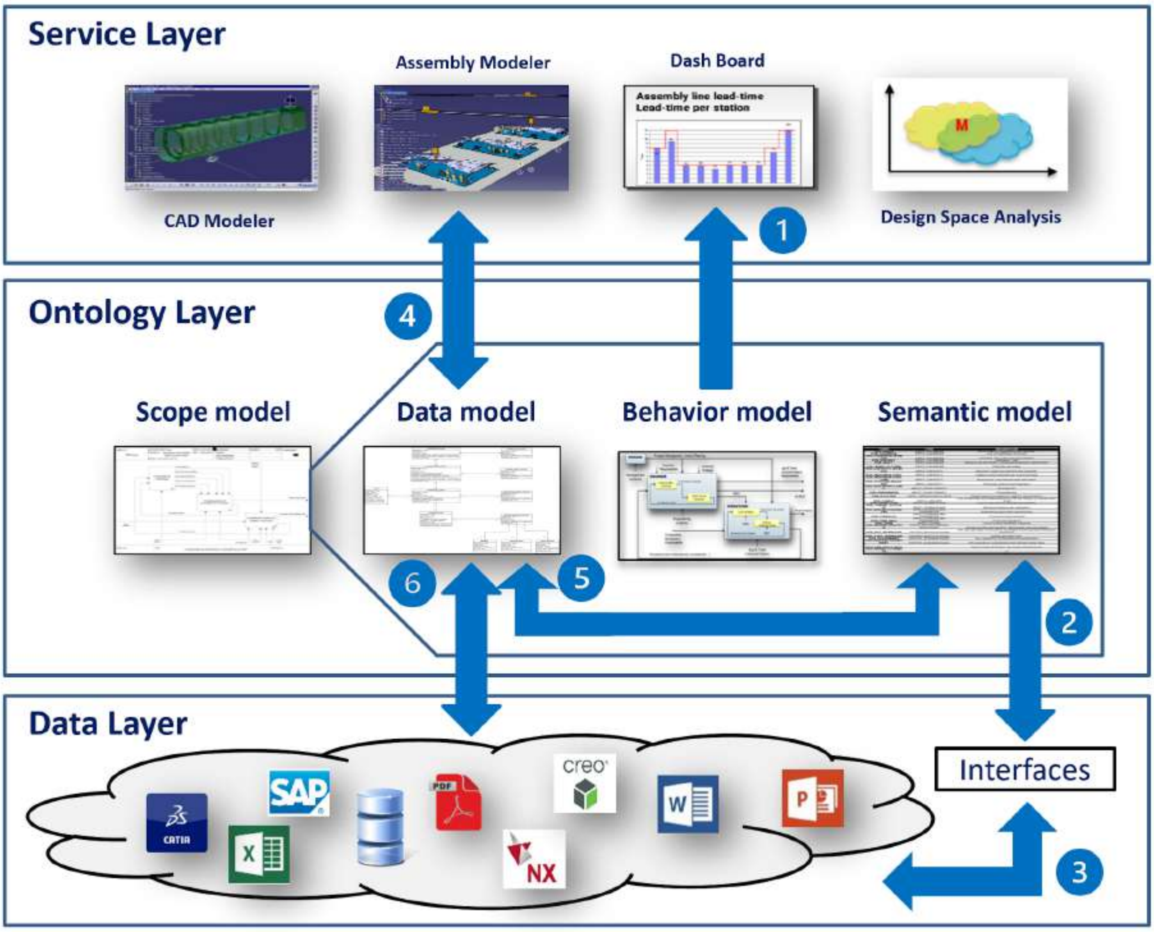


Figure 1: 3-Layers Model (3LM) framework.

The Data model defines the information managed in the selected scope. Using the data objects from the Scope model, engineers can enrich, reuse, and complete the Data model. The Data model allows interaction with the Data layer directly (number 6 in Figure 1) or through the Semantic model and interface (number 5 in Figure 1). The Semantic model keeps the connection between Data model and Data layer through interfaces (numbers 2 and 3 in Figure 1).

The Behaviour model defines the activities and simulations, or other functions presented in the Scope model and completes the full system description with the definition of the applications in the Service layer (number 1 in Figure1).

As mentioned before, MfM methodology is agnostic and is not linked to any software tool. Surveys of ontology software tools show that there are a substantial number of them on the market which are often used [9, 10]. MfM methodology could be implemented in most of them, as far as they comply with the metamodels described in the next section. Arista et al. presented the evaluation of a commercial MLM tool to support MfM methodology [11].

2.2 Introduction to Metamodels

Models are powerful tools for expressing structure, behaviour, and other properties in all areas of engineering. This has led to models being widely used as a description of a product, process, or service. The explicit definition of a modeling language and the manipulation of its corresponding models are linked to the chosen software tools.

To decouple these models from the software tools, it is necessary to specify metamodels of these models, so that any software tool that complies at least with the definition of the metamodel can be used.

Figure 2: Metamodel, Model and Artifact relationship. Adapted from Sprinkle et al. [6].

In Figure 2, adapted from Sprinkle et al. [6], the models from right to left are abstracted by the adjacent model. Artifacts are scope, data, behaviour, and semantics, objects that engineers have being created in the definition of a product, process, or service. Thus, artifacts are abstracted into models that, in turn, conform to metamodels. As you go through these layers of abstraction, the role of each model changes.

2.3 Industrial, Non-Industrial Applications and Model Samples

The MfM methodology is defined and enriched iteratively, following definition and application to different industrial and non-industrial use cases. Parallel to the definition of the methodology, several authors have published research papers for modelling manufacturing systems. Mas et al. researched on aerospace assembly in Final Assembly Lines (FAL) [12, 13] releasing functional and data models and deploying them using data structures available in commercial PLM systems [14].

