Model-Based Approach for the Simultaneous Design of Airframe Components and their Production Process Using Dynamic MDAO Workflows

Anne-Liza M.R.M. Bruggeman[*], Mikhail Nikitin[†], Gianfranco La Rocca[‡], Otto K. Bergsma[§]
Delft University of Technology, 2629 HS Delft, The Netherlands

During the early design stages of airframe components, many possible design architectures and production methods need to be traded to find the best configuration. Evaluating different production methods can be challenging as different production methods put different requirements on the product to be designed. This paper presents a new methodology that enables the inclusion of manufacturing and assembly in the design process. By extending the architectural design space model with components of the production system, the design choices regarding production are made explicit. Through the modeling of product and production requirements and assigning them a verification method, a dynamic MDAO workflow is formulated. Within a dynamic workflow, the design variables, analysis tools, and constraints change depending on the current design vector. The methodology has been applied to the design and manufacturing of a wing rib in which two manufacturing options were traded: metal machining and composite stamp forming. The dynamic MDAO workflow successfully found the Pareto front for both manufacturing methods. The main benefit is that only one workflow needed to be formulated and executed, whereas previously a separate MDAO workflow needed to be created for each combination of product design and production method. Overall, the newly presented methodology enables the optimization and trade-off between different production methods while ensuring the design complies with the production-specific requirements.

I. Nomenclature

CATMAC = Cost Analysis Tool for Manufacturing of Aircraft Components
CMDOWS = Common MDO Workflow Schema
DfM(A) = Design for Manufacturing (and Assembly)
DoE = Design of Experiments
KADMOS = Knowledge- and graph-based Agile Design for Multidisciplinary Optimization System
KBE = Knowledge Based Engineering
MBSE = Model-Based Systems Engineering
MDAO = Multidisciplinary Design Analysis and Optimization
RVF = Requirements Verification Framework
UCI = User Customized Interface
XDSM = eXtended Design Structure Matrix

II. Introduction

During the early design stages of aircraft components, many different design concepts must be analyzed and traded to find the best design. At the same time, the design and analysis of such components is becoming increasingly complex due to the many different disciplines that must be considered. One of the most challenging disciplines to consider during early design stages, although it has a significant influence on the design and cost of the product, is production.

Many design issues are found during the production stage of a product [1] and are often due to an inadequate design [2]. However, making design changes at the production stage is a costly and time-consuming task. Analyzing

---

[*] PhD Candidate, Flight Performance and Propulsion, A.M.R.M.Bruggeman@tudelft.nl, AIAA Student Member
[†] MSc Student, Flight Performance and Propulsion, M.Nikitin@student.tudelft.nl
[‡] Associate Professor, Flight Performance and Propulsion, G.LaRocca@tudelft.nl
[§] Assistent Professor, Structural Integrity & Composites, O.K.Bergsma@tudelft.nl

the producibility during the conceptual design stage will lead to an earlier identification of production issues, which saves both time and costs. Furthermore, small changes to the design can sometimes reduce the production complexity significantly [3], which results in lower manufacturing costs and higher production rates. However, including production in the early design stages is not straightforward. Many production methods exist and each part of a system can be produced by different process(es). Furthermore, each production method puts different constraints and requirements on the component and vice versa. Managing all these different requirements and ensuring compatibility of design and production (including material selection, part production, and assembly) is challenging, especially when performed manually. Automation and digitalization of design processes can support the designer in the analysis and evaluation of different design options including manufacturing and assembly.

Previous research has focused on the integration of manufacturing into the design process. For example, Bajaj et al. [4] have developed a framework for the manufacturability analysis of printed circuit assemblies. In this framework, an expert system evaluates the design against manufacturing guidelines and provides feedback to the designer on possible design improvement opportunities. Ferrer et al. [5] developed a framework based on axiomatic design principles to systematically capture and formalize manufacturing knowledge and link it to the design. The result is a clear trace from manufacturing properties to the impact on functional requirements and design parameters. A Design for Manufacturing (DfM) framework has also been developed during the Pegasus project by van Dijk et al. [6] to automatically design and optimize plastic injection molds. By generating a Knowledge Based Engineering (KBE) model of the mold and setting up a Multidisciplinary Design Analysis and Optimization (MDAO) problem to design the cooling system of the mold, many different designs could automatically be evaluated accounting for aspects such as product quality, production rate, cost, and environmental impact. Finally, van der Laan [7] developed a KBE framework to automatically take manufacturing costs into account during the design of aircraft movable components. Furthermore, the manufacturability of the ribs was evaluated focusing on the forming process of composite ribs.

The main drawback of these frameworks is the combination of their development complexity and design case specificity. They are generally implemented for a specific type of product and manufacturing process only. Switching to a different manufacturing process is possible, but difficult and time-consuming, while the main goal in the early design stages is to quickly evaluate different options i.e. different product architectures and combinations of material and production and assembly methods. A second drawback is that most of the abovementioned frameworks take the manufacturing requirements into account only indirectly, through design guidelines. As a consequence, a clear trace from requirements to product features and analysis methods for verification and validation is missing.

In this paper, a new design methodology is presented that enables the inclusion of production in the design process, by combining principles of Model-Based Systems Engineering (MBSE) and MDAO. While the MBSE approach offers the required formalism and mechanisms to capture requirements and map them on the product and its manufacturing process, the MDAO principles allow one to account for such requirements and exploit the interaction of the involved disciplinary tools to simultaneously address product design and manufacturing.

The details of the new methodology are described in section III. A proof of concept is provided by applying it to the design and manufacturing of a wing rib. Two manufacturing methods are traded: metal machining and composite stamp forming. The results for this use case are presented in section IV. Finally, the conclusions and recommendations are given in section V.

III. Methodology

The goal of the proposed methodology is to support the generation, comparison, and trading of different designs and manufacturing methods. An overview of the different steps is shown in Figure 1. In the first step, the architectural design space model of the Design for Manufacturing and Assembly (DfMA) problem is created. This model, contrary to the conventional product architecture, where only the functions and solutions of the system of interest are modeled, represents the entire design space, including both product design options, as well as material, manufacturing, and assembly options. A detailed explanation of how the extended model is constructed and how production is included is given in subsection III.A. Once the architectural design space model is complete, all product and production requirements are collected and modeled. As discussed in subsection III.B the verification methods for each requirement, consisting of means of compliance and test cases, are determined also. Based on the requirements and their verification methods, a dynamic MDAO workflow can be formulated. A dynamic MDAO workflow is a workflow that changes based on the current design point, meaning that active design variables, disciplinary tools, and constraints can change during execution. How this is achieved is explained in subsection III.C. Finally, once the dynamic MDAO workflow is formulated, it can be executed to generate results and support decision-making.

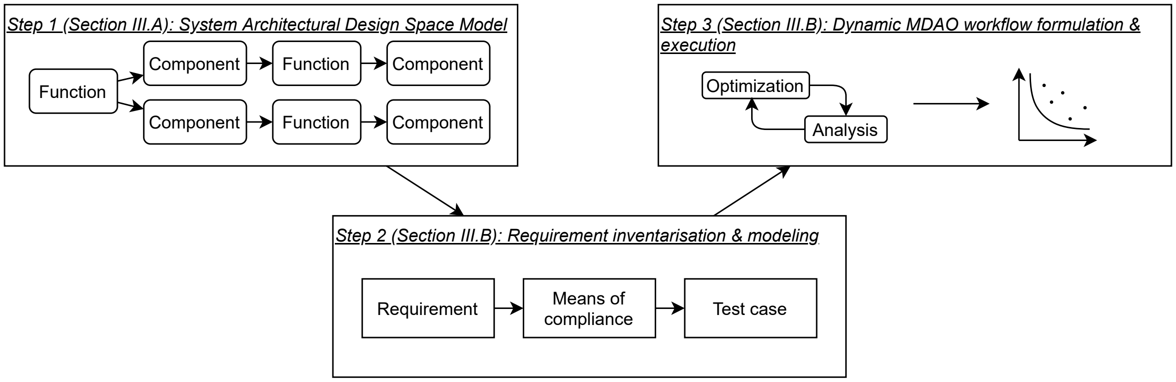


Fig. 1 Schematic overview of the proposed methodology. In step 1, the system architectural design space model is created. In step 2, all requirements are collected together with their verification methods. In step 3, a dynamic MDAO workflow is formulated, based on the set requirements, and executed to generate results, e.g. in the form of a Pareto front

A. Design for Manufacturing and Assembly as an architectural design space problem

As shown in Figure 1, the first step of the methodology involves the creation of an architectural design space model. A simple example of an architectural design space model for an engine is shown in Figure 2.

