Design for Efficient Production, a Model-Based Approach

Thomas Polacsek, Stéphanie Roussel, Cédric Pralet, Claude Cuiller

Design for Efficient Production, a Model-Based Approach

Thomas Polacsek
ONERA
Toulouse, France
Thomas.Polacsek@onera.fr

Stéphanie Roussel
ONERA
Toulouse, France
stephanie.roussel@onera.fr

Cédric Pralet
ONERA
Toulouse, France
Cedric.Pralet@onera.fr

Claude Cuiller
AIRBUS
Blagnac, France
claude.cuiller@airbus.com

***Abstract*—The concept of simultaneous engineering has been used for several years in the industry. However, it rarely aims to think the design of the product and the design of its industrial system (the factory) together. In this paper, we address the need to have a global view of architectural design and manufacturing throughout the entire design process. More precisely, we define a model-based approach which makes it possible to evaluate the impact of a product design on its manufacturability. This approach constitutes a first step, in a long-term perspective, to consider several high-level industrial systems and product designs together and choose the one that gives the best performances.**

***Index Terms*—Conceptual Modelling, Model-Driven Engineering, Design for Manufacturability, Enterprise Modelling, Industry 4.0, Aeronautics.**

I. Introduction

For complex products, such as an aircraft, a satellite, or also some car models, it is necessary to have an industrial system (i.e. all the material and non-material means used to manufacture parts and assemble a product: workers, machines, factories, logistics and tools) specifically designed and dedicated to the manufacturing of the product. In this case, the industrial system is built for a single purpose, that is the construction of the specific product. In the cycle of development of such a complex product, the associated industrial system is usually specified only after the product is completely designed. Such a sequential process leads to a very poor consideration of the interactions between the product and its manufacturing. This is generally leading to severe constraints imposed to the manufacturing, which must be ableto assemble the product at any price.

In this paper, we show how a model-based approach can support simultaneous engineering and, more precisely, design for manufacturability (design a product in such a way that it is easy to manufacture).

In simultaneous engineering, the product and its industrial system are designed in the same time and not sequentially. All the actors of a project are involved from the beginning in order to have a global understanding of the objectives and the interactions between all the activities that will have to be performed. Simultaneous engineering is an old concept [1], but it is mainly applied to the optimization of parts, especially in the automotive industry [2], and not to a global optimization of the product and its production system in the context of complex objects such as an aircraft or a boat.

The first advantage of simultaneous engineering is by considering the problem of the design as a whole instead of two sequential sub-problems, the final global solution can be optimized according to many criteria. Among those criteria, a major one is the global cost. In fact, constraints on the industrial system could sometimes be easily lifted by modifying the product design. However, these product modifications can be extremely costly, and sometime impossible, due to sequential aspect of the standard approach. Considering both designs together allows the product design choices to be driven not only by the product but also by the different possible ways of manufacturing it. It provides a global view on the cost at early stages and therefore potential cost-savings.

The second advantage of simultaneous engineering is the agility that such an approach offers. Indeed, having a single framework for design and production allows to assess the impact of a product design on a given industrial system at an early stage and to compare, for a same design, different industrial systems against each other. As a result, the design of product/production couples that are impossible in a traditional sequential approach can here be considered. It should be noted that having a common model also makes it possible, in the case of an already existing product, to study changes in the product design and in the organization of the industrial system in order to be more efficient and reduce the manufacturing process duration.

Today, manufacturing processes already make good use of models as in blueprints, concrete diagrams, Computer-Aided Design (CAD) models, etc. In addition, different tools, and data, are used to define manufacturing tasks and to evaluate industrial performance. On the design side, we also have a set of tools that manage data such as 3D models. Our long-term goal is to use a model-based approach to structure all this information and to support the construction of a complete chain of methods and tools, for assessing the performance of the global solution design during the whole cycle of development. In other words, based on existing data and information, we seek to use conceptual models to propose an approach for thinking together the design of a product and its production.