For process modelling, Cao et al. [15] published a survey about ontologies for manufacturing process modeling to specify how ontologies help to enhance the information interoperability among different manufacturing systems, applications, and stakeholders.

Sanfilippo et al. [16] performed a preliminary ontology focused on the notion of product from the engineering design and manufacturing perspectives and the relationship with PLM contexts. Arista et al. [17] developed a preliminary ontology to support the collaborative engineering process of industrial system design. Based on this research, Arista et al. [18] and Hu et al. [19] improve and go deeper in ontology research to support industrial design for aerospace assembly.

Modelling of Single Point Incremental Forming (SPIF) manufacturing technology for sheet metal parts using MfM was proven in [20], considering the complexity of the CNC technology which usually requires a numerical study to validate the process. Mas et al. [21] presented an ontology built using MfM for aerospace assembly lines in Airbus, with the introduction of a novel way to characterize the adherence concept in the industrial design process of aerospace assembly lines.

Arista et al. [22] conducted a gender diversity analysis using MfM methodology with the objective of approaching this complex social problem in a novel way, applying MBSE techniques and proving the applicability of MfM methodology to non-industrial cases.

A Scope model and Behaviour model was defined and demonstrated through a use case of a global industrial system design for the DA08 artifact, introduced by the authors to evaluate aerospace use cases [23, 24].

Szejka et al. [25] applied MfM methodology to the development of a Product Design and Manufacturing Knowledge-Based System (PDMKBs) based on MfM and the Semantic Web.

Figure 3 shows examples of Scope model, Data model and Behaviour Model realized with commercial software tools that have been presented in the use cases that the authors have published in the last years. Ramus (IDEF0) [26], CMap (Concept maps) [27], and GraphViz (DOT language) [28] software tools have been used.

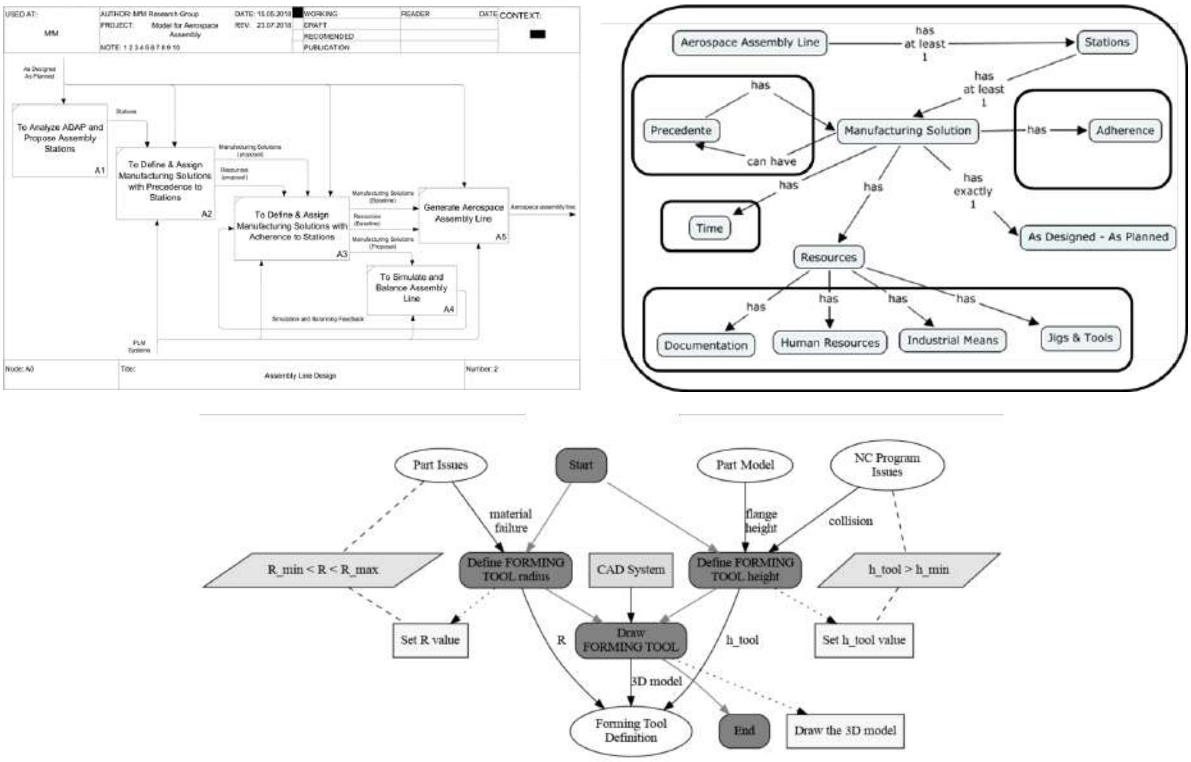


Figure 3: Samples from use cases: Scope model [16] (top left), Data model [16] (top right) and Behaviour model [7] (lower center)

3 Metamodels for the MfM Methodology

The MfM metamodels are defined using the Meta-Object Facility (MOF) standard of the Object Management Group. MOF provides the basis for metamodel definition and is based on a simplification of the UML2 class modeling capabilities [8]. Thus, MfM metamodels are composed of a series of UML class diagrams arranged in different packages. They include all the concepts of the Ontology Layer as well as the relationships between them. They do not include information related to diagram visualization, such as element positions or size.