The creation of the model starts with the top-level functional requirements that the system of interest needs to fulfill. For each top-level requirement, a boundary function can be identified. In the example on the right, the system of interest is an engine and the functional requirement that the engine must fulfill is "*The engine must provide power*". From this requirement, the boundary function "*Provide power*" can be derived.

Once all the boundary functions are collected, one or more components can be identified per boundary function which can fulfill this function. If multiple components can fulfill the same function, a design choice needs to be made. For example, power can be provided by either a gas turbine or an electrical engine. When adding a component to the model, usually new derived functions are introduced. These functions again have to be fulfilled by components. This process of connecting components to functions and deriving new functions from each component is repeated until all (derived) functions have one or more components associated with them. The result is a model of all possible design architectures. By making a choice on which component is going to fulfill which function, a specific architecture instance is created. An example of such an instance is indicated in green in Figure 2. Many different architectural instances can be derived from one architectural design space model.

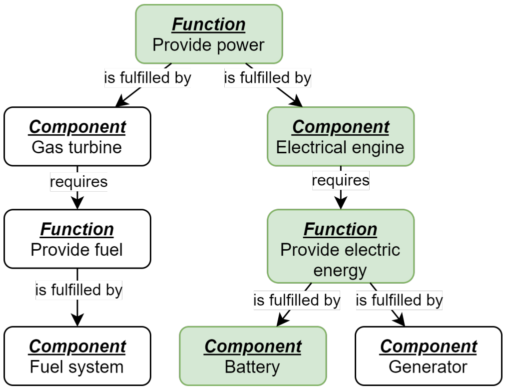
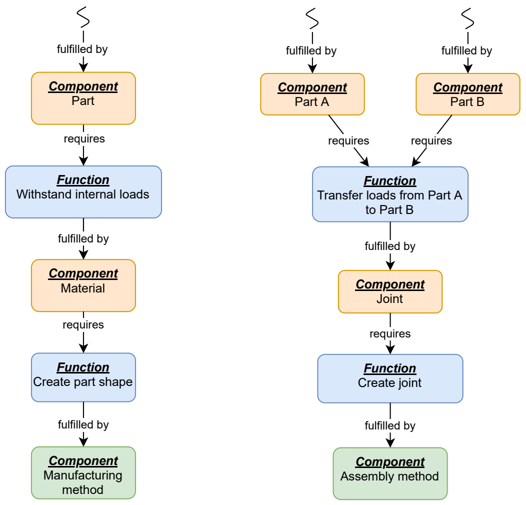


Fig. 2 Architectural design space example for an engine taken from Bruggeman and La Rocca [8]. The functional requirement that needs to be fulfilled is "*The engine must provide power*". Indicated in green is an architectural design instance.

In the example described above, each function was fulfilled by a component of the same system of interest being designed. However, when modeling a DfMA problem, a second system is introduced, namely the production system. In this case, a function can also be fulfilled by a component from the production system. This means that any instance of such an extended architectural design space corresponds to a combination of system architecture, material choice, manufacturing, and assembly method. How this is done for both part manufacturing as well as the assembly of structural elements is shown in Figure 3.

Figure 3a shows the architectural design modeling concept for the manufacturing of a structural part. In this case, the starting point is a part taken from the product architectural design space. The material of each part fulfills a function that is a derived function of the part. For example, for structural components, the part needs to withstand internal loads, which is fulfilled by the material. The material itself introduces the derived function "*Create part shape*". This function can be fulfilled through a manufacturing method which is part of the production system. Therefore, this function is the connection between the product system of interest and the production system.

A similar approach can be taken for the assembly of two parts as shown in Figure 3b. In this case, two parts need to be joined together. For two structural components, the two parts need to be joined because loads need to be transferred between them. Therefore, the derived function of the two parts is "*Transfer loads from Part A to Part B*". Note that if different architectural design choices would have been made and either Part A or B would not have been chosen in the design solution, the derived function "*Transfer loads from Part A to Part B*" and thus the assembly of these two parts will not appear in the architectural design instance. Going back to the example in Figure 3b, the function to transfer loads between the parts can be fulfilled by different types of joints. However, each joint must be created and therefore will introduce the derived function "*Create joint*". This function can then be fulfilled by an assembly method.



(a) Part Manufacturing

(b) Assembly

Fig. 3 Example of an architectural design space including part manufacturing (left) and assembly (right). Note that some derived functions are fulfilled by components from a *different* system, in this case the production system.

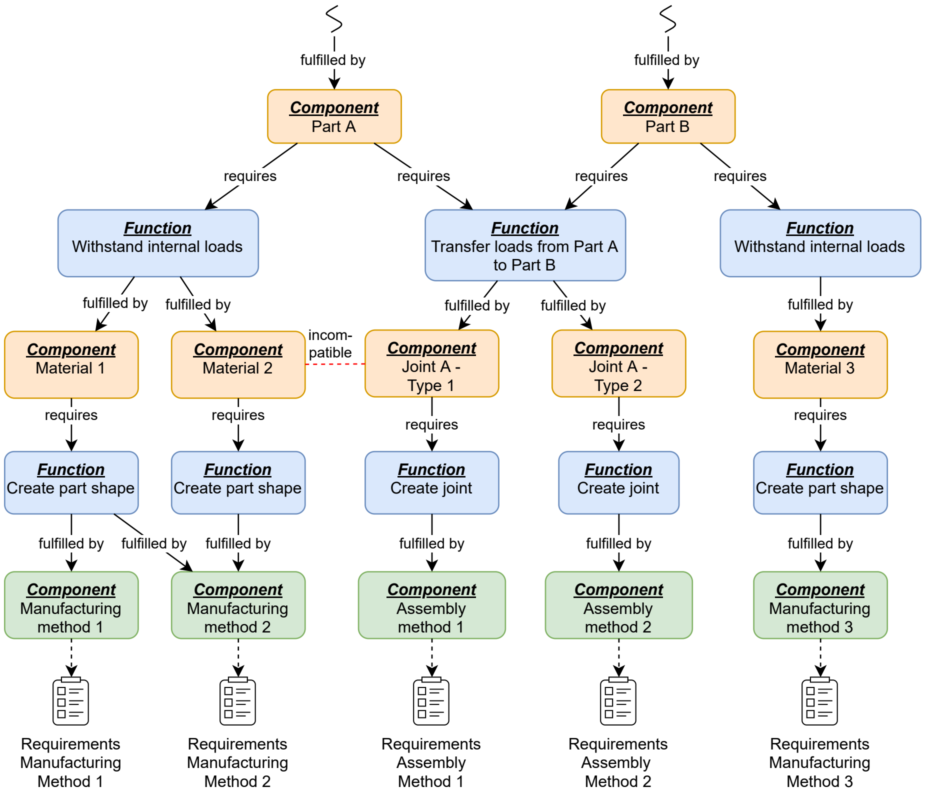


Fig. 4 Example of a system architectural design space model including both part manufacturing as well as assembly. The incompatibility links make sure that feasible design options are evaluated only. As soon as a decision choice is being made for a manufacturing method, the production requirements specific to that manufacturing method become active.

Figure 3 showed how to include part manufacturing and assembly separately in the architectural design space model. However, manufacturing and assembly influence each other and cannot be considered separately. For example, not all assembly methods are suitable for all materials. These incompatibilities also need to be considered in the model. An example of how to combine part manufacturing and assembly is shown in Figure 4. In this example, the manufacturing of two parts and the assembly of the parts with a joint is visualized. In this case, Part A can be produced using two different materials. Furthermore, two manufacturing methods can be used when material 1 is chosen, while only one manufacturing method can be used when material 2 is chosen. Only one material and one manufacturing method is available to produce Part B. For the assembly of Part A and B, two different joint types can be chosen, each joint type having only one assembly method associated to it.

