33rd Annual INCOSE international symposium
hybrid event
Honolulu, HI, USA
July 15 - 20, 2023

Function-Based Architecture Optimization: An Application to Hybrid-Electric Propulsion Systems

Jasper Bussemaker
German Aerospace Center (DLR)
Hein-Saß-Weg 22, 21129 Hamburg, Germany
jasper.bussemaker@dlr.de

Raúl García Sánchez
German Aerospace Center (DLR)
Hein-Saß-Weg 22, 21129 Hamburg, Germany
raul.garciasanchez@dlr.de

Mahmoud Fouda
German Aerospace Center (DLR)
Hein-Saß-Weg 22, 21129 Hamburg, Germany
mahmoud.fouda@dlr.de

Luca Boggero
German Aerospace Center (DLR)
Hein-Saß-Weg 22, 21129 Hamburg, Germany
luca.boggero@dlr.de

Björn Nagel
German Aerospace Center (DLR)
Hein-Saß-Weg 22, 21129 Hamburg, Germany
bjoern.nagel@dlr.de

Abstract. Decisions arising in architecting processes greatly impact the success of the final product, however are subject to high uncertainty and large combinatorial design spaces. Selecting the best architecture for the problem at hand can be supported by architecture optimization techniques. In this paper, we show how architecture optimization can used for designing complex aeronautical systems, with the design of hybrid-electric aircraft propulsion systems as an application case. The function-based architecture optimization problem is formulated using an Architecture Design Space Graph (ADSG) created with the ADORE tool. Automatically generated architecture alternatives are evaluated using a multidisciplinary analysis framework coupling an overall aircraft design tool to mission and propulsion system simulation code. The multidisciplinary analysis toolchain is rebuilt for each architecture, automatically including and coupling selecting components. Architectures are optimized for three objectives using a multi-objective genetic algorithm. It is demonstrated that the large architecture design space can be effectively searched and a Pareto front can be obtained.

1. Introduction

Electric and hybrid-electric propulsion is becoming a feasible option for general aviation and commuter aircraft, opening the door towards a growing field of aviation with the opportunity of zero-emission flight (McDonald et al. 2021). The architecture of the propulsion system plays an important role in the design of (hybrid-)electric aircraft: novel combinations of various electrical, mechanical and thermal components enable the creation of new architectures which must be investigated at the early stage of the design process due to their large impact on the performance of the final product (Biser et al. 2020). Due to the lack of knowledge about the behavior of the system at such early stage, the specific impacts of architectural decisions on the overall aircraft design suffer from a high degree of uncertainty. In addition, the close coupling between the different disciplinary domains, such as aerodynamics, propulsion, and aircraft sizing, makes it difficult for designers to be able to evaluate

the influences of architecting decisions (Palaia et al. 2021). Furthermore, the architecture design space can be extremely large due to a combinatorial explosion of alternatives (Iacobucci 2012). This lack of knowledge and the existence of a large design space often lead to reliance on expert judgement to support taking decisions, which might suffer from bias, subjectivity, conservatism or overconfidence (Roelofs & Vos 2018). To increase the level of certainty of the impact of architectural decisions on performance, and reduce reliance on expert judgement, it is necessary to employ physics-based systematic design space exploration techniques in search for novel architectures for hybrid electric propulsion systems (Bussemaker & Ciampa 2022). These techniques involve the explicit definition and tracking of architectural choices, and rely on the automatic generation and quantitative evaluation of architecture candidates through optimization to find the best architecture, or Pareto-front of architectures, within the design space.

This paper demonstrates the application of **architecture optimization** techniques to the design of hybrid-electric propulsion systems. The system architecting process is connected to the multidisciplinary propulsion system performance analysis code, developed by the authors in (Fouda et al. 2022), to enable systematic design space exploration. The rest of the paper starts with discussion about system architecting and architecture optimization. Then, architecture design space modeling, and modular propulsion system architecture analysis methodologies are introduced. A description of the implementation of the method and descriptions of the tools used are provided after that. Finally, the methodology is demonstrated using a hybrid-electric propulsion architecture optimization study of a short-range commuter aircraft.

The work presented in this paper has been performed as part of the EU-funded AGILE4.0 project, wherein the German Aerospace Center (DLR) lead the development of a Model-Based Systems Engineering (MBSE) framework for complex system design (Bussemaker, Boggero & Nagel 2023; Ciampa, Rocca & Nagel 2020). The framework consists of two consecutive and connected phases, an upstream architecting phase, and a downstream product design phase (Ciampa & Nagel 2021). The upstream phase is executed within an MBSE context, dealing with stakeholders needs, system requirements, scenarios, and architecture design. The downstream phase is executed within a Multidisciplinary Analysis and Optimization (MDAO) context. MDAO deals with solving computational systems coupling multiple disciplinary analysis tools, each dealing with one aspect of the system to be designed (for example: designing an aircraft engine, analyzing aerodynamic performance, or estimating costs) (Sobieszczanski-Sobieski, Morris & Tooren 2015), possibly operated by different experts, departments, or organizations (Ciampa & Nagel 2020). The application of MDAO allows finding the optimal balance between these disciplines with often tightly-coupled and opposing design goals. By bridging the gap between the two phases, the best architecture(s) can be identified by applying formal design optimization techniques to the architecture design space.

1.1. System Architecture Optimization

In systems engineering, the product to be designed is treated as a system; a set of interconnected components that work together to produce some benefit to its users (NASA 2016). To avoid bias towards familiar solutions when designing a system, it is necessary to define what the system does (its function) before how the system does it (its form) (Hirtz et al. 2002). The system architecting process can then be seen as a decision-making process where the allocation of function to form, and the selection of relationships among elements of form (i.e. system components) together define a design space of architecture instances: the architecture design space (Crawley, Cameron & Selva 2015). A systematic approach towards architecting needs to start at the system boundary function, which defines the purpose of existence of the system (Mavris, Tenorio & Armstrong 2008). Then possible design decisions are identified and methodically explored in pursuit of the best possible combination of decisions to reach an optimal architecture to achieve design objective(s) (Judt & Lawson 2016).

Architecture optimization enables the unbiased automated search for the best architecture by defining the design problem as a numerical optimization problem. An optimization algorithm is used to vary a set of design variables (the design vector) and observe their impact on one or more merit functions (objective functions) with the goal of finding the design vector that minimizes (or maximizes) the values of these functions without violating any constraints (Simmons & Crawley 2008). In architecture optimization, design variables represent architecture decisions (Bussemaker, Ciampa & Nagel 2020). Design variables can be either discrete or continuous. Discrete design variables can be either integer, such as the number of propellers, or categorical, as the choice between using a turboshaft engine or an electric motor (Saves et al. 2022). Examples of a continuous design variables are component sizing parameters such as the propeller diameter or the rated power of the electric motor. Another aspect that arises architecture optimization is decision hierarchy: some architecting decisions can be conditionally active based on other decisions (Pelamatti et al. 2020). For example, the type of electric power source only needs to be chosen if an electric motor is selected to provide shaft power in a propulsion system. If, however, a turboshaft engine is selected, the electric power source choice would be inactive.

Several factors contribute to the difficulty of solving architecture optimization problems. One aspect is that there is no a-priori information about the behavior and shape of the objective and constraint functions, and they might take significant time to calculate. Therefore, the performance analysis model is considered as an expensive black-box function (Jones, Schonlau & Welch 1998). Additionally, due to conflicting stakeholder needs, the system might need to be optimized for multiple conflicting objectives. This means there is no one optimal architecture, but rather a set of Pareto-optimal architectures where some architectures are better than others for some objectives, but worse than others for other objectives (Rudenko & Schoenauer 2004). The combination of the previously discussed aspects makes the architecture optimization problem a **mixed-discrete, multi-objective, hierarchical problem** (Bussemaker, Ciampa & Nagel 2020). This class of problems is challenging for existing optimization algorithms as they have significant difficulties with either finding a set of optimal points at all, or finding an approximation of the Pareto-set without needing excessive function evaluations (Bussemaker, Bartoli, et al. 2021).

2. Methodology

In this section the methodology used to implement architecture design space definition, architecture optimization, and hybrid-electric propulsion system architecture performance evaluation is explained in more details. First, a description of how to formally describe the architecture design space is provided. Then, a flexible simulation approach that allows modular definition of a propulsion architecture and its aircraft-level evaluation is presented.

2.1. Function-Based Architecture Design Space Modeling and Evaluation

The design space is modeled using the **Architecture Design Space Graph (ADSG)** (Bussemaker, Ciampa & Nagel 2020): a graph-based formulation mapping functions to components and additionally representing component characterization (instantiation and selection of attribute values) and connection choices. The ADSG is a directed graph where edges between nodes represent derivation relationships: if the source node is contained in an architecture instance, the target node is too. Architectural decisions are represented by *selection-choice nodes*, where target nodes represent mutually-exclusive options. For example, allocating more than one component to a function represents a selection-choice, as each function is fulfilled by only one component in an architecture instance. Component connection choices are modeled using connection-choice nodes, where source and target nodes represent connection sources and targets, respectively. If not possible using selection-choices, technology compatibility constraints can also be modeled using incompatibility edges: connections between nodes that may never appear in an architecture instance together.

Performance metrics (including objectives and constraints) and additional evaluation inputs are described using Quantities of Interest (QOIs). For more information about the ADSG, the reader is referred to (Bussemaker & Ciampa 2022).