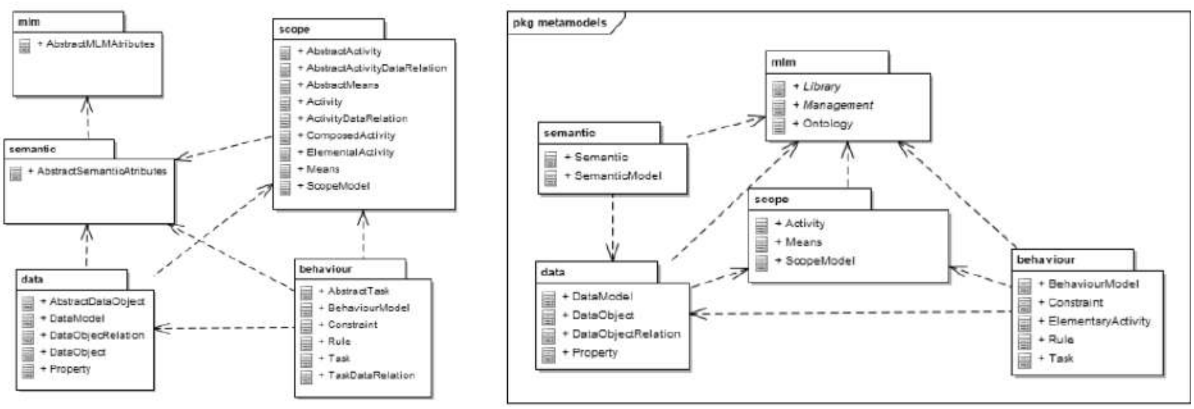


Figure 4: Previous [7] (left) and current (right) versions of the MfM metamodel packages and their dependencies.

Figure 4 shows the previous and current versions of the packages and their dependencies. A package was used to define the metamodel of each type of model (*scope*, *data*, *Behaviour,* and *semantic*) and the *mlm* (Model Lifecycle Management) package was introduced to define concepts related to model management. The content of each package and its evolution in design are described in the following subsections. In general, metamodels have been simplified by removing and/or rearranging some concepts and their relations, as well as introducing new concepts to improve the MfM methodology.

3.1 Scope Metamodel

The Scope model (*ScopeModel* in Figure 5) contains the definition of the main functions or activities (*Activity*) of the system, the resources or means (*Means*) to carry them out and the main data objects (*DataObject*). Figure 5 presents the previous (left) and current (right) proposal of the *scope* package. Note in Figure 5 (right) that concepts that do not belong to the *scope* package are colored gray.

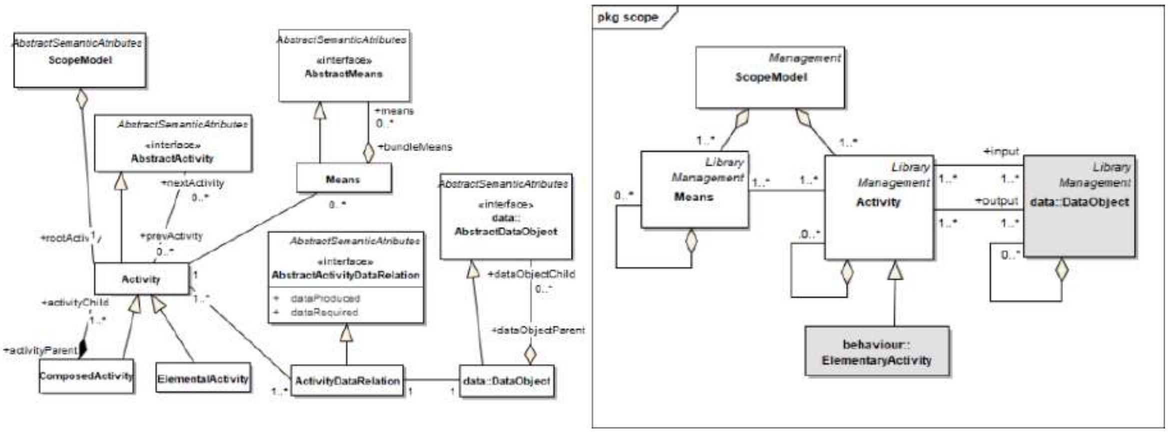


Figure 5: Previous [7] and current MfM Scope metamodel.

An *Activity* can be decomposed into sub-activities, a *Means* set can be grouped into a single resource and *DataObject* can be decomposed into other data objects. In this sense, the Scope metamodel has been simplified by defining in a more convenient way the composition structure of *Activity*, *Means,* and *DataObject* by using UML composite aggregation relationships with themselves. Thus, the previously defined *AbstractX* classes have been removed for clarity and to avoid confusion. Furthermore, older classes (*ComposedActivity*, *ActivityDataRelation*) have been removed when a practical implementation of the metamodel checks that they were not needed.

The relationships between classes and their cardinality have also been refined, as can be seen in Figure 5 (right). In the new metamodel version, *ScopeModel* is composed of all *Activity* and *Means*. An *Activity* can require several *Means* and a *Means* can be used by several *Activity*. These three classes (*ScopeModel*, *Activity,* and *Means*) are the only ones that are defined in the *scope* package. The other two classes (colored gray) are described in the following subsections. They belong to the two metamodels related to the Scope metamodel: *DataObject* of the Data metamodel (*data* package) and *ElementaryActivity* of the Behaviour metamodel (*behaviour* package). The former can be *input* or *output* to/from an *Activity* (an *Activity* can require several *DataObject* and a *DataObject* can be used by several *Activity*). *ElementaryActivity* is an *Activity* that is no longer decomposed into sub-activities (i.e., it has no children in a parent-child structure).