As shown in this figure, incompatibility links exist between Material 2 and Joint type 1. This means that if Material 2 is chosen for Part A, Joint type 1 cannot be chosen as assembly method. Also the other way around holds, if Joint type 1 is chosen as assembly method, Part A cannot be manufacturing using Material 2. Using the incompatibility links between the components ensures that only feasible combinations of part manufacturing and assembly are chosen.

B. Formulating product and manufacturing requirements

The benefit of representing the production system within the same architectural design space of the system of interest is that production methods are now included and modeled as design choices. One can perform more interesting trade-offs to assess product architectural solutions including their production methods. However, as anticipated in section II, a challenge when including production in a trade-off is that each production method puts different requirements on the product being designed. Therefore, each manufacturing and assembly method needs to have its own requirement database, as shown in Figure 4. As soon as a production method is selected for a part or joint, automatically the corresponding set of requirements need to be activated and included in the design process.

Each requirement database needs to be modeled to enable the inclusion of the requirements in the design process. The modeling of the requirements is achieved using the Requirements Verification Framework (RVF) described in Bruggeman et al. [9]. A schematic overview of this framework is shown in Figure 5.

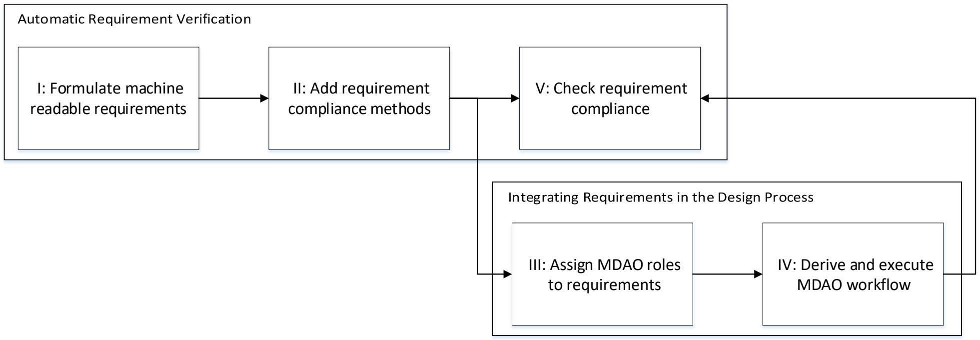


Fig. 5 Overview of the process steps in the Requirements Verification Framework

The first step is to formulate machine-readable requirements. By using patterns, meaning is given to each element of the requirement, thereby making the requirement computer-interpretable. Next, a compliance method or verification method needs to be added to each requirement. A verification method consists of two parts; a means of compliance and a test case. A means of compliance is defined as the agreement between the 'need stakeholder' and the 'responsible stakeholder' on how compliance will be demonstrated. The need stakeholder is the owner of the need from which the requirement is derived; the responsible stakeholder is the one who needs to show compliance with the requirement [9]. The test case is defined as "*the technical implementation of the means of compliance*" [9]. It consists of all the simulation tools (and physical tests) that are needed to show requirement compliance for a given design. As mentioned before, each production method puts different requirements on the product and therefore needs its own requirement database, consisting of the requirements and their associated verification methods. An overview of this concept is given in Figure 6.

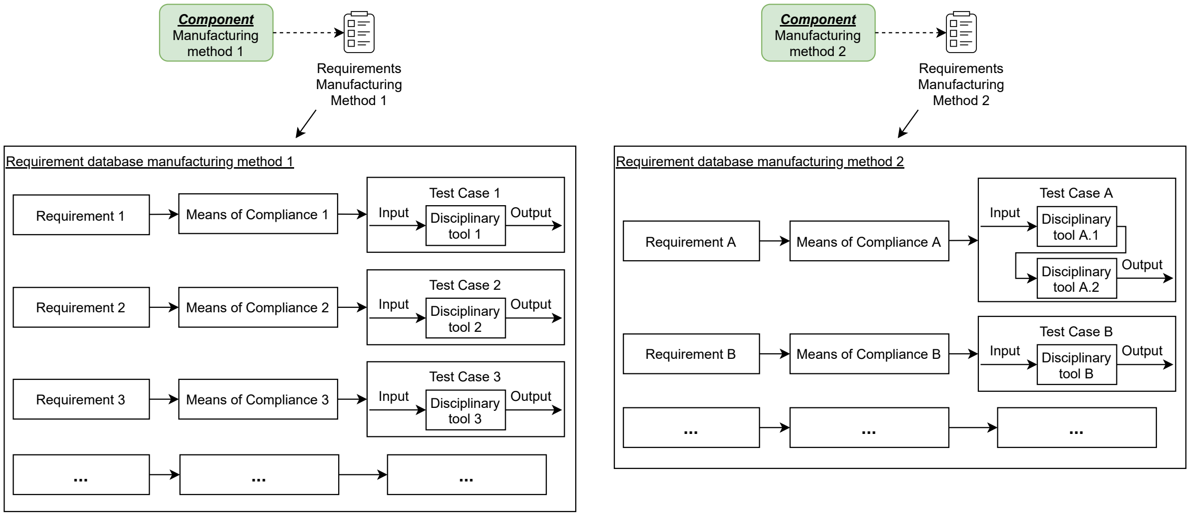


Fig. 6 Modelling of the requirement databases for each of the manufacturing methods. Each requirement in the database has a verification method consisting of a means of compliance and a test case

Once all the requirements have their verification methods assigned, the requirement compliance could be checked for a given design (Step V in Figure 5). However, the goal here is to include the requirements in the design process using MDAO workflows. As a requirement can be implemented in different ways in the MDAO process, a problem role needs to be assigned to each requirement. The possible problem roles are a constraint, objective, design variable (bound), parameter input, or quantity of interest. Knowing the problem role of each requirement and their verification method provides all the information required to formulate the MDAO workflow, which will be discussed in the next subsection. More details on the RVF can be found in Bruggeman et al. [9].

C. Formalizing and executing dynamic MDAO Workflows

Once the requirements have a verification method and a problem role assigned, the next step is to formulate the MDAO Workflow. As explained in subsection III.A, depending on the choice of production methods, different requirements may become active, thereby changing the design variables, disciplinary tools, and constraints to be addressed in the MDAO workflow. As a consequence, the evaluation of different manufacturing methods within one MDAO study requires the ability to formulate and execute dynamic workflows. Dynamic MDAO workflows are workflows that change based on the current design point. For example, when metal machining is chosen to manufacture a part, the design variables, disciplinary tools, and constraints relevant to machining become active. When composite hand lay-up or stamp forming is chosen, different variables, tools, and constraints become active.

To enable dynamic MDAO workflows, three new concepts are introduced in the workflow: subworkflows, switches and branches. A subworkflow is a workflow in a workflow. A top-level workflow is formulated, which contains an executable block that points towards a lower-level subworkflow. This subworkflow can consist of a single tool, a chain of tools, or even a full optimization workflow. A switch activates different branches of a workflow depending on the received input variables. A branch can either be a mathematical function, analysis, or subworkflow. Different conditions must be given for the branches and if a condition is true, the corresponding branch is executed.

An example of a subworkflow is given in Figure 7, while the integration of the subworkflow in the main workflow is given in Figure 8. Examples of switches and branches are also given in Figure 8. This XDSM[1] is based on the design problem modeled in Figure 4. In this example, Part A can be manufactured by two different manufacturing methods. Therefore, a switch is added for Part A, indicated in the XDSM with a blue diamond-shaped block. The input for the switch is in this case the manufacturing method. The outputs of the switch are the so-called "*decision variables*". Decision variables determine whether a branch of the switch must be executed and can therefore either be True or False.

---

[1] The XDSM presented here is a proposed extension to the standard XDSM visualization from Lambe and Martins [10]. New graphical elements are added for the switches and branches as required for the formalization of the dynamic workflows.