The architecture design space is modeled using ADORE: a Python implementation of the ADSG with a web-based user interface for editing the ADSG, creating architecture instances from the ADSG, defining numerical optimization problems, and interfacing with optimization frameworks for driving the design space exploration (Bussemaker, Boggero & Ciampa 2022). fig. 2.1 shows the interactive editing canvas.

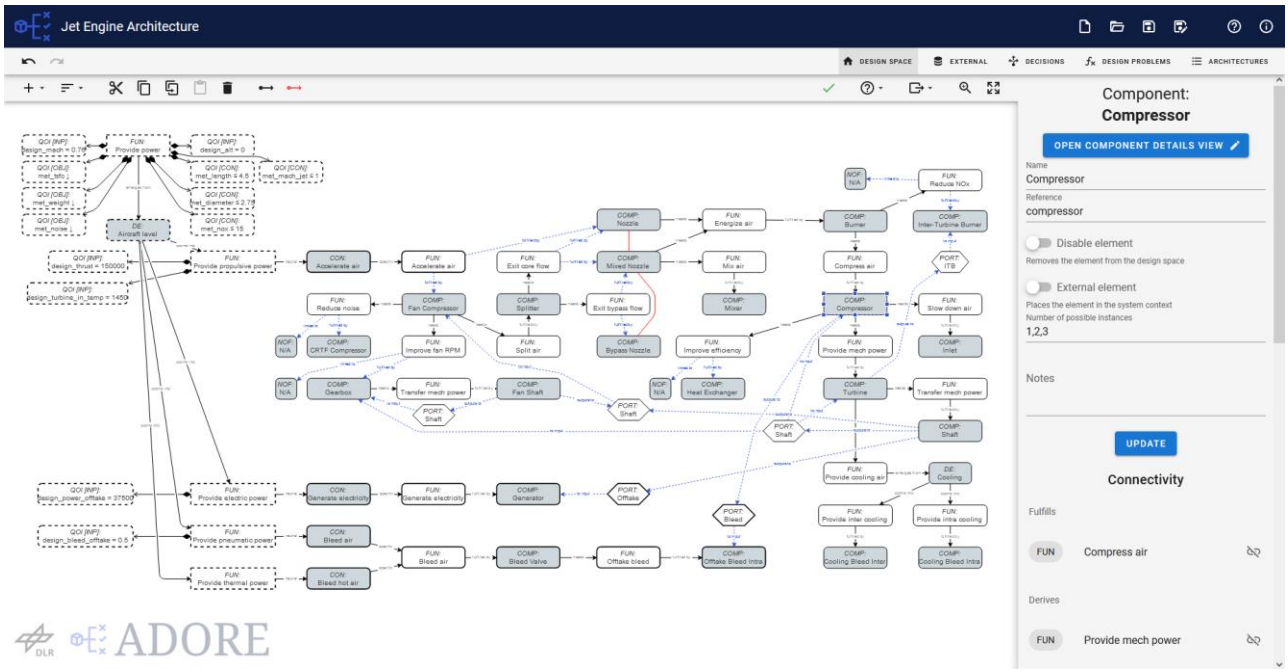

Figure 2.1: ADORE web-based graphical user interface showing the design space editing canvas. Figure reproduced from (Bussemaker, Boggero & Ciampa 2022).

Architecture evaluation, i.e. calculating performance metrics of a generated architecture instance, is not implemented in ADORE itself, however several mechanisms for connecting to evaluation environments are available (Bussemaker & Boggero 2022; Bussemaker, Boggero & Ciampa 2022). One Python-based method is the **Class Factory Evaluator (CFE)**: an object-oriented approach for instantiating classes based on element occurrence in architecture instances. Class properties can be filled from Quantities of Interest (QOIs) or component connections (represented by references to other class instances). Compared to directly parsing ADORE's data model, the CFE offers a more intuitive and flexible approach for implementing Python-based architecture evaluation.

The architecture optimization loop using CFE is presented in fig. 2.2. At each iteration, the optimization algorithm suggests a design vector x, which is then converted to an architecture instance by ADORE. As some design variables might be inactive due to design variable hierarchy, the design vector is corrected to reflect this: the imputed design vector x_{imp} and associated activeness vector δ, specifying which design variable is active in the design vector, are returned back to the optimizer (Bussemaker, Bartoli, et al. 2021). The defined class factories then use the architecture instance to instantiate objects, which are used to define the simulation input. The optimization loop is finally closed by extracting performance metrics from the simulation output, which are integrated in the architecture instance model and communicated back to the optimizer.

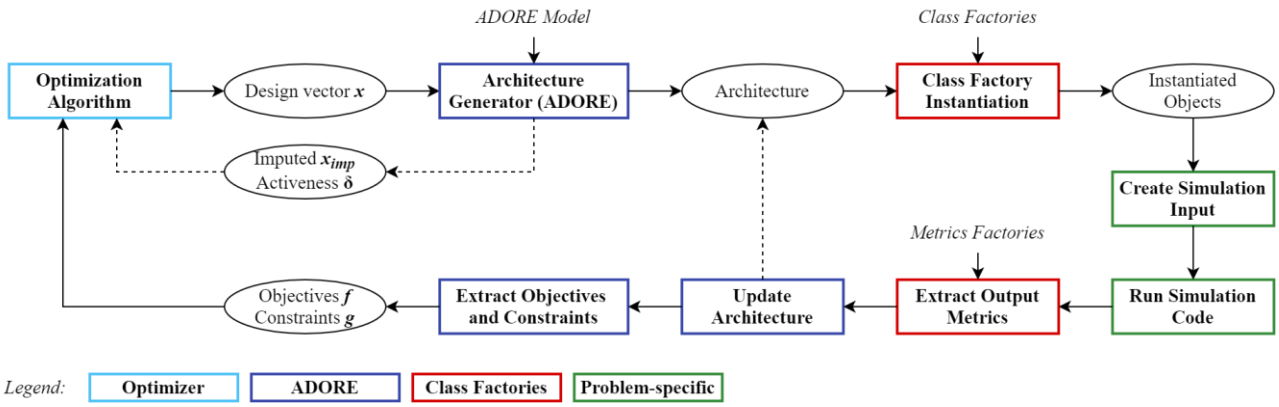

Figure 2.2: Architecture optimization loop with the Class Factory Evaluation (CFE) approach. Figure adopted from (Bussemaker & Boggero 2022).

2.2. Hybrid-Electric Propulsion System Modeling and Evaluation

To apply architecture optimization to the design of some system of interest, in this case the design of hybrid electric propulsion architectures, several aspects should be considered. First, all possible architecture choices must be explicitly defined by constructing a formal definition of the considered architecture design space. In addition, a systematic approach for the design space exploration is needed. Lastly, flexible, modular and efficient simulation environments are required to quantitatively evaluate architectures in order to compare them to each other (Bussemaker, Ciampa & Nagel 2021). Brelje et al. have performed an extensive recent literature study in (Brelje & Martins 2018; Brelje & Martins 2019) to investigate the capabilities of existing simulation environments to address different sub-problems with regard to electric and hybrid electric aircraft design. Existing design frameworks have focused on the detailed-analysis part of the design problem, such as aerodynamic analysis, electric components modeling, and thermal management, which are crucial for architecture performance evaluation. An analysis of a turbo-electric architecture coupled with distributed propulsion is presented in (Vries, Brown & Vos 2018). A procedure for sizing series hybrid electric propulsion architectures with energy management strategy is described in (Finger et al. 2020). A multidisciplinary optimization study for the hybridization of single-aisle tube- and-wing configuration aircraft with distributed electric fans is introduced in (Sgueglia et al. 2020), and a hybridization strategy for sizing non-conventional configurations in provided in (Palaia et al. 2021). Palladino et al. (Palladino et al. 2020, 2023) integrated hybrid-electric propulsion sizing for several system architectures in an overall aircraft design platform. Sahoo et al. (Sahoo, Zhao & Kyprianidis 2020) present an overview of existing hybrid-electric and electric propulsion concepts, and highlight the tightly-coupled nature of the aircraft sizing problem. They compare different propulsion system sizing methods, all of which are only applied for predefined system architectures.

In this work, we use the modular and flexible hybrid-electric propulsion system architecture builder and evaluation framework introduced by Fouda et al. (Fouda et al. 2022) to enable dynamic formulation of the propulsion system numerical evaluation model. The evaluation framework is based on OpenConcept (Brelje & Martins 2018), an open-source mission analysis and propulsion system modeling framework. OpenConcept is a set of component models and mission integration routines programmed using NASA's OpenMDAO (Gray et al. 2019) framework, an open-source Multidisciplinary Design Analysis and Optimization (MDAO) framework with system-level gradients calculation, enabling rapid convergence through the use of gradient-based solvers and optimizers.

The architecture builder enables the definition of the propulsion system architecture using class instances (combined in a *PropSysArch* class instance), which are then used to construct the numerical OpenMDAO model using OpenConcept library items. Each architecture consists of thrust,

mechanical power, and electrical power generation elements. Thrust generation elements convert mechanical power into thrust; mechanical power is generated from fuel or electrical power; and electrical power is generated from batteries and/or generators, and are only included if electrical power is needed for mechanical power generation. The architecture builder logic is shown in fig. 2.3. For a more detailed description of available classes and underlying numerical models, the reader is referred to (Fouda et al. 2022).