3.2 Data Metamodel

The definition of a first set of data objects (*DataObject*) begins within the development of a Scope model. These objects belong to the *data* metamodel package; however, they are mapped into the *scope* package to show their use by activities, as described in the previous subsection. After the Scope model is partially or fully completed, the Data model can be enriched, reused, and completed by adding properties (*Property*) and new data objects.

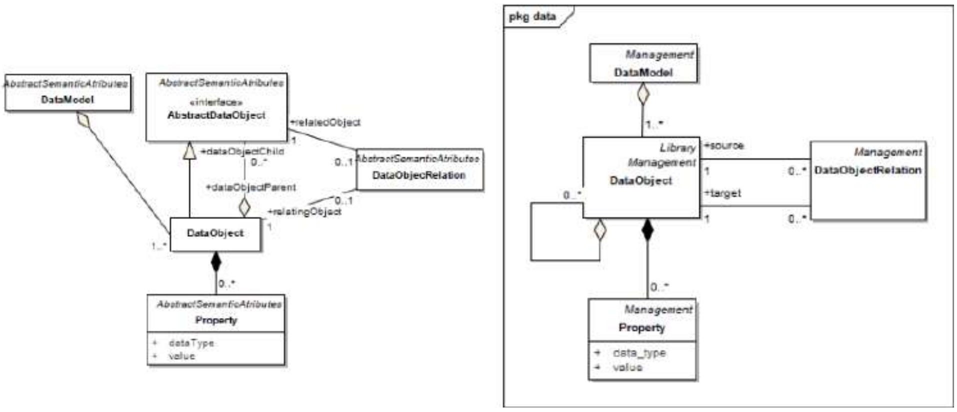


Figure 6: Previous [7] and current MfM Data metamodel.

Figure 6 shows the evolution of the data package. The main change is the removal of the *AbstractDataObject* class, as described above. In the current version, a *DataModel* is composed of all the *DataObject*, which can be arranged with parent-child relationships, and their corresponding *Property*, if any. *Property* has two attributes (*data_type* and *value*) to define primitive data types (e.g., “float” and “2.3”) or any other object types (e.g., “Material” and “Aluminum 2024-T3”). The Data metamodel is completed with *DataObjectRelation* concept that allows relating two *DataObject* and specifying the type of relationship.

3.3 Behaviour Metamodel

The Behaviour model defines the simulation requirements for the company processes or activities previously defined in the Scope model. In this sense, it is assumed that the simulation of an activity that is divided into sub-activities is delegated to and satisfied by the simulation of those sub-activities. Thus, the Behaviour model focuses on defining the simulation requirements of the elementary activities (see Scope model). It is modelled as the set of tasks that allows each elementary activity to be conducted. Task performance is defined by rules or procedures, the data objects that are needed and produced, and the constraints, if any.

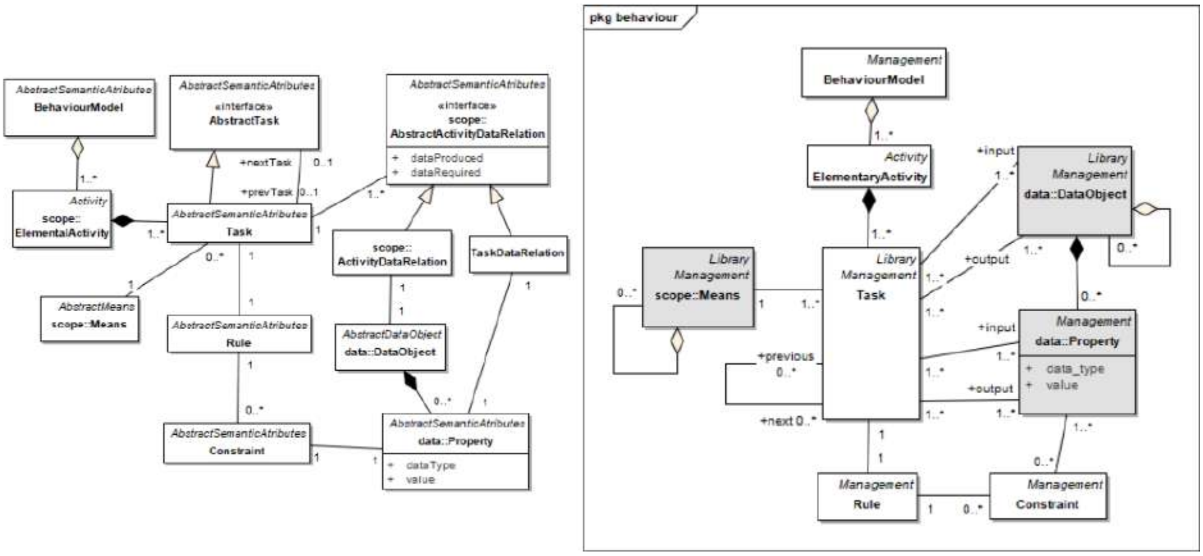


Figure 7: Previous [7] and current MfM Behaviour metamodel.

Figure 7 shows the Behaviour metamodel in its previous and current version. Classes *AbstractTask* and *TaskDataRelation* have been removed from the *behaviour* package for clarity, and the relationships between classes and cardinality have been refined. In the actual version, the *BehaviourModel* is composed of all the *ElementaryActivity*. Each *ElementaryActivity* is made up of at least one *Task*. The mechanisms or means previously related to activities in the Scope model are now reassigned to specific tasks so that each *Task* is linked to the *Means* on which it is performed. Tasks can be related to each other using the previous and next roles, which will allow the system to infer the complete flow of activities and tasks (note that the current version of the Scope model does not define the activity sequence). Each *Task* (what to do) has its own *Rule* (how to do it).