In this direction, we proposed in [3] a model-based approach to switch from the sequential development to a simultaneous development of the product and its industrial system. Based on Airbus A320 aircraft production data, we identified the key elements of an assembly line and then abstracted these elements into a generic pattern. This generic pattern allows to unify the product design office view and the manufacturing department view, within the context of the A320. In addition, we proposed to use abstraction models of both the aircraft and its industrial system.

In this article, we take one more step towards the previously described global approach by proposing a methodology of models refinement starting at a high-level of abstraction and ending at complete detailed design description level. The idea is to support a simultaneous engineering activity between design and production by allowing to reason both about product design and assembly line at each level of abstraction. Thus, the high level of abstraction corresponds to the early stage of the development cycle and already takes into account the manufacturing point of view, which is new compared to the nowadays approach. The low level of abstraction corresponds to the final stages of the development cycle in which both designs are completely defined and detailed.

The paper is organized as follows. Section II is dedicated to the presentation of the hierarchical methodology. More precisely, we describe the pattern that can be used for each abstraction level and the refinement mechanism. In section III, we present the industrial context and apply this methodology on a simplified example. Finally, we conclude by presenting some related work in sectionIV and the main perspectives of this work in sectionV.

II. Methodology

A. An approach by levels of abstraction

For many complex products, like aircraft, the definition of the final product is an input for the definition of the industrial system. Furthermore, manufacturing architects handle detailed product definition elements, such as screws or cables, that only come up in the completely detailed product design. This state of affairs is problematic, as the aircraft design work begins its cycle of development with general concepts, such as a wing or a landing gear, corresponding to high levels of abstraction. Having the final product design, and consequently the design of the corresponding industrial system, at this level is obviously impossible with the current approach. In order to achieve our goal of simultaneous design, there is a need to model the product and the production designs whatever the level of granularity.

We propose, in the tradition of architectural frameworks, a hierarchical approach based on views where each view corresponds to a layer of abstraction. As [4], we use a mechanism that makes explicit the transition from one level of abstraction to the next. In [3], we proposed a pattern dedicated to the A320 manufacturing use case that could be specialized in concrete classes for each level of abstraction. This specialization was explicitly described for the lowest level of abstraction but we did not define the associated transition mechanism between levels. We present here a more generic version of the pattern, applicable to assembly lines, and that will be explicitly specialized for each level of abstraction of our example in the following sections.

The pattern is composed of the following elements (as is illustrated on Figure 1 with classes in strong line):

- *Physical Element*: physical part of the product. This is the bridge between the design world and the manufacturing world. Depending on the granularity, a Physical Element can be a constituent element of the product (wing, cockpit for an aircraft, power-train, body for a car, etc.), a module (dashboard, engine, etc.) or a very detailed part of the product (screw, cable, etc.);
- *Design*: design of the product, i.e. the definition of all physical elements composing the product and the technical way they are assembled together. It is composed of Physical Elements and has a *level of maturity* feature that is directly linked to the level of abstraction considered in our methodology;
- *Function*: service that a Physical Element contributes to provide. A Physical Element can be associated with several Functions and, conversely, one Function can be provided by several Physical Element. For instance, the ballpoint in a pen could be associated with the writing function, or an inertial measurement unit could be associated to the navigation function or to the attitude, velocity and position functions;
- *Operation*: action that must be performed in order to manufacture the product. A key point here is the *precedence* constraint between operations. It represents the fact that an operation cannot start before some others are finished, for technical reasons. For example, there is a precedence constraint between the operation "*screwing a screw*" and the operation "*making a hole*", or between the operation "*move a furniture*" and the operation "*fix the furniture*". In addition, an operation has a attribute *duration* which is the time required to perform the Operation.
- *Location*: the physical location at which the operation takes place. This can be a very precise localization, as required when assembling several equipment on an aircraft. It can also be a geographical position as in the case of logistic aspects, when operations involve moving an element from one building to another or from one factory to another;
- *Port and Connection*[1]: two classes that allow abstraction of physical elements that are not yet specified at a given design level, but that specify the connections between two Physical Element. More precisely, the Port class is provided with four attributes to specify the flow: *flowType* which indicates the type of flow (air, water, electricity, etc.), *flowIn* and *flowOut*, two booleans which indicate if the flow is incoming, outgoing or both and *flowValue* which is the quantity of flow going through the port.