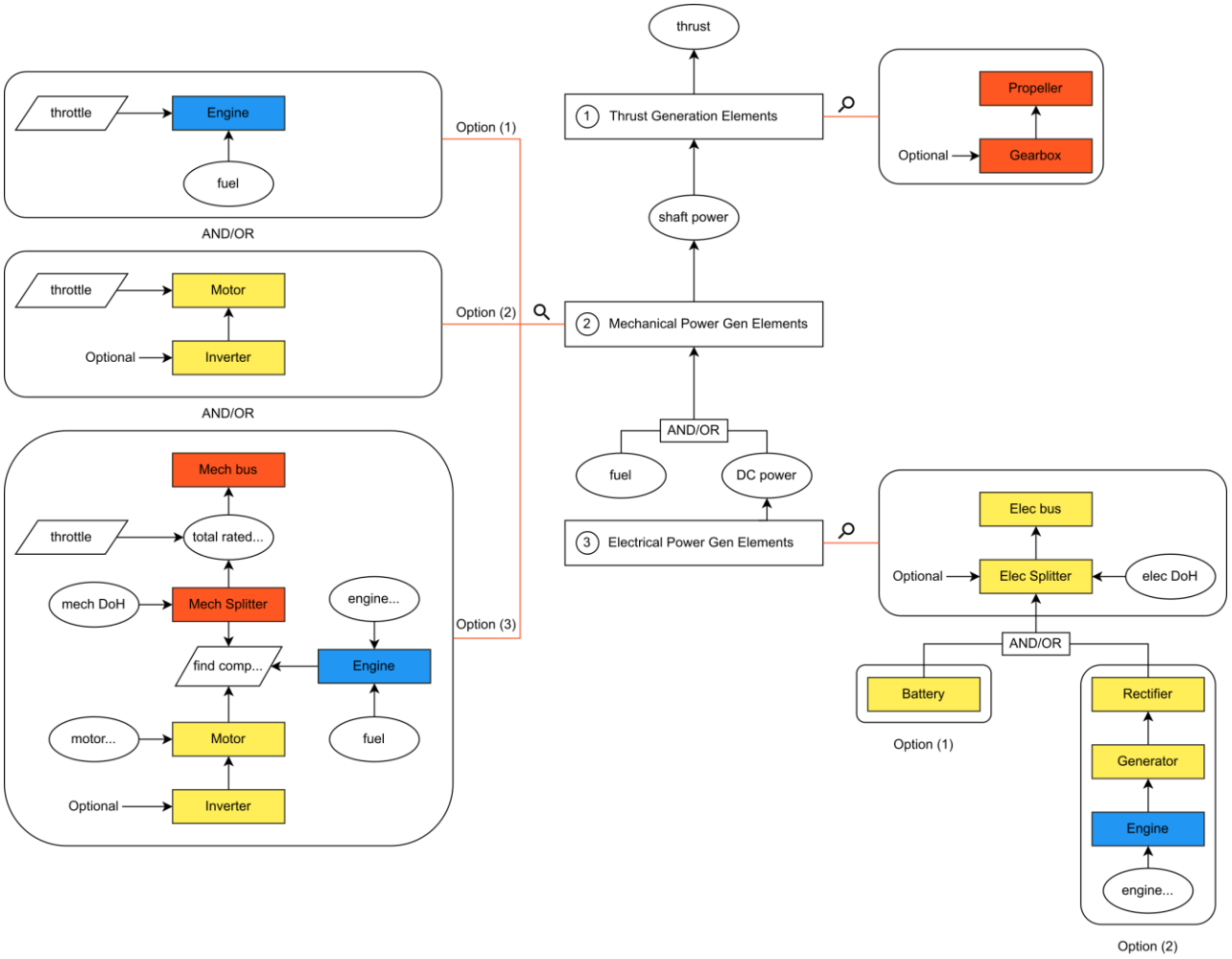

Figure 2.3: Architecture builder logic. Architectures are constructed from thrust, mechanical power, and electrical power generation elements. Figure adopted from (Fouda et al. 2022).

The appropriateness of the architecture builder for architecture optimization comes from the fact that all propulsion architectures have a common computational interface for connecting to analysis at a higher system level (e.g. mission analysis and aircraft sizing): the computational elements that are different between architecture instances are contained within the common interface, and are therefore effectively "hidden" from the rest of the computational problem. Additionally, the class-based definition used by the architecture builder makes it possible to use the Class Factory Evaluator (CFE) approach for converting ADORE architectures to evaluation models (i.e. the green boxes in fig. 2.2).

The architecture builder framework is integrated in the architecture optimization loop such that when a new architecture instance is created, the numerical model of the same architecture is automatically constructed and can be used to size the architecture and calculate its performance in terms of weight, fuel burn, and electrical energy usage.

3. Implementation

The methodology discussed in previous sections is implemented using several tools to cover the modeling, evaluation, and optimization of propulsion system architectures. This section describes the tools involved, and how they are connected to each other.

3.1. Structure of the Architecture Optimization Problem

The architecture optimization problem is implemented as a bi-level optimization loop:

1. An outer **architecture optimization** loop generates new propulsion system architectures that are analyzed by the inner loop. The architecture optimization loop is formulated as a multi-objective global optimization problem.
2. An inner **sizing optimization** loop sizes one particular architecture at a time, and calculates its performance in terms of weight, fuel burn, and electricity usage. The sizing optimization loop is formulated as a single-objective gradient-based (i.e. local) optimization problem

The architecture optimization loop starts by an optimizer suggesting some design vector x_{arch}, that is converted to an ADORE architecture instance. The Class Factory Evaluator (CFE) code then translates the ADORE architecture to a *PropSysArch* instance (see sec. 3.3), which is input for the architecture builder that constructs an OpenMDAO/OpenConcept model from it. The sizing optimization loop is then executed and resulting performance metrics are returned to the architecture optimizer, together with the imputed architecture design vector $x_{arch_{imp}}$, to start the next architecture optimization loop. This process is analogous to fig. 2.2 (where the sizing optimizer plays the role of problem-specific evaluation code) and is visualized in fig. 3.1. The architecture optimization problem, in terms of design vector, objectives, and constraints, is defined by the ADORE model. The design vector consists of architectural choices and several other architecture-level design variables, such as Degree of Hybridization (DoH) and number of propeller blades.

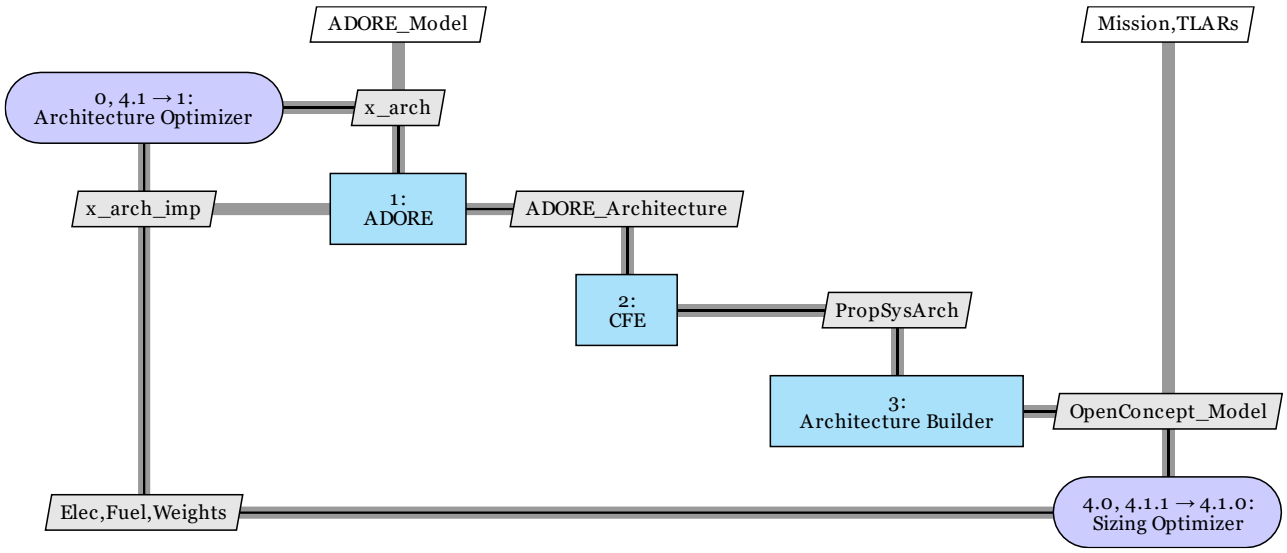

Figure 3.1: XDSM view of the architecture optimization loop. Compare to fig. 2.2. CFE means Class Factory Evaluator.

The sizing optimization loop takes one specific architecture instance at a time and sizes the aircraft, evaluates and optimizes the mission profile. During this process, the engines of the aircraft, as well as the batteries required to complete the mission (if the architecture has batteries), are sized. The aircraft is sized by the Overall Aircraft Design (OAD) tool OpenAD (Wöhler et al. 2020), which takes as inputs the union of the aircraft's Top-Level Aircraft Requirements (TLARs) and outputs of the mission simulation using OpenConcept. TLARs include the design range, cruise altitude, design

payload, cruise Mach number and the wing loading. The output of OpenAD is a consistent aircraft design, including geometry, weights breakdown, and drag polar. These parameters are taken as input to OpenConcept, together with the mission definition, and *PropSysArch*. OpenConcept then simulates the mission and calculates the propulsion system weight, amount of fuel used, and battery state-of-charge at the end of the mission. Fuel usages, the propulsion system weight, and mission segment durations are fed back into OpenAD. Fig. 3.2 shows the sizing optimization loop conceptually.