All *DataObject* attached to an *ElementaryActivity* as *input* or *output* in the Scope model are now reassigned to their corresponding *Task*. Moreover, the Behaviour model is enriched from the Data model by selecting all specific *Property* of the *DataObject* that are *input* or *output* of the *Task*. On the other hand, it has been assumed that the performance of a task can be affected by limitations or restrictions of the property values of the data objects. Thus, a *Constraint* concept has been defined that links the *Rule* of the task with the *Property* of the data object.

3.4 Semantic Metamodel

The Data model could be instantiated from real databases defined in diverse ways, formats, and languages. The Semantic model aims to avoid ambiguities in the use of databases, ensure consistency in the connections with the models, and give continuity to the ontologies throughout their life cycle.

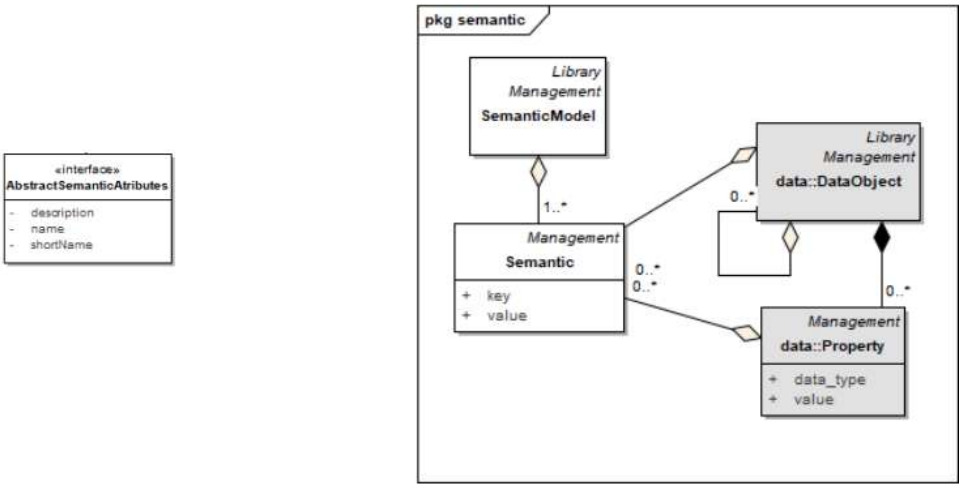


Figure 8: Previous [7] and current MfM Semantic metamodel.

Figure 8 shows the previous and current version of the *semantic* package. In the previous approach, the Semantic metamodel was defined by the class *AbstractSemanticAttributes* that were inherited by the rest of classes, as can be seen in Figures 5, 6 and 7 (inheritance relationships are typed as italic text above the class name). This class adds common attributes such as *name* and *description* to keep a detailed description of all objects in the models. This simple approach may be adequate to help the Data Modeler build interfaces between the Data model and the actual databases; however, it lacks adequate automation of semantic data processing. Therefore, a novel approach has been adopted for the Semantic metamodel (*SemanticModel* in Figure 8) which consists of defining a series of semantic characteristics of *key=value* pairs (modeled as two attributes of the *Semantic* class, see Figure 8) assigned to the data objects and/or their properties.

Currently, for prototyping purposes, a table-based diagram type is used to build Semantic models. Figure 9 depicts an example of a Semantic model for different data object types. For instance, *Connector* object has two parameters: *diameter* and *model 3D*. Following the previous version of the Semantic metamodel, the *description* column may contain some semantic type of information such as the *diameter* is the "inner diameter of the connector tube", and the 3D *model* is a "3D solid model of the connector". In the current version, the semantic information is explicitly defined by *key=value* pairs such as *measure=length*, *unit=mm*, *definition=inner diameter* or *file type=IGES*. In another example, *Manipulator* and *Crane operator* may be objects that stand for similar information. Relevant semantic data may be that the former has *category=robot* and its property *name* is just a robot identification label (definition=label), while the latter has *category=human* and its property *name* is just the person's last name (*definition=last name*). As represented in Figure 9, the Semantic model is made up of all combinations of *key=value* pairs that are used in the Data model.

from Data Model		from Semantic Model	
Data Object *> Property : type*	*description*	*key*	*value*
Connector	Tube connector		
> diameter : *float*	Inner diameter of the tube	measure	length
		unit	mm
		definition	inner diameter
> model 3D : *3d solid*	3D solid model of the tube	file type	IGES
Manipulator	Industrial robot in plant	category	robot
> name : *str*	Name of the robot	definition	label
Crane operator	Factory worker in plant	category	human
> name : *str*	Last name of worker	definition	last name
> age : *int*	Age of worker in years	measure	age
		unit	year

Semantic Model	
key	*values*
measure	{length, age}
unit	{mm, year}
definition	{inner diameter, label, last name}
file	IGES
category	{robot, human}

Figure 9: Example of a Semantic Model instance.

3.4 Model Lifecycle Management (MLM)

In previous work [7], an *AbstractMLMAtributes* class was defined in the *mlm* package to contain a series of common attributes for managing the model lifecycle. In the current approach, this class has been refined and renamed *Management*. Figure 10 shows the *Management* class with its attributes along with all classes that inherit these attributes. In the current version, the attribute list includes *date*, *description*, *id*, *label,* and *version*. However, the definitive definition of attributes can vary during the ongoing implementation phase of the MfM methodology in a PLM system.