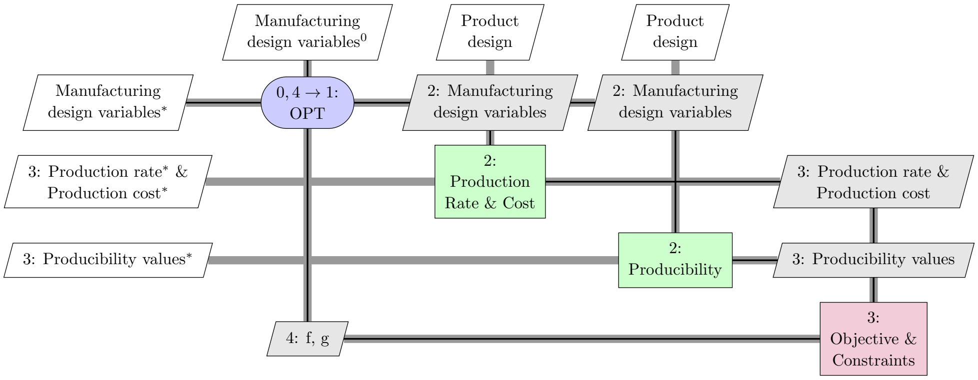


Fig. 7 XDSM of an MDAO workflow derived from the requirements of a given production process

The value of each decision variable is based on the condition specified for each branch and evaluated by the switch. For example, a condition for a branch could be "*Production method is machining*". In this case, the branch will be executed only if the variable production method is equal to machining. Note that the conditions must be specified in such a way that only one branch is executed per iteration.

The branches of the switch are indicated in the XDSM by the green boxes with a blue diamond in the upper left corner. In Figure 8, the branches represent subworkflows (visualized in Figure 7). Each subworkflow focuses on a different manufacturing method. To limit the size of the XDSM, a collapsed notation of the switch is introduced with stacked branch blocks (see Figure 9).

As explained in subsection III.B, the goal is to formulate the MDAO workflows based on the different requirement databases. Each manufacturing requirement database forms the basis for the formulation of a subworkflow as for example shown in Figure 7. This subworkflow focuses on the optimization of the manufacturing process by changing the manufacturing design variables (for example the feed rate for machining) or on the analysis of a design given a fixed manufacturing process. The manufacturing requirements are given an MDAO problem role, such as design variable or constraint and their verification methods determine which analyses need to be performed. Examples of analyses could be tools to evaluate the production rate, cost, or producibility of the design.

Two types of requirements are stored in the requirement databases for the manufacturing processes. The first type of requirements applies to the manufacturing process, for example, "*The machining process must have a maximum feed rate of less than xx millimeters per minute.*". The compliance of this type of requirement is influenced by the design and settings of the manufacturing process. These requirements can be implemented in the subworkflow as constraints or design variables. The second type of requirements applies to the product, for example, "*The product must have a maximum outer dimension of less than xx millimeters.*" in case the product needs to go into the autoclave. Compliance with these requirements is influenced by the design of the product. However, as the subworkflow focuses on the manufacturing process only, the product design is treated as a fixed input. Therefore, these requirements are evaluated in the manufacturing subworkflows as state variables (i.e., they get assigned the role of quantity of interest), and then passed to the main-level workflow, where they are treated as constraints. How this can be implemented will be shown in section IV.

The main or top-level workflow is formulated based on the roles and verification methods for the product requirements and the system architectural design space model. In the example of Figure 8, two switches are implemented, one for Part A and one for Joint A. This corresponds to the system architectural design space model in Figure 4. Note that Part B does not have a switch as only one manufacturing method can be used to produce this part.

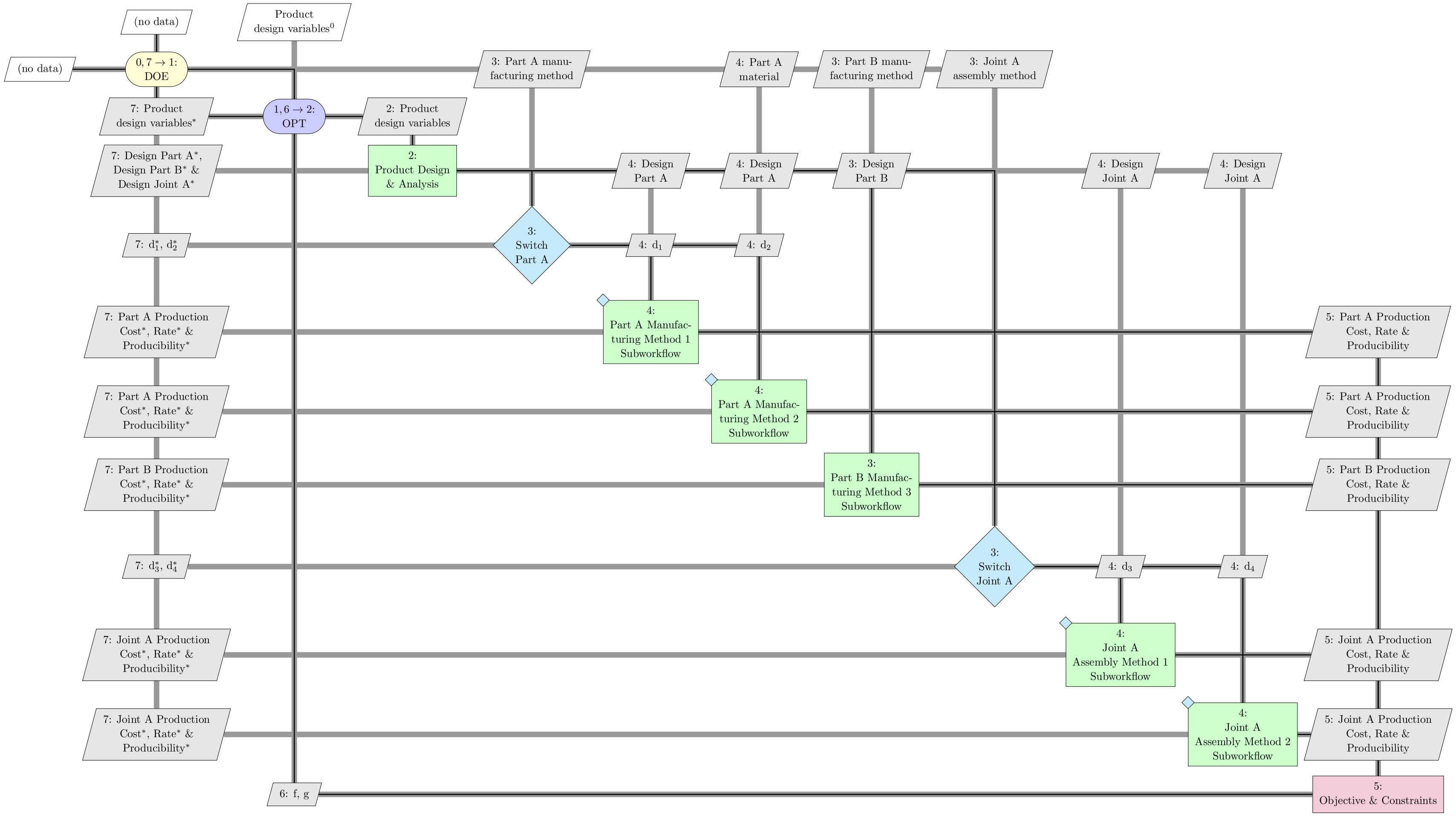


Fig. 8 XDSM formalization of a dynamic workflow for the exploration and optimization of the architectural design space modeled in Figure 4. Switches (blue diamond-shaped blocks) are introduced to select among alternative workflow branches. The branches of each switch (which in this case are all subworkflows) are indicated by green blocks with a small blue diamond in the upper left corner. In this XDSM, two switches are present, one for the manufacturing method of Part A and one for Joint A. For Part B, only one manufacturing method can be used, hence no switch is required.