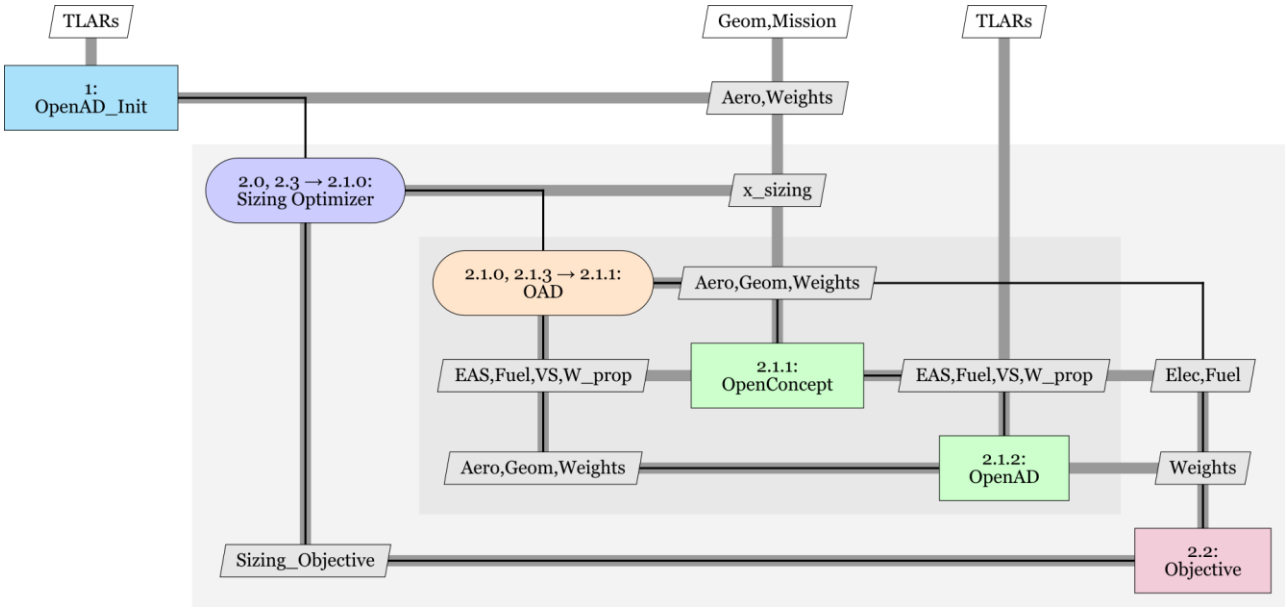

Figure 3.2: XDSM view of the sizing optimization loop. OAD mean Overall Aircraft Design. The Sizing_Objective parameter refers to f_{sizing}.

The sizing optimizer then optimizes the architecture by sizing the propeller blades, motors and engines to provide enough thrust, sizing the batteries such that there is enough charge left at the end of the flight, and optimizing airspeeds and vertical speeds for climb, cruise, and descend segments. Following objective is optimized for, based on (Fouda et al. 2022):

$$f_{sizing}(x_{sizing}) = (1 - t_{coeff}) \cdot (W_{fuel} + 0.01 \cdot MTOW) + t_{coeff} \cdot 0.01 \cdot t_{flight} \qquad (3.1)$$

with W_{fuel} the mission fuel burn, $MTOW$ the maximum take-off weight and t_{flight} the flight time . t_{coeff} is a design variable of the outer optimization loop that can take any value from 0 to 1 and that determines the importance of the different objectives when sizing the desired architecture. The higher its value, the more important reducing flight time is compared to “green” objectives (fuel consumed and MTOW) when sizing the architecture.

OpenAD takes in the order of 30 seconds to produce a consistent aircraft model, whereas OpenConcept takes only in the order of seconds to simulate the mission. To improve convergence speed, OpenAD is coupled to OpenConcept as a Kriging surrogate model (similar to the idea presented in (Dubreuil et al. 2020)). This surrogate model is created using the SMT library (Bouhlel et al. 2019), and is initialized by a Design of Experiments over OpenAD inputs. After OpenConcept and the OpenAD surrogate model have converged, the real OpenAD tool is executed to ensure that the surrogate model and the real outputs are consistent, and to update the surrogate model so it is more accurate in future evaluations.

3.2. Modeling the Architecture Design Space

The architecture design space model is an ADORE model mapping functions to components and modeling architectural decisions. The design space model starts from a boundary function: the main function a system provides to its users (Mavris, Tenorio & Armstrong 2008). For a propulsion system this is to "Provide Propulsive Power" (Esdras & Liscouët-Hanke 2015). This is fulfilled by the "Thrust generator" component, which in order to produce thrust needs another function to be provided, which is to "Accelerate air", see fig. 3.3. For this problem, only propellers are considered, although in a broader design space also turbofans or turboprops could fulfill this function. The optimization objectives, fuel consumption, Maximum Take-off Weight (MTOW), and flight duration are system-level Quantities of Interest (QOIs), and are therefore associated to the boundary function.

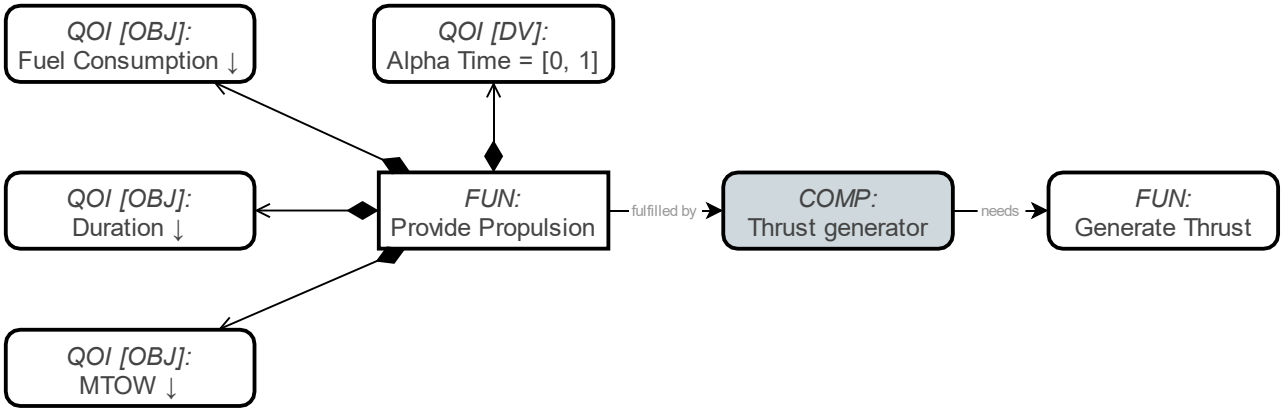

Figure 3.3: Origin of the design space model, where the boundary function as well as the different objectives are introduced. Element types: function (FUN), component (COMP), Quantity of Interest (QOI).

Propellers need two functions to be fulfilled, see fig. 3.4. The first is to decouple the RPM of the propeller from the RPM of the shaft , implemented by a gearbox. The second function is the generation of shaft power. In this case three different options arise (see fig. 3.4): the first option is to use a mechanical turboshaft, which would lead to the conventional architectures currently found on most turboprop aircraft. The second option would be to use only electrical power converted into shaft power by some electric motors. The third option would be a combination of the conventional and electric sources, leading to hybrid-electric architectures. In this last option, the component that is providing the power is a mechanic bus, and thus a component that is able to receive both electrical and mechanical power, and transmit them to the propellers, would also be needed. This would be done by a mechanic splitter.