Because it is a generic pattern, the precision will depend on the level of abstraction. For instance, Operation, that consists to build the product, can be quite abstract for high-level models and intermediate level ones. For example, an aircraft equipping Operation could be "*fix a pipe*" or "*lay the floor*".

[1]We use the *SysML* terminology here.

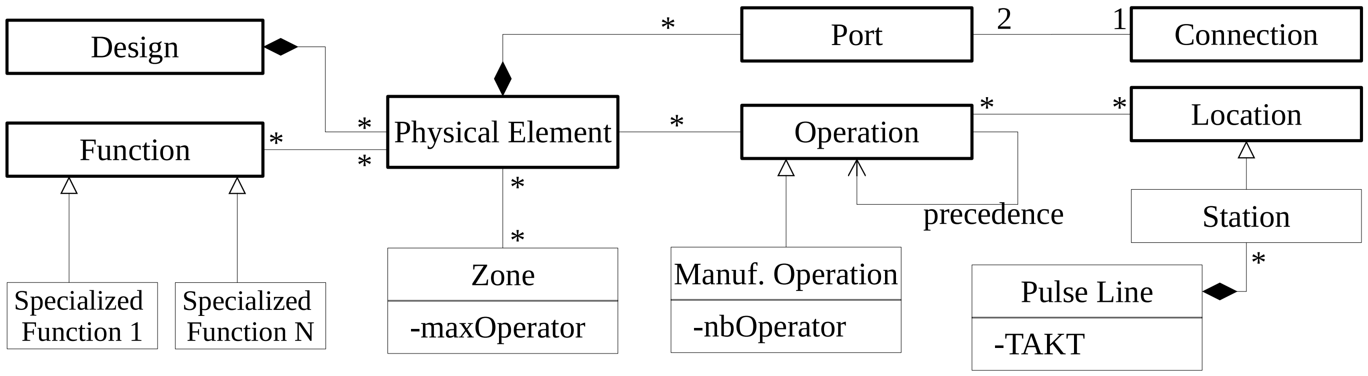

Fig. 1. Example: product and production class diagram.

B. Refinement from one level of abstraction to another

Our methodology is based on the abstraction of the physical elements and their connections. Therefore, the refinement mechanism can be split into the two following processes.

1) Definition and refinement of physical elements and associated operations: A physical element, like an embedded computer, is not just an object in our model. It is also a precise specification that is given through 3D modeling tools. For us, a physical element is therefore linked to a set of data, exogenous to our model, which specifies its weight, size, etc. In the high levels of abstraction, there can be a lot of uncertainty about this data or can even be completely unknown.

For the example of the embedded computer at an early stage of the development cycle, an imprecise idea of its size, weight and location is the only information available. Following the pattern, at this level, we define a very abstract Operation "*computer installation*" with a duration equal to a rough estimation of the installation time. From there, at each step of refinement, as the design data of the element is getting more detailed, it is possible to refine the associated Operation and installation times or replace them with more specific ones, until the final design of the Physical Element is reached. Functions are also refined between levels: at each step, functions from level $N+1$ are all the technical functions necessary to provide services represented by functions of level N (level just above in the hierarchy).