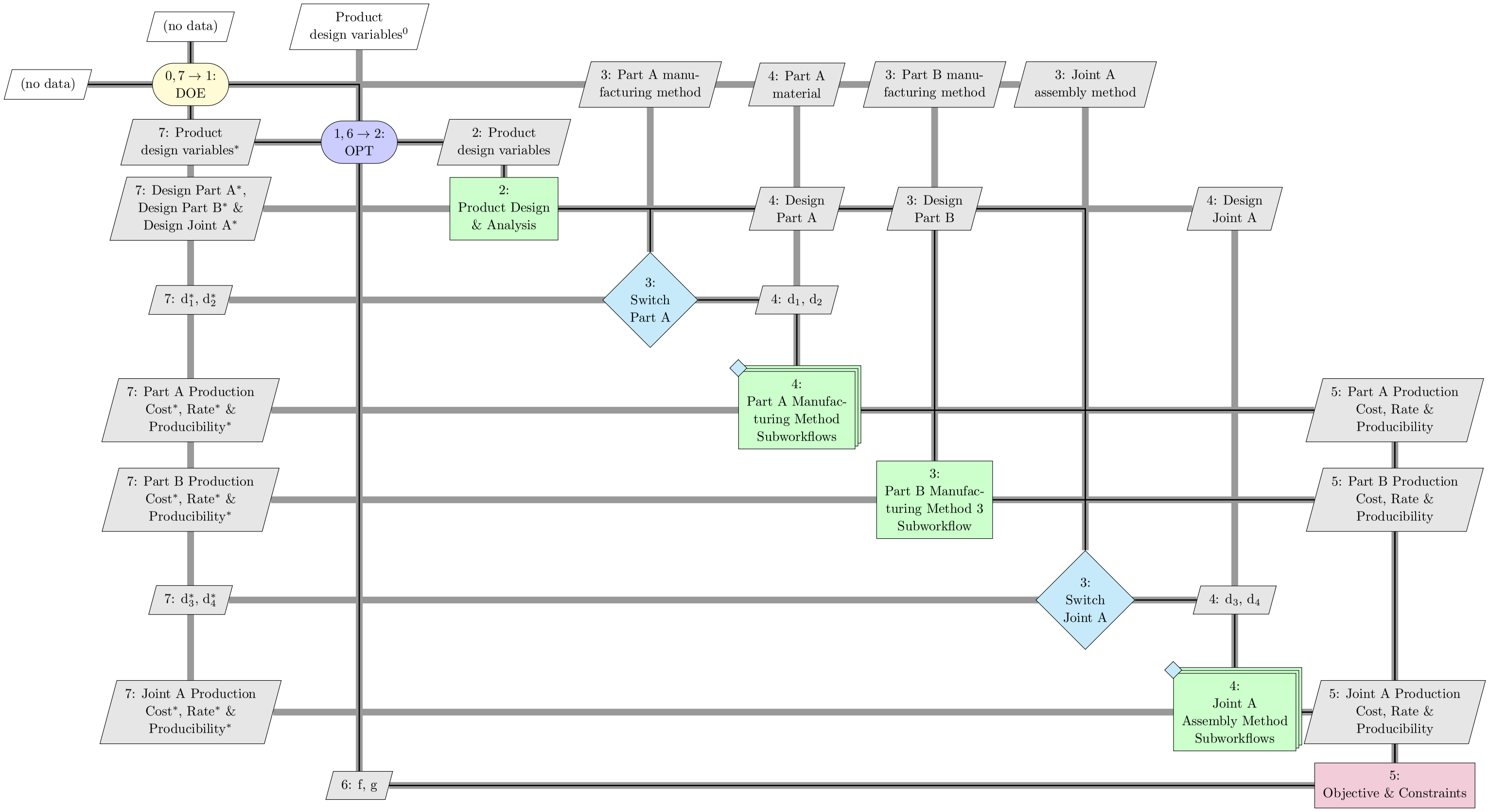


Fig. 9 Compact version of the XDSM from Figure 8, with the alternative branches of each switch shown as stacked blocks

In this example, a DoE of optimization is performed. The DoE varies the possible manufacturing and assembly options for parts and joints, while the optimizer tries to find the best product design for each combination of production methods. The result of each DoE iteration is an optimized product design together with its optimized production methods (as the production methods are optimized in the subworkflows). During the post-processing, the different optimized points of the Doe can be plotted in a Pareto front as shown in Figure 10. Note that the combination of Joint A-type 1 with Material 2 for Part is not present. This combination was marked unfeasible in the system architectural design space model and is therefore not evaluated in the MDAO workflow. Furthermore, as the MDAO workflow has been derived from the requirements, a requirement compliance report can be generated for each point in the Pareto front automatically, as described in Bruggeman et al. [9].

Once the MDAO problem is fully formulated, a neutral format like CMDOWS[2] [11] is used to transfer it to a Process Integration & Design Optimization (PIDO) tool, where the executable workflow is automatically materialized and executed.

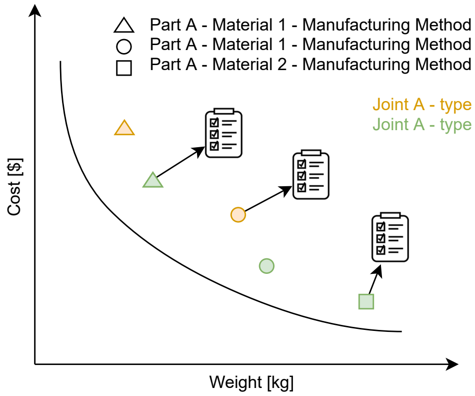
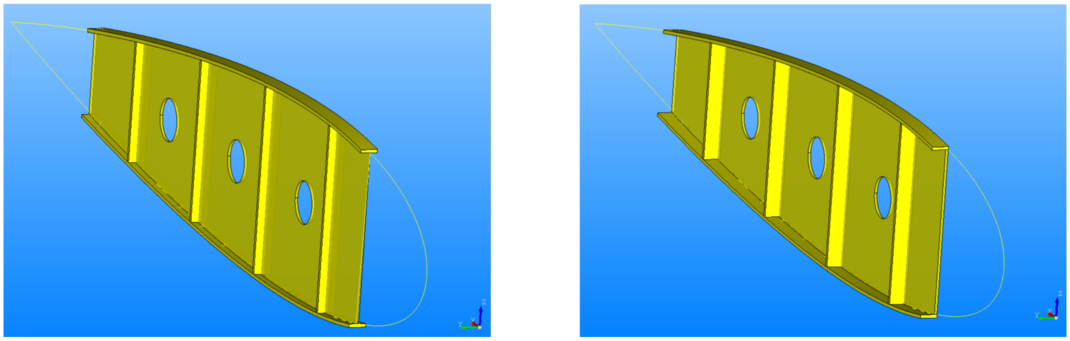


Fig. 10 Representative Pareto front for the fictitious DfMA problem formulated in Figures 8-9. Each point represents an optimized design for a specific combination of production processes. The colors indicate the used joint type and the symbols indicate the material and part manufacturing. For each point, a requirement compliance report can be generated automatically

IV. Use Case Description

The methodology described in the previous section is demonstrated here for the design and manufacturing of a wing rib. This structural component consists of a web with holes, two flanges, and stiffeners. Two manufacturing methods are considered, namely machining and stamp forming. When the rib is produced using machining, the starting point is a block of metal. Material is removed until the rib and its features (holes, stiffeners, etc.) are obtained. Material can be removed on both sides of the rib, so flanges can be present on both sides as shown in Figure 11a. When using stamp forming, the starting point is a sheet of composite which is stamped in the required shape. Therefore, the flange can only be present on one side of the web as shown in Figure 11b. The stiffeners are produced in a separate process and later bonded onto the web.



(a) Rib model to be produced with metal machining. Flanges are present on both sides of the web

(b) Rib model for composite stamp forming. Flanges are present on one side of the web only

Fig. 11 CAD models of a wing rib, consisting of a web with holes, two flanges, and stiffeners

As described in section III, the architectural design space model for the rib is created first (see Figure 12). The starting point is the boundary function "*Provide stability against panel buckling*". This function is fulfilled by the rib. The rib has the derived function "*Withstand internal loads*", which can be fulfilled by two different materials, Aluminum 2024-T4 (metal) or Carbon fiber LM-PAEK (composite). Each of these materials introduces a new derived

---

[2] Code repository: `https://gitlab.tudelft.nl/lr-fpp-mdo/cmdows`

function to create the rib shape from that specific material. For metal, the function is fulfilled using machining, while for composites, stamp forming is used. Note that only one part is created in this example and no assembly is included. Therefore, no incompatibility links are present.