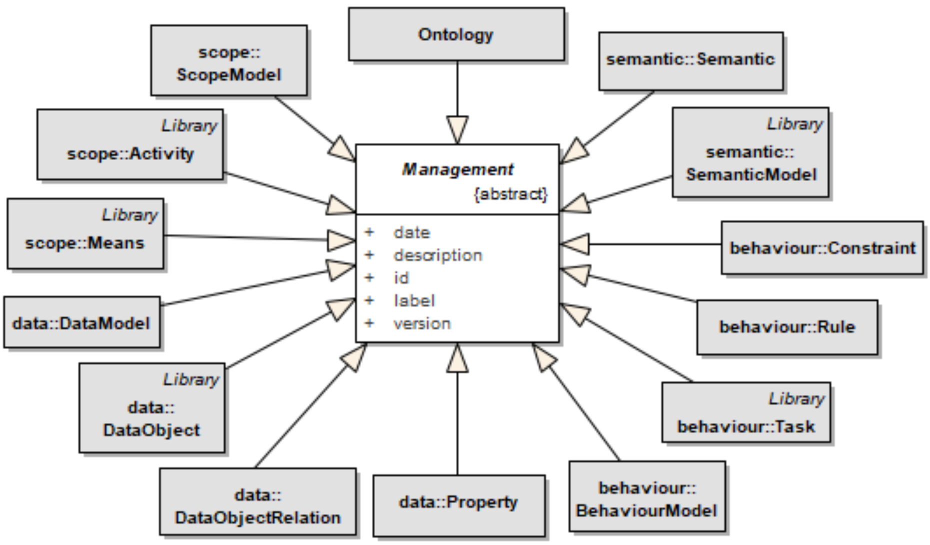


Figure 10: *Management* attributes to manage the lifecycle of the model objects.

Two new concepts (*Ontology* and *Library*) have been added to the *mlm* package. *Ontology* is just a container for a *ScopeModel*, a *DataModel*, a *BehaviourModel*, and an optional *SemanticModel* (the Semantic model is not mandatory in the current approach), as can be seen in Figure 11.

The *Library* abstract class in Figure 11 represents an ontology part that can be reutilized between models. In the current approach, it can be an *Activity*, a *Means*, a *DataObject*, a *Task,* or a *SemanticModel*. An *Activity Library* type may include its related sub-activities and data objects (inputs and outputs), and even the Behaviour (tasks, rules, and constraints) of its elementary activities. *Means* and *DataObject Library* types can be parent-child structures of means and data objects, respectively. A *Task Library* type captures all information associated with the task Behaviour (rules and constraints). A *SemanticModel Library* type aims to reutilize the semantic information (*key=value* pairs defined above) between ontologies.

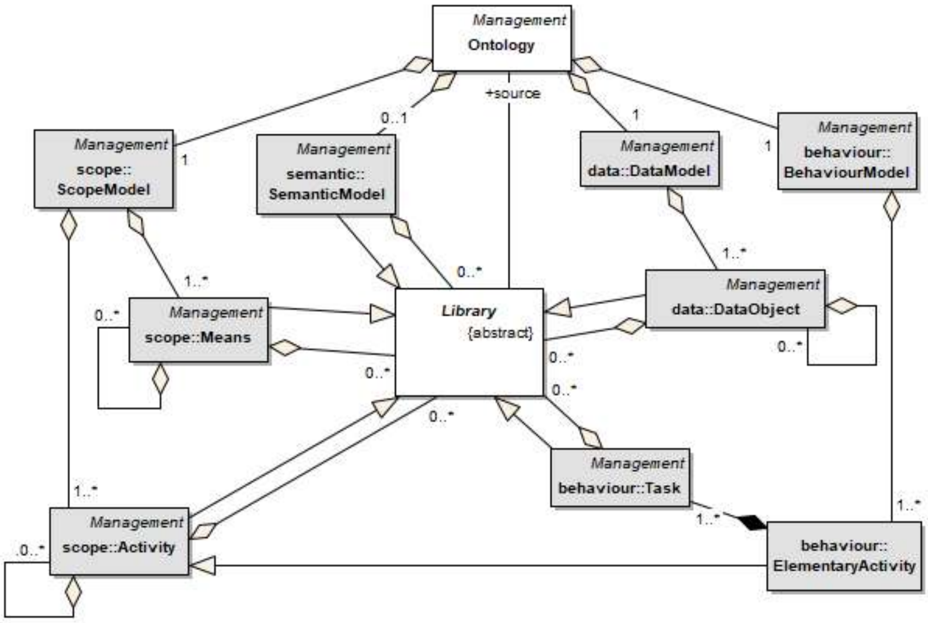


Figure 11: *Ontology* groups all four models (Scope, Data, Behaviour, Semantic) and *Library* allows reusing parts between MfM ontologies.

4 Discussion and Further Work

Models for Manufacturing aims to be an agnostic methodology capable of being used with any commercial or non-commercial modelling software that meets a minimum set of requirements. These requirements are defined by MfM metamodels to set up an independent link to software modelers. Metamodels are designed to represent the set of concepts of a manufacturing ontology in the simplest feasible way. This enables a larger number of modelling software suitable for use in the MfM methodology.