2) Refinement of connections and ports: One of the key points of our approach is that, at a high level of abstraction, we do not need to consider all the real elements that connect Physical Elements together. For instance, at level 0 (highest level), we will consider an airplane wing and an engine that are both Physical Elements connected to each other, but we do not need to explicitly consider the elements that connect them. This abstraction is done using the Port and Connection classes. Back to the example, this connection is modeled by two instances of Port (one for each Physical Elements) and an instance of Connection linked to those ports. In the next step of refinement (level 1), the connection is refined into physical elements, such as a gas hose and an electric harness, which are themselves connected through ports, on one hand to the wing on the other hand to the engine. Formally, each Connection of a level N is refined into a Physical Element in the level $N+1$. Ports in level N remain in Level $N+1$, possibly with updated attribute values, and new Ports and Connections are created between the old Physical Elements and the new ones.

The process is then repeated until the complete description of the product is provided (attachment systems, switches, valves, screws and bolts, ...). In the final step, there is no more Connection type object, nor Port as they have all been refined into Physical Element. All the components of the model are Physical Elements directly connected together, as shown in [3].

III. An aeronautical industrial example

A. Context

In the aeronautics field, the increase in demand, and therefore the increase in aircraft sales, leads to the need for an increase in production rates. This can be achieved either by duplicating assembly lines or by modifying the aircraft design to get a faster manufacturing. To make an optimal financial choice, there is a need to assess the performances of each of these two options. To put it simply, duplicating a factory will be extremely expensive but will make it possible to double the production (which is not necessarily the goal to achieve), whereas a design change will be potentially much cheaper but will allow to manufacture just a little more.

While the performance of an assembly line duplication is quite easy to assess, the impact of a design update can be hard to evaluate and it requires to have a good understanding of the interactions between the design and the production system. On this point, there is a major difficulty for the aircraft design office to anticipate and understand the impact of a design change on the production, as it is managed by different teams at the assembly line level. Moreover, it is not necessarily easy for design teams to identify the bottlenecks of a given assembly line, especially for old aircraft programs for which the digitization process is not complete yet. The current process is really time-consuming as the impact cannot be evaluated until the new design is completely detailed.

Therefore, there is a need to assess the impact of an aircraft design not completely detailed, which can be seen as an abstract design, on the industrial system performance. The methodology described in the paper applies to this case, and being able to model a new design and the current industrial system at a high level of abstraction allows to assess the performance of the global solution. This performance is quantified here through the expected production rate.

B. A dedicated model

We focus more specifically on the use case that aims at increasing production rates on a real assembly line in Airbus. This assembly line is composed of stations, which are the physical space in which the aircraft section is equipped along with the associated tools and parts. In order for the aircraft to be completely equipped, it has to go through all the stations of the assembly line in a given order. The aircraft stays for a given constant duration, called *TAKT*, at each station of the line. When this duration is over, the aircraft goes to the next station of the line or is shipped to another plant if it was in the last station.

To handle this assembly line, we enrich the previously defined pattern with specific technical classes as illustrated on Figure 1[2]. More precisely, we specialize classes of the pattern the following way:

- *Station*: physical space of the assembly line in which the aircraft section is equipped. Station is a specialization of the pattern's class *Location*;
- *Manuf. Operation*: operation inheriting from the pattern's class *Operation* that allows to define specific attributes such as the number of operators required (*nbOperator*) to perform the operation;
- *Specialized Function*: function that enrich the pattern's class *Function* with dedicated attributes.

We also add the following classes:

- *Zone*: physical area of the aircraft in which a manufacturing operation is performed. In fact, an aircraft is divided into manufacturing zones that are characterized by their geographic perimeter. A Zone has a *maxOperator* attribute representing the maximum number of operators that can simultaneously work in the zone ;
- *Pulse Line*: assembly line composed of the stations that altogether deliver completely equipped aircraft sections at every TATK.

C. Refinement illustration

In this paper, we focus on an very simplified example involving two elements that are inserted in a new design: a device (*Eqt*) and an extraction fan. The device considered here is an electronic element such as an embedded computer or the aircraft dashboard (its precise nature is not relevant for this case). The fan is an extraction fan used for cooling *Eqt*. In this section, starting at a given level of abstraction N, we show how this simple example is modeled and refined. The details about the origin of the data presented in the model are not provided in this section but in the next one.