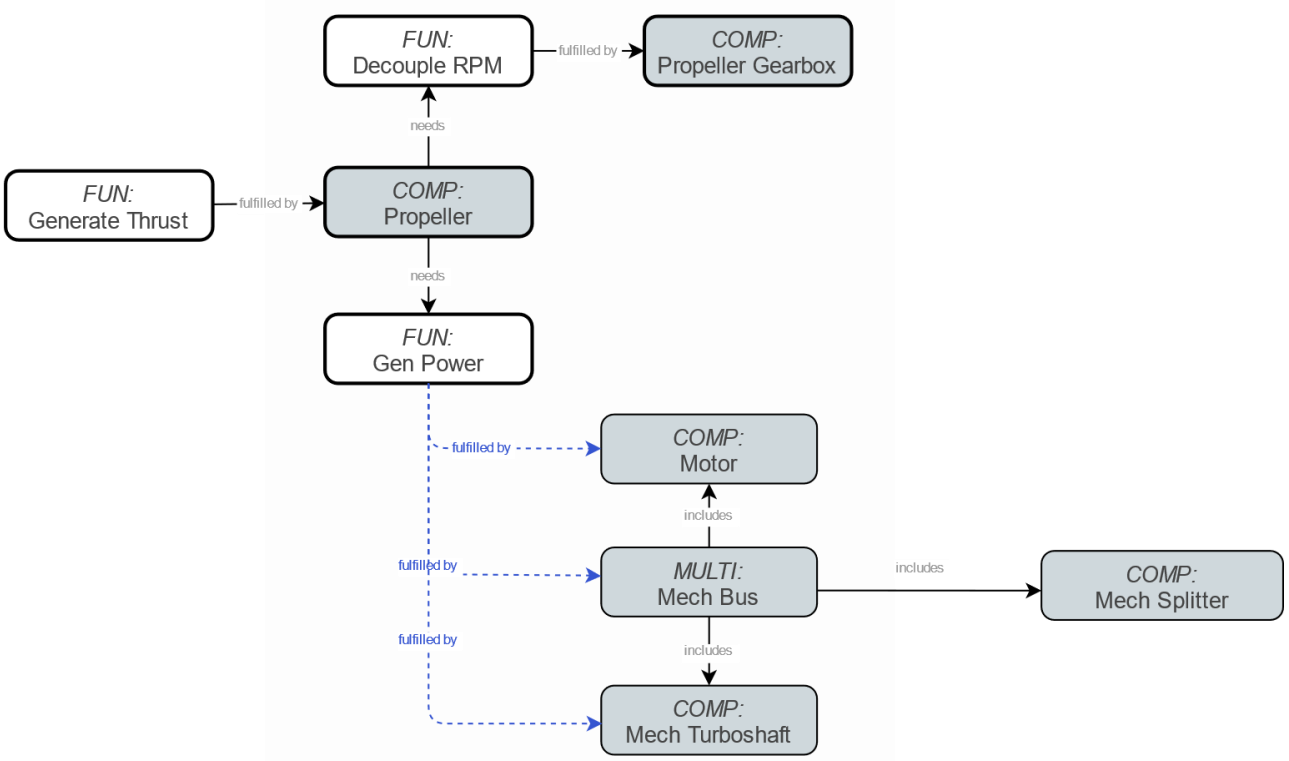

Figure 3.4: Functions to be fulfilled for the propeller component in order to produce thrust. Also, the different options in order to produce power shaft are introduced, and the blue-dashed lines represent an architectural decision. Element types: function (FUN), component (COMP), multi-fulfillment (MULTI).

If the electric motor is selected to provide (part of) the mechanical power, the architecture related to electric power generation and distribution is included. Motors only work with AC power, and therefore inverters are needed in order to convert the DC power provided by the DC bus into AC power, as shown in fig. 3.5.

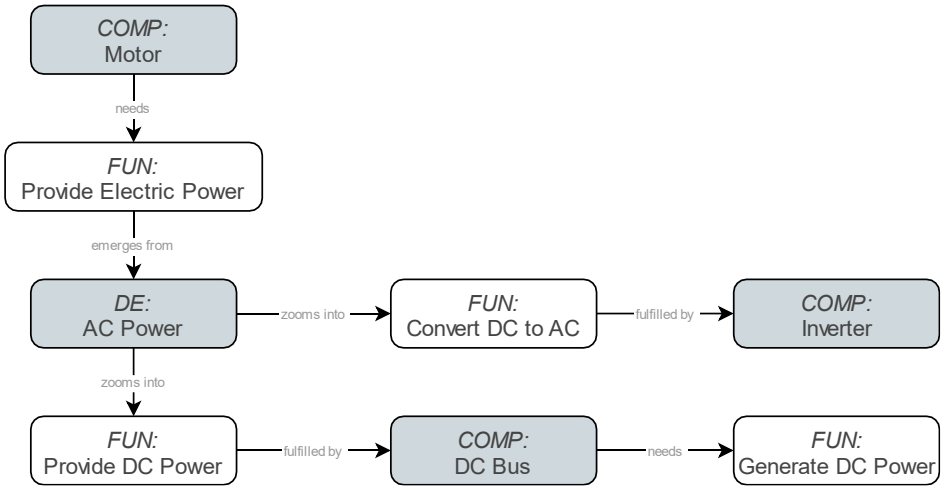

Figure 3.5: Functions that need to be fulfilled for the motor. Both the conversion from DC into AC and the generation of DC electrical power can be observed. Element types: function (FUN), component (COMP), decomposition (DE).

Then, there are three different options in order to produce this DC electrical power: the first option is to use batteries, the second option is to use a turboshaft to generate shaft power and attach it to a generator to generate DC power. In the second case, a rectifier would also be needed in order to convert the AC electrical power into the DC electrical power that the DC bus needs.

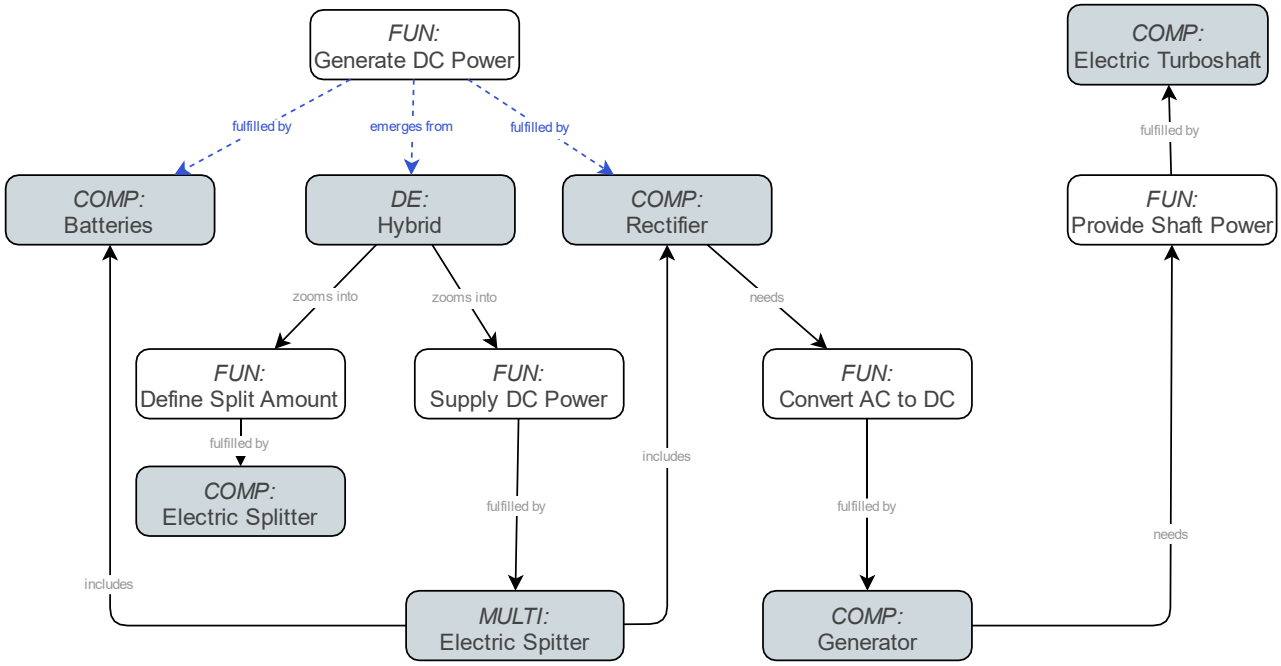

Figure 3.6: Different options for the generation of the electrical power needed by the motors. Element types: function (FUN), component (COMP), decomposition (DE), multi-fulfillment (MULTI).

Similarly as for generating mechanical power, the third option here is to generate part of the power from each of the two sources. In that case, part of the electrical power would be coming from the batteries and other part from the generator, and an electric splitter would be needed to merge the electrical power sources into a single input for the motors.

There are some decisions, such as Degree of Hybridization (DoH) for different mission segments, that are modeled at the component-level, see fig. 3.7. A DoH value of 0 means power comes from turbomachinery elements, and a value of 1 means power comes from electrical elements.

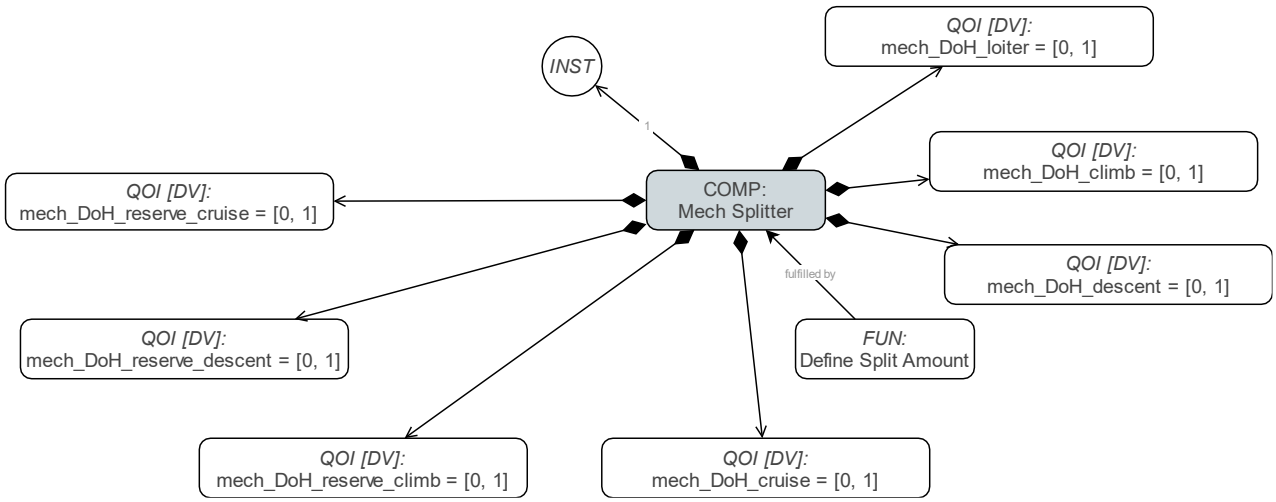

Figure 3.7: Implementation of the different DoH inside an ADORE component, in this case the mechanic splitter.