From the architectural design space model, it can be concluded that three different requirement databases need to be created. One for the rib design, one for the machining process, and one for the stamp forming process. Table 1 presents the requirements for the rib design, together with their means of compliance and test cases. Requirements RR1001-1004 focus on geometrical features of the rib. For all four requirements, their compliance must be checked through geometrical inspection of the CAD model. This can automatically be checked using a KBE tool developed for this use case to generate the CAD models of the rib. Requirement RR1005 ensures the rib can carry the acting loads without structural failure. Within the rib tool, a structural analysis module has been implemented to calculate the maximum load and shear flow the rib can carry for a given fixed design. Therefore, the rib tool is also the test case for this requirement. Requirements RR1006-1007 put constraints on the cost and the weight of the rib. The cost of the rib is calculated using the open-source CATMAC (Cost Analysis Tool for Manufacturing of Aircraft Components) tool, developed by GKN-Fokker Aerostructures [12]. CATMAC estimates the process time of each manufacturing process based on part characteristics like size, material type, and thickness. By multiplying the process time by hourly labor and machine rates, the part costs are estimated. The weight of the rib is calculated using a weight estimation module in the rib tool. This module calculates the volume of the part and multiplies it with the density of the material. Lastly, requirement RR1008 focuses on the producibility of the rib and is explained later when introducing the requirements for the production methods.

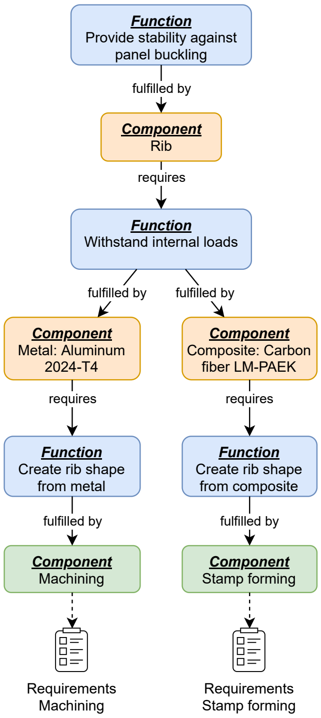


Fig. 12 Architectural design space model for the rib design, including two materials and manufacturing methods

Table 1 Rib design requirements

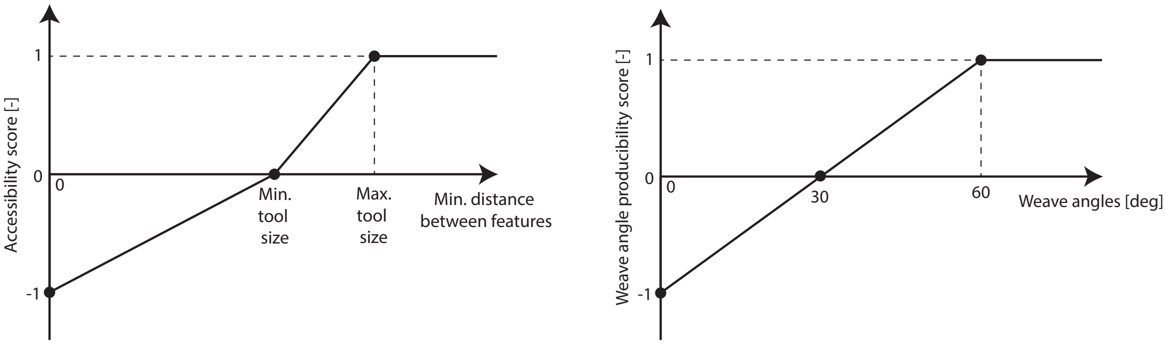
ID	Requirement text	Means of compliance	Disciplinary tools in test case	MDAO problem role
RR-1001	The rib must have a rib thickness between 2 and 10 mm.	Geometrical inspection of the CAD model	Rib tool	Design variable bound
RR-1002	The rib must have a stiffener spacing between 0 and 300 mm.	Geometrical inspection of the CAD model	Rib tool	Design variable bound
RR-1003	The rib must have 1 to 4 holes.	Geometrical inspection of the CAD model	Rib tool	Design variable
RR-1004	The rib must have hole radii between 10 and 40 mm.	Geometrical inspection of the CAD model	Rib tool	Design variable bound
RR-1005	The rib must have a critical buckling shear flow of at least 150 kN/m.	The structural analysis module of the rib tool has to be used	Rib tool	Constraint
RR-1006	The rib must have a maximum cost of 5000$.	A process-based cost estimation tool has to be used	CATMAC	Objective
RR-1007	The rib must have a maximum weight of 20 kg.	The weight calculation module of the rib tool has to be used	Rib tool	Objective
RR-1008	The rib must have a producibility score of at least 0	Physics-based producibility tools have to be used	Manufacturing producibility tools	Constraint

The requirement database for machining is shown in Table 2. In this example, only one requirement is formulated. This requirement focuses on the accessibility for the tooling during the machining process. If two features (e.g. two stiffeners) are too close to each other, the tool cannot remove the material in between these two features, and therefore the part is not producible.

Table 2 Machining requirements

ID	Requirement text	Means of compliance	Disciplinary tools in test case	MDAO problem role
RM-1001	The rib must have a machining accessibility score of at least 0	A physics-based producibility tool has to be used	Machining producibility tool	Quantity of interest

To measure how well the tool can reach the material that needs to be removed, an accessibility score is defined. This score ranges from -1 to 1, with a negative score meaning that the product is not producible, and a positive score meaning that it is producible. The relation between the accessibility score and the minimum distance between two features is shown in Figure 13a. A score with the value of -1 means that the minimal distance is zero and therefore the rib is not producible. A score value of 0 means that the smallest feature distance equals the smallest available tool size diameter. When the distance between features is increased, the possibility arises to use a bigger tool, which is beneficial as it can remove more material in less time. This increases the producibility and thus the accessibility score. The maximum score of 1 is assigned to parts where the smallest distance equals the largest available tool size. This means that the entire part can be produced using the largest tool.



(a) Plot of the accessibility score used for the machining process

(b) Plot of the weave angle producibility score used for the stamp forming process

Fig. 13 Visualizations of the producibility scores developed for the rib use case

The requirement database for the stamp forming process is given in Table 3. Just as for the machining process, only one requirement is formulated, focusing on the producibility of the rib. More specifically, the requirement focuses on the weave angles within the rib after deformation. For the stamp forming process, carbon-woven fabric is used. Before deformation, all angles between the fibers in the fabric are 90 degrees. However, during the forming process, the fabric gets deformed and thus the angles between the fibers change. These angles are called the weave angles.

Table 3 Stamp forming requirements

ID	Requirement text	Means of compliance	Disciplinary tools in test case	MDAO problem role
RS-1001	The rib must have a weave angle producibility score of at least 0	A physics-based producibility tool has to be used	Drape tool	Quantity of interest

The relation between the weave angles and the producibility score is given in Figure 13b. Weave angles below 30 degrees are generally not producible, hence a score value of 0 is given for parts whose smallest angles are 30 degrees. If the angles are lower, a negative producibility score is assigned. When the angles increase it becomes easier to manufacture the part. Angles above 60 degrees are always manufacturable. Therefore, a producibility score of 1 is assigned to parts with angles of at least 60 degrees. For angles between 30 and 60 degrees, a linear relation between 0 and 1 is used. To evaluate the weave angles in the rib model, the drape tool, an in-house developed tool based on the work of Bergsma [13], is used.

As stated in section III, there are two types of requirements in the manufacturing requirement databases. Requirements that focus on the production process and requirements that focus on the product. Both requirements in the machining and stamp forming databases are requirements that focus on the product (the rib). Compliance with these requirements can only be achieved by adjusting the rib design, which is however fixed within the subworkflows created from the manufacturing databases. Therefore, both requirements are assigned the MDAO problem role *quantity of interest*, as shown in Table 2 and 3, respectively. The resulting MDAO subworkflows for the machining and stamp forming process can be seen in Figure 14a and 14b, respectively. Both workflows consist of only one tool that evaluates the manufacturability. The inputs for these tools are the material and the CAD model of the rib design. The outputs are the different producibility scores.