[2]As proposed in [5], we chose, on Figures 2, 3 and 3, to give a more intelligible representation of the object diagrams by playing with shapes and colours. For instance, connection objects are represented by a line and port objets by a box with the same shape as the SysML ports with provided and required interface.

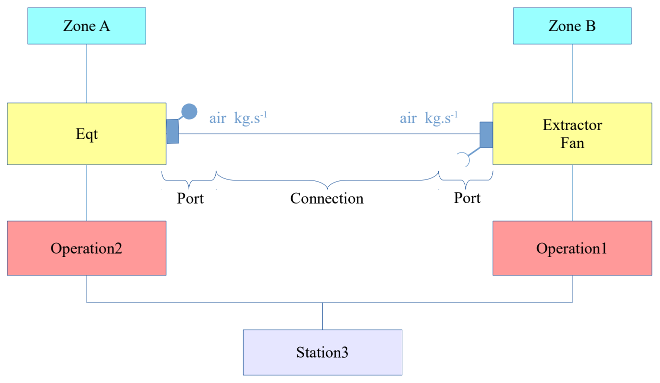

Fig. 2. Example model instantiation at level N.

Figure 2 illustrates the model instantiated at a level N. For legibility reasons, we do not detail the Function, Design and Pulseline objects in this figure. There are only two Physical Elements: the *Extractor Fan* and the *Eqt*. They are coupled together through Ports and Connection objects. We use here a *SysML* notation,i.e. we have chosen to graphically represent the ports by a small rectangle attached to the Physical element and the Connection objects by a line that connects those two ports. The type of the flow going through the two ports is *air*, expressed in $kg.s^{-1}$. The flow is going out of the port of *Eqt* and is going in the port of *Extractor Fan*. At this level, it is already known that these two elements are in two different zones, namely *ZoneA* and *ZoneB*. Operations associated with these elements are abstract: *Operation1* and *Operation2* are respectively the "*installation of* Extractor Fan" and "*installation of* Eqt" . For the moment, there is no precedence relation between the two operations since there is no constraint on the ordering of the installation of these two Physical Elements. Finally, those two operations are performed at station 3 on the assembly line.

A instantiation of this example at the next level, $N + 1$, is illustrated in Figure 3. This instantiation refines level N (Figure 2) in the following way:

- the connection object between *Eqt* and the *Extractor Fan* from level N is replaced here by the element *Ducting Pathway*;
- new connections are created to link this new element to the others: a first one between *Eqt* and *Ducting Pathway* and a second one between *Ducting Pathway* and *Extractor Fan*, along with the associated ports;
- as *Ducting Pathway* goes through the two zones associated with *Eqt* and *Extractor Fan*, it is linked to both of these zones;
- a new operation *Operation3* "*installation of Ducting*" is associated with *Ducting Pathway*;
- even if the operations are quite abstract, it is possible now to define precedence constraints between them. For instance, the *Extractor Fan* must be installed before the *Eqt* and *Ducting Pathway* can be installed;
- duration of *Operation2* and *Operation1* are updated according to new data;

- *Operation1* is still performed in station 3 of the assembly line, but *Operation2* is transferred to station 4. *Operation3* is also performed in station 4.

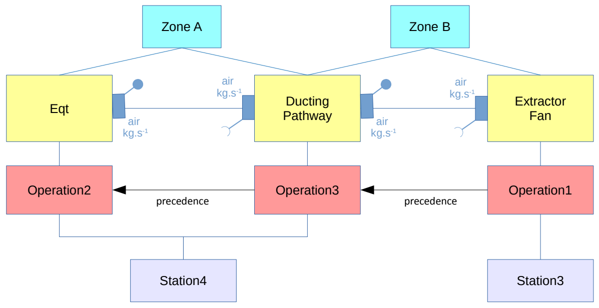

Fig. 3. Example model instantiation at level $N + 1$.