Each propeller is modelled as a subsystem, as shown at fig. 3.8 , as this allows to define the number of engines as an architecture choice. Between 1 and 5 engines per wing can be selected.

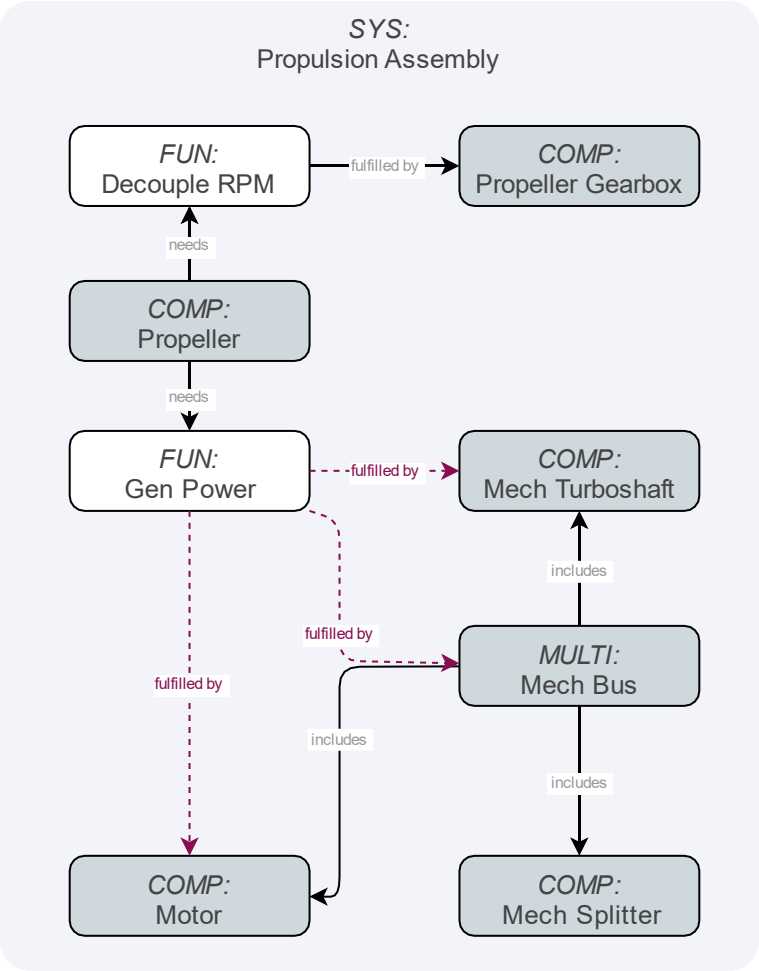

Figure 3.8: Example of the propeller subsystem, with the constrained choices represented in purple.

Each propeller can have its own source of shaft power. In this application case, the order at which the propeller subsystems are placed along the wing has no influence on calculated performance metrics. For example, a wing with two propellers where the first has a turboprop and the second has an electric motor for mechanical power generation results in the same weight, fuel burn, and flight time as a wing where these two have swapped places. This limitation stems from the available analysis tools that cannot represent effects of engine order on wing root bending moment, aerodynamic interactions or other phenomena. In order to avoid repeated definition of architectures with the same output metrics, a choice constraint is applied such that only unordered combinations of mechanical power generation components can be defined. The constrained nature of the decision is shown by purple-dashed lines in fig. 3.8

Combining the previous models, the complete design space can be observed in fig. 3.9

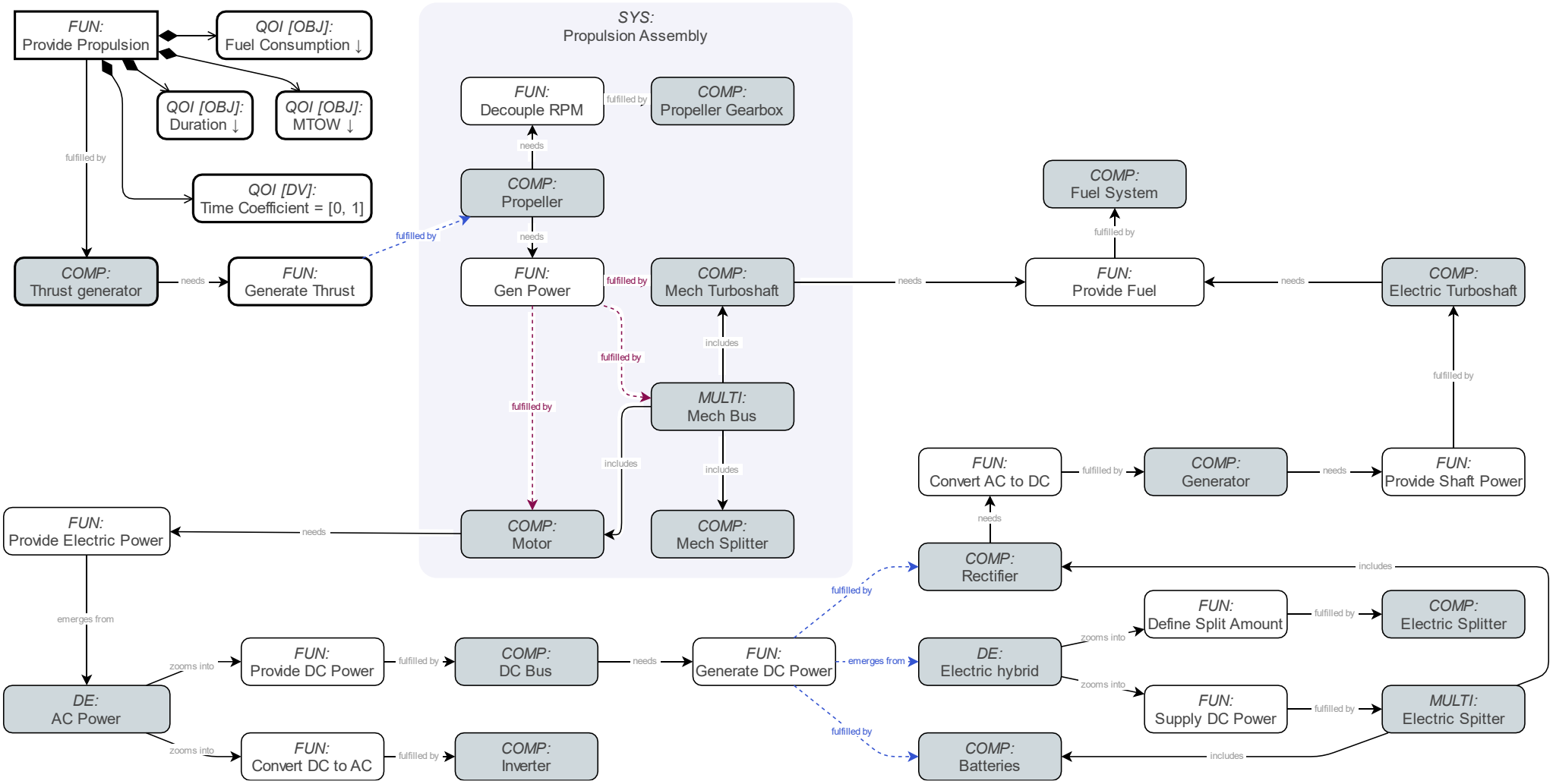

Figure 3.9: Complete Architecture Design Space Graph of the optimization problem modeled in ADORE.

Considering the choice constraints mentioned before and other assumptions, such as a constant number of blades for all propellers or the existence of a common electric generation source, the valid design space consists of 310 architectures. This number does not take continuous variables, such as the different degrees of hybridization. The architectural design space therefore is larger than only these 310 architectures.

3.3. Architecture Instance Model Translator

Each generated architecture instance needs to be translated to a *PropSysArch* instance in order to create and run the sizing optimization. This is done using class factories, which define the logic for instantiating classes based on component occurrence in architecture instances. For example, a class factory might specify to instantiate the *Propeller* class if the propeller component occurs in an architecture instance, and give it properties taken from associated Quantities of Interest such as diameter and number of blades. As the *PropSysArch* class itself is comprised of class instances representing architecture components, this plays very well with this class instantiation approach.

Each component class that can be part of the *PropSysArch* has an associated class factory, defining for which ADORE model elements it should be instantiated, and which QOIs from the ADORE model should be taken as property values. The architecture evaluation code itself then becomes relatively simple:

1. Instantiate component classes from the architecture instance using the class factories ("Class Factory Instantiation" in fig. 2.2);
2. Construct the *PropSysArch* instance ("Create Simulation Input");
3. Build and run the sizing optimization OpenMDAO problem ("Run Simulation Code");
4. Extract results from the optimization and return them to ADORE ("Extract Output Metrics").

4. Demonstration and Results

In this section, the design problem demonstrating hybrid-electric aircraft propulsion system architecture optimization is presented. The section finished with a description of the executed optimization problem, and a presentation and discussion of results.

4.1. Design Problem Description

The propulsion system architecture will be designed and optimized for the King Air C90GT aircraft: a popular twin-turboprop aircraft with a capacity for seven passengers. TLARs are obtained from (*Beechcraft kingair C90GT* n.d.; McClellan 2021; *OpenConcept KingAirC90GT* n.d.). A representation of the aircraft created using OpenAD is shown in fig. 4.1.