(a) XDSM visualization of the machining subworkflow **(b) XDSM visualization of the stamp forming subworkflow**

Fig. 14 XDSM visualizations of the production subworkflows derived from the requirements in Table 2 and 3, respectively

From the rib design requirements, the top-level MDAO workflow can be formulated. In this example, it was chosen to implement requirements RR1001-1004 from Table 1 as design variable (bounds). The shear flow and producibility requirements are implemented as constraints, while the cost and weight are both assigned as objectives. This results in a multi-objective optimization problem. The resulting XDSM is shown in Figure 15. This XDSM can be automatically

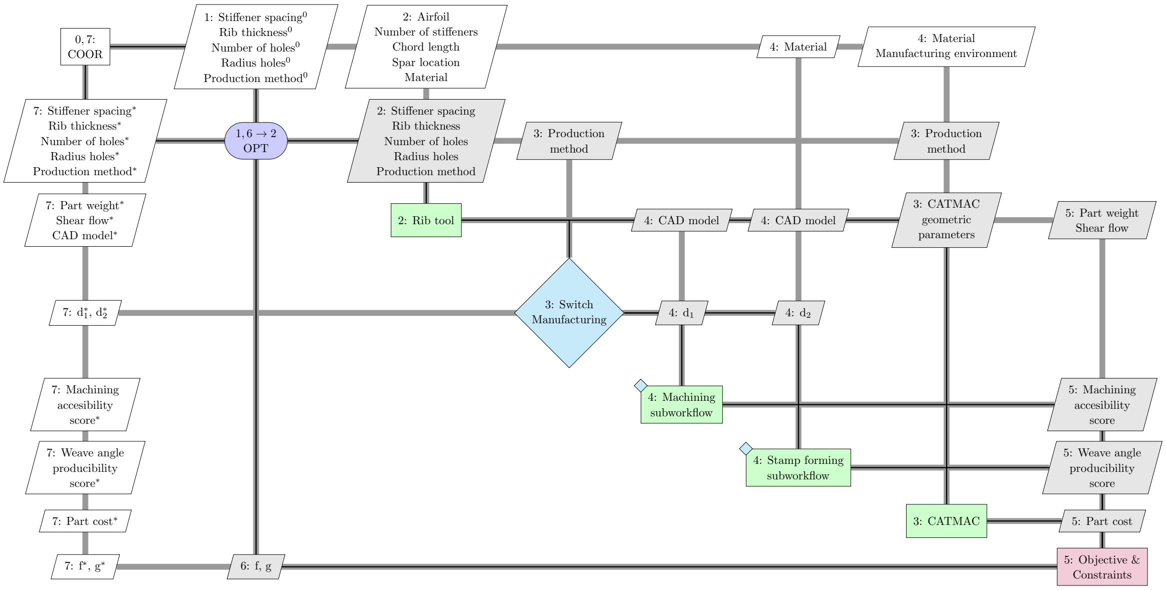


Fig. 15 XDSM of the rib optimization process including manufacturing considerations

formulated using software like for example KADMOS[3] (Knowledge- and graph-based Agile Design for Multidisciplinary Optimization System) [14].

Producibility requirement RR-1008 is implemented as a constraint, however, this requirement is closely linked to the two requirements in the manufacturing databases. Which variable to evaluate for this requirement depends on whether machining or stamp forming is used as manufacturing process. This is implemented in the main workflow by adding both the accessibility score and the weave angle producibility score as constraint. If one of the variables is not calculated (e.g. the accessibility score is not calculated because stamp forming was chosen), this variable gets a default value greater than zero such that the constraint is not violated. This way, the constraint is ’deactivated’ when it is irrelevant.

Once the full XDSM has been formulated, it can be translated into an executable workflow. In this example, the rib design case has been integrated into Optimus[4]. A new CMDOWS importer for Optimus is under development, such that the executable workflow can be generated automatically. The Optimus workflow for the main workflow is shown Figure 16, while the Optimus workflows for the manufacturing subworkflows are shown in Figure 17.

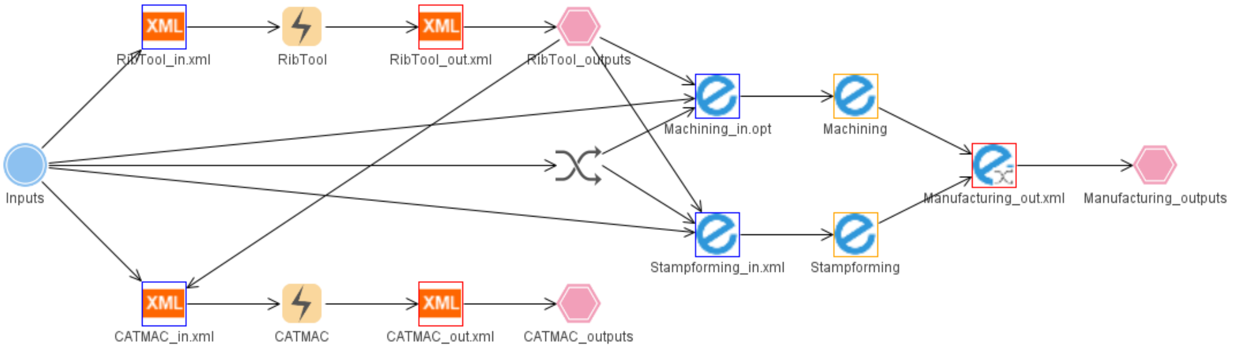


Fig. 16 Executable workflow of the rib optimization process including manufacturing considerations as implemented in Optimus

(a) Machining subworkflow **(b) Stamp forming subworkflow**

Fig. 17 Manufacturing subworkflows as implemented in Optimus

As explained in section III, three new elements were introduced in the XDSM to enable the dynamic workflows: subworkflows, switches, and branches. These elements also needed to be implemented in Optimus to execute the dynamic workflows. The subworkflows already existed in Optimus through OptInOpts. An OptInOpt is a pointer towards a different Optimus workflow. It is represented by the blue e-icons (see Figure 16). The OptInOpt provides the correct inputs to the subworkflow (blue e-icon with a blue frame), executes the subworkflow (blue e with a yellow frame), and extracts the output values (blue e with a red frame). Besides the subworkflows, also the switch and branches already existed in Optimus. Within the Optimus switch, one can set different conditions for the execution of each branch. The conditions for the branches are evaluated one by one. As soon as one condition equals True, that branch is executed. The evaluation of the other conditions is skipped as only one branch can be executed. Although subworkflows and switches were features already present in Optimus, their combination was not supported yet. Therefore, a customized output UCI (User Customized Interface; blue e with red frame) was developed in this research work. This output UCI takes the output from the subworkflow that has been executed and provides it to the main workflow.

Once the Optimus workflow has been created, it can be executed. The results for the rib design can be seen in Figure 18. This figure shows the Pareto front for the recurring cost and weight for different rib designs. The dynamic workflow is a new concept. To check whether the dynamic workflow can find the entire Pareto front, the results from the dynamic workflow are compared with the results from the two corresponding static workflows (workflows without a

---

[3] Code repository: `https://gitlab.tudelft.nl/lr-fpp-mdo/kadmos`

[4] `https://www.noesissolutions.com/our-products/optimus`, accessed on: 30-11-2023

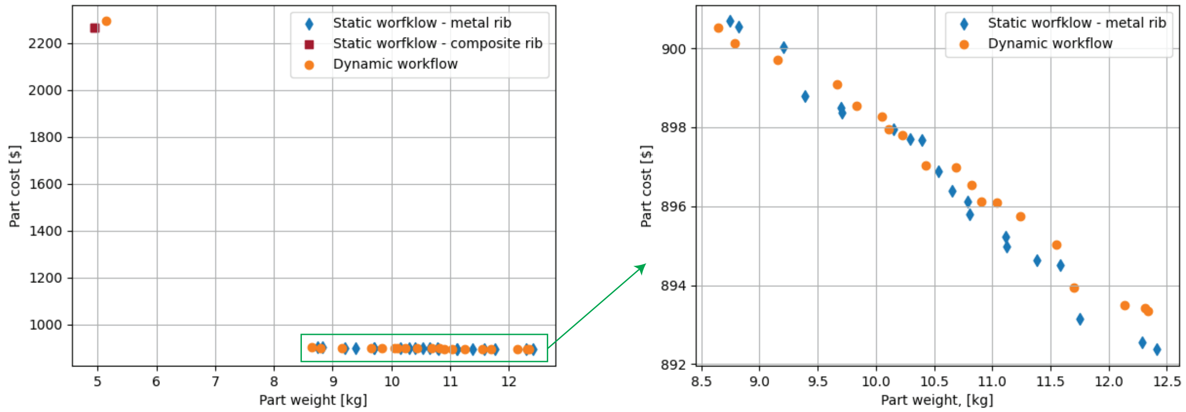


Fig. 18 Pareto front of the part weight versus recurring cost for the rib design. Both the Pareto front of the dynamic workflows as well as the equivalent static workflows are plotted

switch). One static workflow optimizes the rib design using metal machining, while the other optimizes the rib using composite stamp forming.