We continue to refine the current example and present the result at level $N + 2$ in Figure 4:

- each Connection of level $N + 1$ is refined by introducing a new Physical Element. More precisely, the *Ducting Pathway* is connected to *Eqt* through an Physical Element called *Socket 1* that represents the way *Eqt* and *Ducting Pathway* are connected to each other;
- these new elements are associated with their respective Zone;
- Operation instances are also refined. This time, the refinement of *Operation2* creates two operations: one related to the installation of *Eqt* (*Operation2.1*), and one related to the installation of the socket (*Operation2.2*); the former is still associated with *Eqt*, while the latter is associated with *Socket 1*. The same applies for *Extractor Fan* and *Socket 2*;
- *Operation3* is refined in *Operation3.1* and it is no longer performed at station 4 but at station 3;
- the precedence constraint among operations of level $N+1$ are also refined. For instance, the precedence between *Operation2* and *Operation3* is replaced by two precedence constraints, respectively between *Operation3.1* and *Operation2.2*, and between *Operation2.1* and *Operation2.2*.

At this level, all the elements are still quite abstract. For instance, the Socket 1 between *Eqt* and *Ducting Pathway* has also to be instantiated into a specific Socket, with which it will be possible to update the associated installation operations. Zones A and B might also be refined and split into smaller zones in the lower intermediate levels. As the cycle of development continues, the new data on the physical elements can allow the architects to realize that the *Ducting Pathway* cannot be one element (for instance, because of its length) but must be split into several pathways connected together. In that case, it is possible to locally backtrack at level N, divide the pathway into as many as required smaller ones, and refine again the relevant part of the model.

The sequence of refinements leads to a final instantiation of the aircraft design and manufacturing in which the Connection and Port objects are not required any more. Indeed, in our approach, Ports represent the abstract notion of interface, the place where two elements are connected. However, the last level represents real objects, at least objects that will be manufactured. All elements must therefore have a physical reality and be attached to a manufacturing operation. In the same way that the concept of man-machine interface is concretized in reality through the keyboard, the mouse,... our concept of Part is concretized by a physical element which is really the interface between two other elements, like a valve or a plug.

So, this final instantiation can first be enriched in order to take into account practical manufacturing aspects, such as logistics, tooling, operators schedule, etc. Therefore, it provides data to the existing manufacturing tools that perform, for instance, logistic simulation. It can also be connected to physical data to perform aircraft design assessments (acoustic, thermal, etc.).

IV. Related Work

In [6], authors define a Domain-Specific Language built on SysML and transformation rules to address discrete event simulators. Their purpose is to allow the business experts to define the indutrial system in their own terms, in details, and to automate the creation of simulations. They do not address the issue of design for manufacturability, they simulate different resource configurations and schedules operations for a single product design. The output of our refinement process could be the input for their approach.

In order to support the industrial system design, and more precisely for the warehouses, [7] gives a SysML methodology based on functional flows. Again, this work is not intended to address simultaneous engineering, but it could be used to enrich our methodology which does not take into account the manufacturing functional flows. Regarding the industrial system, we only focus on assembly line. In [8], the authors propose an architectural framework for designing the manufacturing system, incorporating concepts such as *shop floor*, tooling, machines or artificial hands. Always in the idea of enriching our model, [9] proposes an ontology for lifecyle management relatively manufacturing-oriented.

In our approach, manufacturing operations are an input. At each refinement stage, it is therefore necessary to give the sequence of operations attached to a physical element. Many works are focused on automatically finding the sequence of operations from a physical model [10]–[13]. These works do not consider complex objects, but only small elements, like a catalytic converter or transmission system. However, they could be a good starting point for defining operations in our refining process.