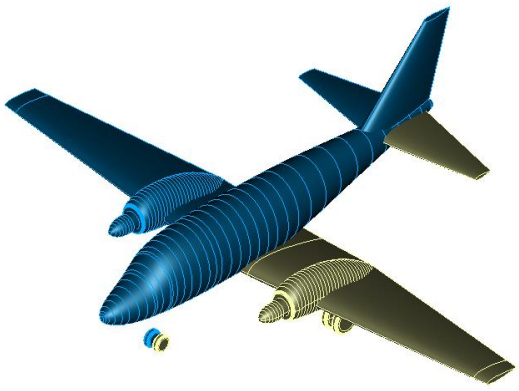

Figure 4.1: Visualization of the reference aircraft.

The design mission range is taken from a typical mission with 4 passengers, in order to have a fair comparison between different architectures. Wing loading is kept fixed for all architectures to meet take-off and landing distance requirements by automatically resizing the wing for changing MTOW. The simulated mission consists of climb, cruise, and descend segments, both for the main and the reserve mission, and a loiter segment to account for holding time. Furthermore, it is assumed that all turboshaft components have a specific fuel consumption of 0.6 lbf/hp/hr (*Engine specification - evolutionaircraft.com* n.d.), and batteries have an energy density of 1200 Wh/kg and specific power of 5000 W/kg.

Some of the design variables of the inner loop are common across all architectures, whereas others are only included when specific architecture components are present. For example, design variables related to power rating are specific for the power-generating elements, and therefore are only included if those elements are included. There are a total of 12 common design variables and 4 optional design variables, depending on the architecture that is chosen by the external optimizer, presented in table 4.1.

Table 4.1: Inner optimization loop design variables with the components that activate them.

Design variable	Activated for Component
Equivalent air speed at each segment (x7)	All
Vertical speeds (x4)	All
Propeller diameter	Propeller (always included)
Motor rating	Electric Motor
Engine rating	Mechanical Turboshaft
Electric engine rating	Electric Turboshaft
Batteries weight	Batteries

Constraints behave similarly: common constraints include ones for the throttle at each segment (7 constraints), as well as the propeller diameter. Then depending on the architecture, additional constraints might be included, such those that ensure that each component has the capacity of producing the power required at each segment (sizing margin constraints). Thus, there will be always at least 8 constraints that are common for all architectures, and there is a total of 50 optional constraints too, see table 4.2 for more details.

Table 4.2: Inner loop constraints with the components that activate them.

Constraint	Activated for Component
Throttle at each segment (x7)	All
Propeller diameter	Propeller (always included)
Turboshaft sizing margin (x14)	Mechanical Turboshaft
Motor sizing margin (x14)	Electric Motor
Electric turboshaft sizing margin (x7)	Electric Turboshaft
Batteries sizing margin (x14)	Batteries
Final state of charge (SOC)	Batteries

4.2. Architecture Optimization Results

The inner sizing loop is driven by the gradient-based SLSQP algorithm implemented in Scipy[1]. The outer architecture optimization loop is driven by a surrogate-based multi-objective optimization algorithm based on work presented in (Bussemaker, Bartoli, et al. 2021), with an initial Design of Experiments size of 100 points and 150 subsequent infill points.

[1] https://scipy.org/

Results of the architecture optimization are shown in fig. 4.2 where a Pareto front with a total of 88 design points has been obtained.

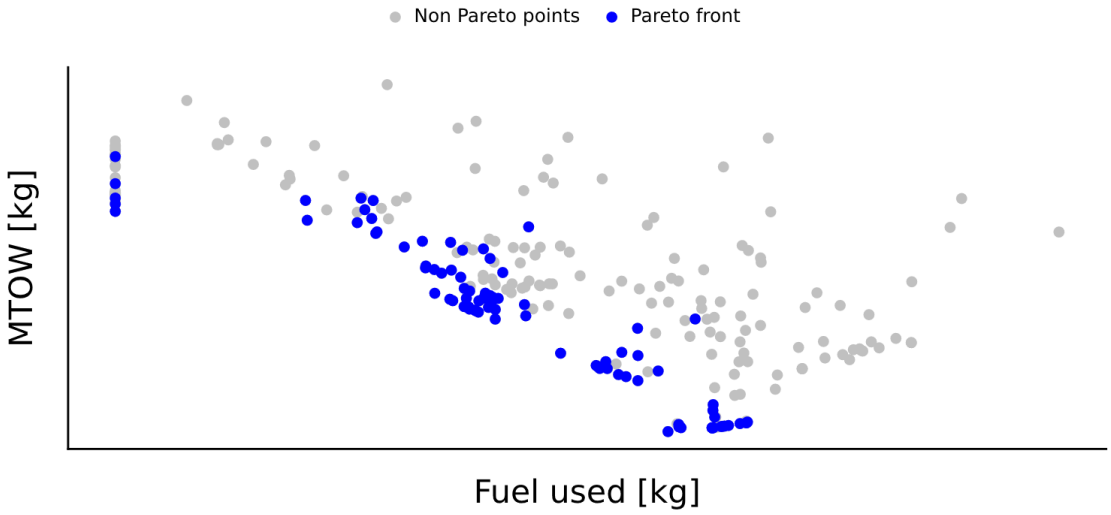

Figure 4.2: Plot showing all evaluated architectures. The Pareto front is shown in blue.

As shown in fig. 4.3, the extreme points of the Pareto front correspond to the fully electric architecture (lowest fuel consumption and highest MTOW) and the conventional architecture (highest fuel consumption and lowest MTOW). Between these two clusters of points, almost all the architectures that can be found in the Pareto front are parallel hybrid-electric architectures (i.e. mechanical power is generated both by electric motors and turboshafts), which can be explained by the great flexible performance of these architectures due to the different mechanic DoH they can work with.

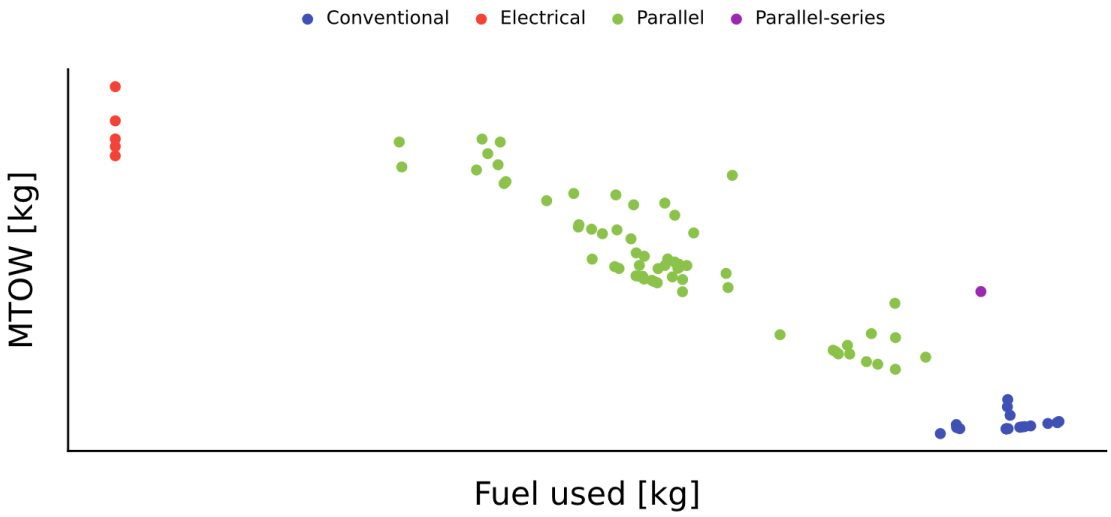

Figure 4.3: Different architectures of the Pareto front according to the source of power.

Finally, when it comes to the competitiveness of the aircraft in terms of flight time, a general trend emerges as seen in fig. 4.4: if a lower flight time is wanted, more energy will be necessary to complete the mission, in the form of a higher fuel consumption or a heavier aircraft. This trend is consistent with (Proesmans & Vos 2022).

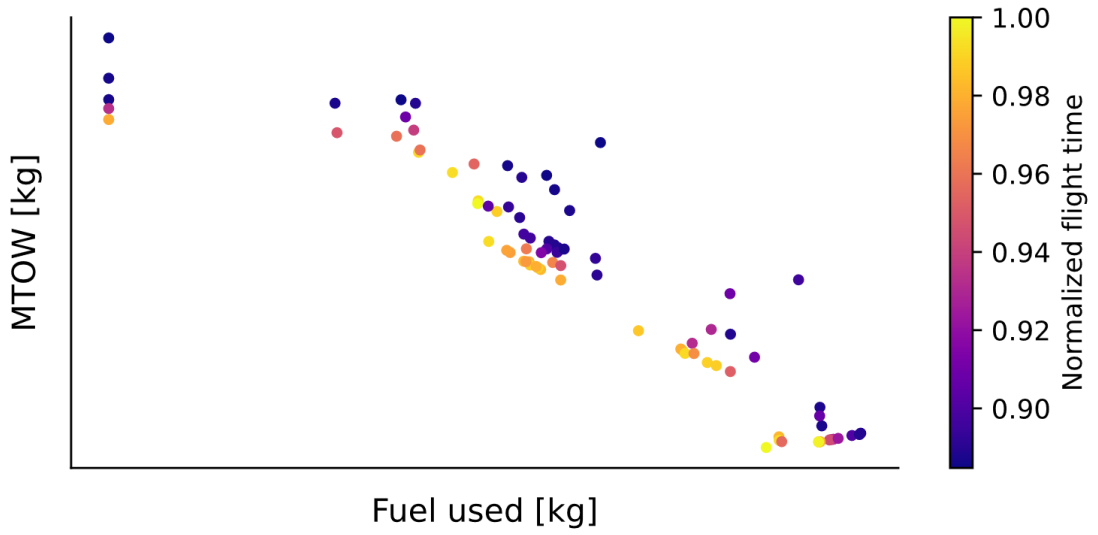

Figure 4.4: Flight time of the architectures found in the Pareto front. A lower normalized flight time (shown in darker color) represents a higher cruise speed.