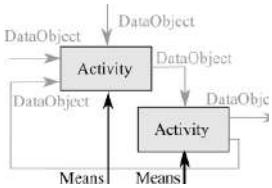
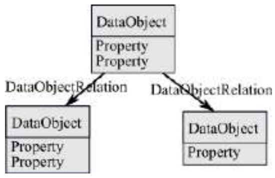
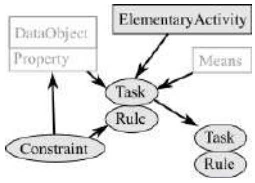
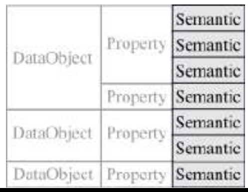
Regarding the revision made in the definition of the metamodels, the changes redefine the original concepts and refine the relationships between them to simplify the UML class diagrams and improve their understanding. For instance, the definition of the concept of *Activity*, *Means*, *DataObject* and *Task* has been rearranged from original *X* and *AbstractX* class pairs that were used to allow several types of relationship between each other (see Figures 5-7).

Other most significant changes come from the feedback of the MfM methodology application tests currently being conducted on real use cases, such as process planning of final assembly lines in the aerospace industry, manufacturing of aeronautical parts by conventional sheet metal forming (e.g., hydroforming) as well as novel incremental sheet metal forming processes. For example, *ComposedActivity*, *ActivityDataRelation* and *TaskDataRelation* classes have been removed from the metamodels (see Figures 5 and 7) as they were found to be redundant or ineffective. Similarly, the Semantic model is completely redefined (see Figure 8) to help more appropriate automation of semantic data processing.

The current version of metamodels also addresses the reuse of parts between MfM ontologies. In this sense, a part of an ontology to be reused groups a main element and several closely related elements of several types. A new *Library* concept has been added for this purpose (see Figure 11). In the current proposal, *Library* is limited to the reuse of the main concepts of the MfM methodology (*Activity*, *Means*, *DataObject*, *Task*) as well as a complete Semantic model. However, further work is needed to determine whether it is appropriate to expand the list of reusable elements as well as how to implement the *Library* concept by delimiting the set of related elements to be reused.

Regarding the real use cases currently in progress stated above, a series of diagram types and software are being used to model them according to the MfM methodology. Table 1 shows a summary of the diagram type, the software used, and an outline of the diagram type for each Ontology Layer model. The selected diagram types fit the requirements of the MfM metamodels, as each has sufficient modelling elements to cover all concepts and relationships defined in the corresponding metamodel.

Table 1
Overview of diagram types, software, and schematics of diagram types

Model	Scope	Data	Behaviour	Semantic
Diagram	IDEF0	Concept map	Ad-hoc diagram	Spreadsheet
Software	Ramus	CMapTools GraphViz DOT	GraphViz DOT	MO Excel LibreOffice Calc
Schema				

Current work is focused on the development of a prototype to simulate virtual manufacturing environments and the real use cases discussed above. The preliminary results of the prototype confirm the success of the proposed MfM methodology, as well as the validity of the metamodels described in this paper. The results of this work are expected to be published soon.

Acknowledgement

The authors thank colleagues from Sevilla University, AIRBUS and M&M Group for their support and contribution during the development of this work.

References

[1] L. Yang, K. Cormican, M. Yu. Ontology-Based Systems Engineering: A State-of-the-Art Review. Computers in Industry 111 (2019) 148–71. doi: 10.1016/j.compind.2019.05.003.

[2] S. El Kadiri, W. Terkaj, E.N. Urwin, C. Palmer, D. Kiritsis, R. Young. Ontology in Engineering Applications. In: Formal Ontologies Meet Industry. FOMI 2015. Lecture Notes in Business Information Processing, 225. Springer, Cham, 2015. doi:10.1007/978-3-319-21545-7_11.

[3] O. Holland. Model-Based Systems Engineering. In: Modeling and Simulation in the Systems Engineering Life Cycle. Simulation Foundations, Methods and Applications. Springer, London, 2015. doi:10.1007/978-1-4471-5634-5_23.

[4] F. Mas, J. Racero, M. Oliva, D. Morales-Palma. A Preliminary Methodological Approach to Models for Manufacturing (MfM). In: Product Lifecycle Management to Support Industry 4.0. PLM 2018. IFIP Advances in Information and Communication Technology, 540. Springer, Cham, 2018. doi:10.1007/978-3-030-01614-2_25.

[5] R. Arista, F. Mas, M. Oliva, J. Racero, D. Morales-Palma. Framework to Support Models for Manufacturing (MfM) Methodology. IFAC-PapersOnLine 2019, 52 (13): 1584–89. doi:10.1016/j.ifacol.2019.11.426.

[6] J. Sprinkle, B. Rumpe, H. Vangheluwe, G. Karsai. Metamodelling: State of -the-Art and Research Challenges. In: Model-Based Engineering of Embedded Real-Time Systems. MBEERTS 2007. Lecture Notes in Computer Science, 6100. Springer, Berlin, Heidelberg, 2010. doi:10.1007/978-3-642-16277-0_3.

[7] D. Morales-Palma, M. Oliva, J. Racero, I. Eguia, R. Arista, F. Mas. Metamodels Approach Supporting Models for Manufacturing (MfM) Methodology. In: Product Lifecycle Management. Green and Blue Technologies to Support Smart and Sustainable Organizations.