More focused on simultaneous engineering problems, "*Design for Assembly*" methods are techniques that aim to design a product as a set of modules with an efficient assemblability [14]. Works like [15], [16] propose to support the designer, in preliminary designs, to take into account the manufacturability of the product. These approaches are very complementary to ours, they are located upstream of our refining cycle. Indeed, unlike us, they only aim to define the product, not the industrial

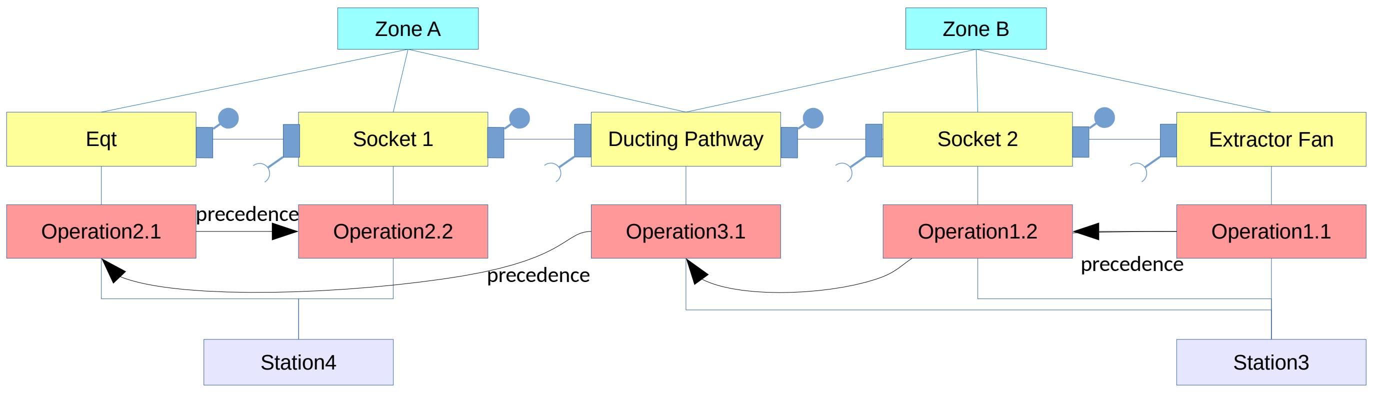

Fig. 4. Example model instantiation at level $N + 2$.

system, and the preliminary designs produced correspond to the models we take as input.

V. Conclusion

In this article, we proposed a model-based approach allowing us to think design and production of an aircraft together. Our contribution is above all methodological. We presented a pattern to support a hierarchical modeling approach and a refinement method.

Regarding automation, even if we do not address the problem in this article, some works have shown the possibility of using logic solvers to design assistance [17], [18]. This type of automatic analysis could be used to validate the integrity of our models and support design tasks. An example of automatic verification could target the flow attributes of Port objects: check that two ports connected to the same Connection object have the same type and that the flowIn and flowOut are consistent.

From a more theoretical aspect, future work could consist in formally characterizing the transition from one model to another in our refinement process. For that, we will be able to rely on many existing works like [19], [20].

Finally, future work may also address the generalization of the approach. We have focused on aeronautical test cases, but our method seems sufficiently generic to be suitable for the manufacturing of other complex products.

References

[1] D. G. Shenas and S. Derakhshan, "Organizational approaches to the implementation of simultaneous engineering," *International Journal of Operations & Production Management*, vol. 14, no. 10, pp. 30–43, 1994.

[2] I. Göpfert and M. Schulz, "Logistics integrated product development in the german automotive industry: current state, trends and challenges," in *LDIC 2012. Proceedings*. Springer, 2013, pp. 509–519.

[3] T. Polacsek, S. Roussel, F. Bouissiere, C. Cuiller, P. Dereux, and S. Kersuzan, "Towards thinking manufacturing and design together: An aeronautical case study," in *Conceptual Modeling - 36th International Conference, ER 2017. Proceedings*, vol. 10650. Springer, 2017, pp. 340–353.

[4] R. Salay, J. Mylopoulos, and S. M. Easterbrook, "Using macromodels to manage collections of related models," in *Advanced Information Systems Engineering, 21st International Conference, CAiSE 2009. Proceedings*, ser. Lecture Notes in Business Information Processing, vol. 5565. Springer, 2009, pp. 141–155.