From the results in Figure 18, it can be seen that the lightest rib is made out of composite, however, this comes at a big cost compared to the metal ribs. The metal ribs are much heavier than the composite ones, but also much cheaper. Note that the results are preliminary results. Due to time limitations, not all cost elements for the rib are considered (e.g. the bonding of the stiffeners to the web for stamp forming is missing). However, the main goal of this plot is not to give the most accurate results for the rib design but to show a proof of concept of the dynamic workflows.

Figure 18 shows that the dynamic workflow has successfully found both the optimum point for composite as well as the Pareto front for metal. When machining a rib, one starts from a block of metal and material is removed from the block until the required shape is obtained. Therefore, a lighter rib is more expensive, as more material needs to be machined away. This results in a Pareto front for machining as is shown on the right-hand side of Figure 18. When the rib is stamp formed, different plies are built up until the correct thickness for the rib is obtained. This means that the heavier the rib (= more plies), the more the rib costs. As both objectives (minimizing cost and minimizing weight) are aligned, only one optimum point is found for the composite rib instead of a Pareto front.

The main benefit of using dynamic workflows is that only one workflow is now required to solve problems that in the past could only be solved by formulating and executing multiple workflows. Table 4 compares the different run times between the dynamic workflow and the static workflows for the rib design case. The dynamic workflow and static workflow for the metal rib were solved using a multi-objective evolutionary optimization algorithm[5], while the static workflow for the composite rib was solved using a single-objective evolutionary algorithm[6]. The reason for this is that the two objectives for the composite rib are aligned and can therefore be solved in a single-objective optimization.

Table 4 Comparison of execution time and number of iterations for the dynamic workflows and the static workflows when solving the rib design case

	Dynamic workflow	**Static workflow - metal rib**	**Static workflow - composite rib**	**Static runs combined**
Total run time [s]	3414	1336	2640	3967
Number of iterations [-]	140	80	141	-
Average iteration time [s]	24.4	16.7	18.7	-
Run time difference (Compared to dynamic run)	-	-60.9%	-22.7%	16.2%

---

[5] Optimus2021.1SP2, NSEA+ algorithm
[6] Optimus2021.1SP2, NAVIRUN algorithm

The execution times presented in Table 4 show that the implementation of the subworkflows in the dynamic workflows causes some overhead as the average time for one iteration is a bit higher than for the static workflows. However, fewer iterations needed to be performed compared to the combined static runs, resulting in a lower execution time for the dynamic workflow.

V. Conclusions & Recommendations

Many different design options and production methods need to be traded during the early design stages of airframe components. Including the analysis of the production process at the beginning of the design process leads to an earlier identification of production issues and can lead to a reduction in production complexity, saving both time and cost. However, trading different production methods is challenging as each production method puts different requirements on the product design. Furthermore, different design variables and tools are required to analyze or optimize each production process.

This paper presented a new methodology that enables the inclusion of production in the design process. It proposed a new concept to include production considerations in the architectural design space model, making the possible design choices for material, manufacturing, and assembly explicit. Each production method gets assigned a requirement database, in which the requirements are modeled together with their verification methods. By assigning different MDAO problem roles to each requirement, an MDAO workflow is formulated that is fully based on both the product as well as the production requirements. Due to the development of dynamic MDAO workflows, the relevant design variables, analysis tools, and constraints are automatically activated based on the current design point. Finally, incompatibility links ensure that infeasible combinations of manufacturing and assembly are filtered out.

The presented methodology was demonstrated for the design and manufacturing of a wing rib. The dynamic workflow successfully found the Pareto front for both composite stamp forming as well as metal machining. As expected, the composite rib was lighter than the metal rib but also more expensive. With the new methodology, a clear trace from requirements to MDAO workflow was created. Due to the introduction of the newly developed dynamic workflows, only one workflow needed to be formalized and executed, while previously a separate workflow was required for each combination of design and production method. Even though the dynamic workflows do introduce some extra overhead, in this example, it was still faster than the execution time of the corresponding static workflows combined. Using dynamic MDAO workflows will be especially beneficial when more production methods need to be traded, as the number of individual static workflows would make the problem intractable. Overall, the methodology enables the trade-off and optimization of different production methods while ensuring the design still complies with the production-specific requirements.

Within this paper, only a simple rib with two manufacturing methods was investigated. In the future, it would be interesting to see whether the methodology is scalable when more parts, manufacturing methods, and assembly are included. Furthermore, more research needs to be performed on the benefits and drawbacks of dynamic MDAO workflows.

Acknowledgments

The research presented in this paper has been performed in the framework of the AGILE 4.0 (Towards Cyber-physical Collaborative Aircraft Development) and DEFAINE (Design Exploration Framework based on AI for froNt-loaded Engineering) projects and has received funding from the European Union programs Horizon 2020 (grant agreement n° 815122) and ITEA 3 Call 6 project 19009. Furthermore, the authors would like to thank Ton van der Laan (GKN Fokker Aerostructures) for his ideas and feedback on including production in the system architectural design space model.

References

[1] Vianello, G., and Ahmed-Kristensen, S., "A comparative study of changes across the lifecycle of complex products in a variant and a customised industry," *Journal of Engineering design*, Vol. 23, No. 2, 2012, pp. 99–117.

[2] Dostaler, I., "Avoiding rework in product design: evidence from the aerospace industry," *International journal of Quality & Reliability Management*, Vol. 27, No. 1, 2010, pp. 5–26.

[3] Polacsek, T., Roussel, S., Bouissiere, F., Cuiller, C., Dereux, P., and Kersuzan, S., "Towards thinking Manufacturing and Design Together: An Aeronautical Case Study," *International Conference on Conceptual Modeling*, Springer, Cham, 2017, pp. 340–353.

[4] Bajaj, M., Peak, R., Wilson, M., Kim, I., Thurman, T., Jothishankar, M., Benda, M., Ferreira, P., and Stori, J., "Towards Next-Generation Design-for-Manufacturability (DFM) Frameworks for Electronics Product Realization," *IEEE/CPMT/SEMI 28th International Electronics Manufacturing Technology Symposium*, 2003.

[5] Ferrer, I., Rios, J., Ciurana, J., and Carcia-Romeu, M., "Methodology for capturing and formalizing DFM Knowledge," *Robotics and Computer-Integrated Manufacturing*, Vol. 26, 2010, pp. 420–429.

[6] van Dijk, R. E. C., d'Ippolito, R., Tosi, G., and La Rocca, G., "Multidisciplinary Design and Optimization of a Plastic Injection Mold using an Integrated Design and Engineering Environment," *NAFEMS World Congress 2011*, 2011.

[7] van der Laan, A. H., "Knowledge based engineering support for aircraft component design," Ph.D. thesis, Delft University of Technology, 2008.

[8] Bruggeman, A. M. R. M., and La Rocca, G., "From Requirements to Product: an MBSE Approach for the Digitalization of the Aircraft Design Process," *33rd Annual INCOSE International Symposium*, 2023.

[9] Bruggeman, A. M. R. M., van Manen, B., van der Laan, T., van den Berg, T., and La Rocca, G., "An MBSE-Based Requirement Verification Framework to support the MDAO Process," *AIAA Aviation 2022 Forum*, 2022.

[10] Lambe, A. B., and Martins, J. R. R. A., "Extensions to the design structure matrix for the description of multidisciplinary design, analysis, and optimization processes," *Structural and Multidisciplinary Optimization*, Vol. 46, 2012, pp. 273–284.

[11] Van Gent, I., La Rocca, G., and Hoogreef, M. F. M., "CMDOWS: a proposed new standard to store and exchange MDO systems," *CEAS Aeronautical Journal*, Vol. 9, No. 4, 2018, pp. 607–627.

[12] van der Laan, T., Johman, A., van Puffelen, T., Nolet, S., van Maanen, B., Daugulis, E., and van den Berg, T., "An open source part cost estimation tool for MDO purposes," *AIAA Aviation Forum*, 2021.

[13] Bergsma, O. K., "Three Dimensional Simulation of Fabric Draping," Ph.D. thesis, Delft University of Technology, 1995.

[14] Van Gent, I., and La Rocca, G., "Formulation and integration of MDAO systems for collaborative design: A graph-based methodological approach," *Aerospace Science and Technology*, Vol. 90, 2019, pp. 410–433.