PLM 2021. IFIP Advances in Information and Communication Technology, 640. Springer, Cham, 2022. doi:10.1007/978-3-030-94399-8_29.
[8] Meta Object Facility (MOF) Specification, Object Management Group, 2016, URL: https://www.omg.org/spec/MOF.
[9] T. Slimani. Ontology Development: A Comparing Study on Tools, Languages and Formalisms, Indian Journal of Science and Technology (2015), 8(24): 1–12.
[10] J. Lu. MBSE data analysis EPFL, Video, 2021, URL: https://youtu.be/kTxaLVdu-pw.
[11] Arista, Rebeca; Mas, Fernando; Morales-Palma, Domingo; Ernadote, Dominique; Oliva, Manuel; Vallellano, Carpoforo. Evaluation of a commercial Model Lifecycle Management (MLM) tool to support Models for Manufacturing (MfM) methodology. 2022. Presented in IFIP 19th International Conference of PLM. In press.
[12] F. Mas, J. Ríos, J.L. Menéndez, A. Gómez. A Process-Oriented Approach to Modeling the Conceptual Design of Aircraft Assembly Lines. The International Journal of Advanced Manufacturing Technology (2013), 67 (1–4): 771–84. doi:10.1007/s00170-012-4521-5.
[13] F. Mas, M. Oliva, J. Ríos, A. Gómez, V. Olmos, J.A. García. PLM Based Approach to the Industrialization of Aeronautical Assemblies. Procedia Engineering (2015), 132, 1045–1052. doi:10.1016/j.proeng.2015.12.594.
[14] A. Gómez, J. Ríos, F. Mas, A. Vizán. Method and Software Application to Assist in the Conceptual Design of Aircraft Final Assembly Lines. Journal of Manufacturing Systems 40 (2016): 37–53. doi:10.1016/j.jmsy.2016.04.002.
[15] Q. Cao, C. Zanni-Merk, C. Reich. Ontologies for Manufacturing Process Modeling: A Survey. In: Sustainable Design and Manufacturing 2018. KES-SDM 2018. Smart Innovation, Systems and Technologies, 130. Springer, Cham, 2019. doi:10.1007/978-3-030-04290-5_7.
[16] E.M. Sanfilippo. Towards an Ontological Formalization of Technical Product for Design and Manufacturing. In: Formal Ontologies Meet Industry. FOMI 2015. Lecture Notes in Business Information Processing, 225. Springer, Cham, 2015. doi:10.1007/978-3-319-21545-7_7.
[17] R. Arista, F. Mas, C. Vallellano, D. Morales-Palma, M. Oliva. Towards Manufacturing Ontologies for Resources Management in the Aerospace Industry. In Enterprise Interoperability IX- Interoperability in the Era of Artificial Intelligence, 2020.
[18] R. Arista, F. Mas, D. Morales-Palma, M. Oliva, C. Vallellano. A Preliminary Ontology-Based Engineering Application to Industrial System Reconfiguration in Conceptual Phase. In: 11th International Workshop on Formal Ontologies meet Industry Proceedings, FOMI 2021, CEUR Workshop Proceedings (CEUR-WS.org), 2021.
[19] X. Hu, R. Arista, X. Zheng, J. Lentes, J. Sorvari, J. Lu, F. Ubis, D. Kiritsis. Ontology-Based System to Support Industrial System Design for Aircraft Assembly. IFAC-PapersOnLine (2022), 55 (2): 175–80. doi:10.1016/j.ifacol.2022.04.189.
[20] D. Morales-Palma, F. Mas, J. Racero, C. Vallellano. A Preliminary Study of Models for Manufacturing (MfM) Applied to Incremental Sheet Forming. In: Product Lifecycle Management to Support Industry 4.0. PLM 2018. IFIP Advances in Information and Communication Technology, 540. Springer, Cham, 2018. doi:10.1007/978-3-030-01614-2_26.
[21] F. Mas, J. Racero, M. Oliva, D. Morales-Palma. Preliminary Ontology Definition for Aerospace Assembly Lines in Airbus Using Models for Manufacturing Methodology. Procedia Manufacturing (2019), 28: 207–13. doi:10.1016/j.promfg.2018.12.034.
[22] R. Arista, F. Mas. A Preliminary Model-Based Approach for Gender Analysis of Airbus Research Organization. In: 2018 IEEE International Conference on Engineering, Technology and Innovation (ICE/ITMC), 2018, 1–6. doi:10.1109/ICE.2018.8436319.
[23] F. Mas, J. Ríos, A. Gómez, J.C. Hernández. Knowledge-Based Application to Define Aircraft Final Assembly Lines at the Industrialisation Conceptual Design Phase. International Journal of Computer Integrated Manufacturing (2016), 29 (6): 677–91. doi:10.1080/0951192X.2015.1068453.
[24] F. Mas, J. Ríos, J.L. Menéndez, J.C. Hernández, A. Vizán. Concurrent Conceptual Design of Aero-Structure Assembly Lines. In 2008 IEEE International Technology Management Conference (ICE), 1–8. IEEE, 2008.

[25] A.L. Szejka, F. Mas, O. Canciglieri. Towards Knowledge-Based System to Support Smart Manufacturing Processes in Aerospace Industry Based on Models for Manufacturing (MfM). In: Product Lifecycle Management. Green and Blue Technologies to Support Smart and Sustainable Organizations. PLM 2021. IFIP Advances in Information and Communication Technology, 640. Springer, Cham, 2021. doi:10.1007/978-3-030-94399-8_31.
[26] NIST: Integration Definition for Function Modeling (IDEF0). Computer Systems Laboratory of the National Institute of Standards and Technology, December 1993, URL: http://www.idef.com/wp-content/uploads/2016/02/idef0.pdf.
[27] A. Cañas, G. Hill, R. Carff, N. Suri, J. Lott, T. Eskridge, G. Gomez, M. Arroyo, R. Carvajal. CmapTools: A Knowledge Modeling and Sharing Environment. Concept Maps: Theory, Methodology, Technology Proceedings of the First International Conference on Concept Mapping, 2004.
[28] E.R. Gansner, Drawing Graphs with Graphviz, 2011, URL: http://www.ammd.ch/1.pdf.