[5] D. Bihanic and T. Polacsek, "Models for visualisation of complex information systems," in *16th International Conference on Information Visualisation, IV 2012. Proceedings*. IEEE Computer Society, 2012, pp. 130–135.

[6] O. Batarseh and L. F. McGinnis, "Sysml to discrete-event simulation to analyze electronic assembly systems," in *Symposium on Theory of Modeling and Simulation - DEVS Integrative M&S Symposium*, ser. TMS/DEVS '12. Society for Computer Simulation International, 2012, pp. 48:1–48:8.

[7] T. Sprock and L. F. McGinnis, "Analysis of functional architectures for discrete event logistics systems (dels)," *Procedia Computer Science*, vol. 44, pp. 517 – 526, 2015.

[8] N. Benkamoun, W. ElMaraghy, A.-L. Huyet, and K. Kouiss, "Architecture framework for manufacturing system design," *Procedia CIRP*, vol. 17, pp. 88 – 93, 2014.

[9] G. Bruno, D. Antonelli, and A. Villa, "A reference ontology to support product lifecycle management," *Procedia CIRP*, vol. 33, pp. 41–46, 2015.

[10] R. B. Hadj, I. Belhadj, M. Trigui, and N. Aifaoui, "Assembly sequences plan generation using features simplification," *Advances in Engineering Software*, vol. 119, pp. 1 – 11, 2018.

[11] R. Viganò and G. O. Gómez, "Automatic assembly sequence exploration without precedence definition," *International Journal on Interactive Design and Manufacturing (IJIDeM)*, vol. 7, no. 2, pp. 79–89, 2013.

[12] M. Wu, Y. Zhao, and C. Wang, "Knowledge-based approach to assembly sequence planning for wind-driven generator," *Mathematical Problems in Engineering*, vol. 2013, 2013.

[13] F. Demoly, X. Yan, B. Eynard, L. Rivest, and S. Gomes, "An assembly oriented design framework for product structure engineering and assembly sequence planning," *Robotics and Computer Integrated Manufacturing*, vol. 27, no. 1, pp. 33–46, February 2011.

[14] X. Zha, H. Du, and J. Qiu, "Knowledge-based approach and system for assembly oriented design, part i: the approach," *Engineering Applications of Artificial Intelligence*, vol. 14, no. 1, pp. 61 – 75, 2001.

[15] C. Favi and M. Germani, "A method to optimize assemblability of industrial product in early design phase: from product architecture to assembly sequence," *International Journal on Interactive Design and Manufacturing (IJIDeM)*, vol. 6, no. 3, pp. 155–169, 2012.

[16] C. Favi, M. Germani, and M. Mandolini, "A multi-objective design approach to include material, manufacturing and assembly costs in the early design phase," in *Procedia CIRP*, vol. 52, 2016, pp. 251 – 256.

[17] R. Delmas, D. Doose, A. F. Pires, and T. Polacsek, "Supporting model based design," in *MEDI 2011. Proceedings*, ser. Lecture Notes in Computer Science, vol. 6918. Springer, 2011, pp. 237–248.

[18] R. Delmas and T. Polacsek, "Formal methods for exchange policy specification," in *Advanced Information Systems Engineering - 25th International Conference, CAiSE 2013. Proceedings*, ser. Lecture Notes in Computer Science, vol. 7908. Springer, 2013, pp. 288–303.

[19] U. Frank, "Multilevel modeling," *Business & Information Systems Engineering*, vol. 6, no. 6, pp. 319–337, 2014.

[20] M. Balaban, I. Khitron, M. Kifer, and A. Maraee, "Formal executable theory of multilevel modeling," in *Advanced Information Systems Engineering - 30th International Conference, CAiSE 2018. Proceedings*, vol. 316. Springer, 2018, pp. 391–406.