5. Conclusion and Outlook

This work has shown that it is feasible to apply architecture optimization to hybrid-electric propulsion system design problems. For these types of problems, the architecture plays an important role in the performance of the system, and especially when searching for novel architectures numerical optimization techniques should be applied to enable searching a larger design space.

The architecture optimization problem is implemented in a bi-level formulation, with the outer optimization loop generating new architectures and optimizing for fuel burn, MTOW and flight time, and the inner optimization loop sizing specific architecture instances to ensure they can fly the mission. OpenConcept is used for mission and propulsion architecture simulation, OpenAD is used for overall aircraft design, and the sizing optimization problem is implemented in OpenMDAO. The OpenMDAO problem is automatically constructed for each architecture instance using an architecture builder code that uses Python class instances to define the propulsion system architecture.

The architecture design space is modeled using the Architecture Design Space Graph (ADSG) methodology, implemented in ADORE. The ADORE model is converted to the Python classes using the Class Factory Evaluator approach: class factories are rules that define how to instantiate classes for different architecture elements (e.g. the Propeller element instances the *Propeller* class, with certain properties). The architecture design space modeling method follows a function-based approach, where the main architectural decisions are defined from decisions regarding which component fulfills a function. In the case of the hybrid-electric propulsion architecture, for example, one such choice is how to generate mechanical power: either by electric motor, a turboshaft engine, or a combination of both.

The architecture optimization formulation is demonstrated by optimizing a hybrid-electric propulsion system for the King Air C90GT regional commuter aircraft. It is shown that a Pareto front trading-off fuel burn, MTOW and flight time can be obtained using a multi-objective evolutionary optimization algorithm. A Pareto front can then be used in subsequent decision-making and scenario analysis steps.

The potential of architecture optimization should be investigated along several directions. First, the design space size of this type of design problem should be increased to show that architecture optimization is also effective for extremely large number of possible architectures (e.g. 123 000

possible architectures as seen in (Frank et al. 2016)). Then, higher fidelity analysis and more disciplines should be included in the architecture optimization problem, including the possibility to integrate analysis tools from different partners. Finally, a more general method for implementing architecture evaluation using MDAO should be developed, so that less problem-specific code is needed and more logic can be reused across design problems.

6. Acknowledgments

Part of the research presented in this paper has been performed in the framework of the AGILE 4.0 project (Towards cyber-physical collaborative aircraft development) and has received funding from the European Union Horizon 2020 Programme under grant agreement n. 815122. Additionally, the authors would like to acknowledge the helpful discussions with Georgi Atanasov and Thomas Zill about the aircraft design problem.

References

'Beechcraft kingair C90GT' n.d., *Aviation broker*, viewed <https://www.aviation-broker.com/fileadmin/user_upload/flugzeuge/turboprops/raytheon_kingair90/C90GT_Spec_Perf.pdf>.

Biser, S, Atanasov, G, Hepperle, M, Filipenko, M, Keller, D, Vechtel, D, Boll, M, Kastner, N & Noe, M 2020, 'Design space exploration study and optimization of a distributed turbo-electric propulsion system for a regional passenger aircraft', *AIAA propulsion and energy 2020 forum*, American Institute of Aeronautics; Astronautics.

Bouhlel, MA, Hwang, JT, Bartoli, N, Lafage, R, Morlier, J & Martins, JRRA 2019, 'A python surrogate modeling framework with derivatives', *Advances in Engineering Software*, vol. 135, Elsevier BV, p. 102662.

Brelje, BJ & Martins, JR 2018, 'Development of a conceptual design model for aircraft electric propulsion with efficient gradients', *2018 AIAA/IEEE electric aircraft technologies symposium*, American Institute of Aeronautics; Astronautics.

Brelje, BJ & Martins, JRRA 2019, 'Electric, hybrid, and turboelectric fixed-wing aircraft: A review of concepts, models, and design approaches', *Progress in Aerospace Sciences*, vol. 104, Elsevier BV, pp. 1–19.

Bussemaker, JH, Bartoli, N, Lefebvre, T, Ciampa, Pier Davide & Nagel, B 2021, 'Effectiveness of surrogate-based optimization algorithms for system architecture optimization', *AIAA AVIATION 2021 FORUM*, American Institute of Aeronautics; Astronautics.

Bussemaker, JH & Boggero, L 2022, 'Technologies for enabling system architecture optimization', *ONERA-DLR aerospace symposium (ODAS) 2022*, viewed <https://elib.dlr.de/186720/>.

Bussemaker, JH, Boggero, L & Ciampa, PD 2022, 'From system architecting to system design and optimization: A link between MBSE and MDAO', *32nd annual INCOSE international symposium*, Detroit, MI, USA.

Bussemaker, JH, Boggero, L & Nagel, B 2023, 'The AGILE 4.0 project: MBSE to support cyber-physical collaborative aircraft development', *33rd annual INCOSE international symposium*, Honolulu, HI, USA.

Bussemaker, JH & Ciampa, PD 2022, 'MBSE in architecture design space exploration', in AM Madni, N Augustine & M Sievers (eds), *Handbook of model-based systems engineering*, Springer, viewed <https://elib.dlr.de/186203/>.

Bussemaker, JH, Ciampa, PD & Nagel, B 2020, 'System architecture design space exploration: An approach to modeling and optimization', *AIAA AVIATION 2020 FORUM*, American Institute of Aeronautics; Astronautics.

Bussemaker, JH, Ciampa, PD & Nagel, B 2021, 'System architecture design space modeling and optimization elements', *32nd congress of the international council of the aeronautical sciences, ICAS 2020*, Shanghai, China.

Ciampa, PD & Nagel, B 2021, 'Accelerating the development of complex systems in aeronautics via MBSE and MDAO: A roadmap to agility', *AIAA AVIATION 2021 FORUM*, American Institute of Aeronautics; Astronautics, Virtual Event.

Ciampa, PD & Nagel, B 2020, 'AGILE paradigm: The next generation of collaborative MDO for the development of aeronautical systems', *Progress in Aerospace Sciences*, vol. 119.

Ciampa, PD, Rocca, GL & Nagel, B 2020, 'A MBSE approach to MDAO systems for the development of complex products', *AIAA AVIATION 2020 FORUM*, American Institute of Aeronautics; Astronautics.

Crawley, E, Cameron, B & Selva, D 2015, *System architecture: strategy and product development for complex systems*, Pearson Education, p. 465.

Dubreuil, S, Bartoli, N, Gogu, C & Lefebvre, T 2020, 'Towards an efficient global multidisciplinary design optimization algorithm', *Structural and Multidisciplinary Optimization*, vol. 62, no. 4, Springer Science; Business Media LLC, pp. 1739–1765.

'Engine specification - evolutionaircraft.com' n.d., *PT6A-135A engine specification*, viewed <https://www.evolutionaircraft.com/wp-content/uploads/PT6A135A-803.pdf>.

Esdras, G & Liscouët-Hanke, S 2015, 'Development of core functions for aircraft conceptual design: Methodology and results', *Canadian aeronautics and space institute conference*, Montreal, CA.

Finger, DF, Vries, R de, Vos, R, Braun, C & Bil, C 2020, 'A comparison of hybrid-electric aircraft sizing methods', *AIAA scitech 2020 forum*, American Institute of Aeronautics; Astronautics.

Fouda, M, Adler, EJ, Bussemaker, JH, Martins, JRRA, Kurtulus, DF, Boggero, L & Nagel, B 2022, 'Automated hybrid propulsion model construction for conceptual aircraft design and optimization', *33rd congress of the international council of the aeronautical sciences, ICAS 2022*, Stockholm, Sweden.

Frank, CP, Marlier, R, Pinon-Fischer, OJ & Mavris, DN 2016, 'An Evolutionary Multi-Architecture Multi-Objective Optimization Algorithm for Design Space Exploration', *57th AIAA/ASCE/AHS/ASC structures, structural dynamics, and materials conference*, Reston, Virginia, pp. 1–19.

Gray, JS, Hwang, JT, Martins, JRRA, Moore, KT & Naylor, BA 2019, 'OpenMDAO: An open-source framework for multidisciplinary design, analysis, and optimization', *Structural and Multidisciplinary Optimization*, vol. 59, no. 4, Springer Science; Business Media LLC, pp. 1075–1104.

Hirtz, J, Stone, RB, McAdams, DA, Szykman, S & Wood, KL 2002, 'A functional basis for engineering design: Reconciling and evolving previous efforts', *Research in Engineering Design*, vol. 13, no. 2, Springer Science; Business Media LLC, pp. 65–82.

Iacobucci, JV 2012, *Rapid Architecture Alternative Modeling (Raam): a Framework for Capability-Based Analysis of System of Systems Architectures*, PhD thesis, Georgia Institute of Technology.

Jones, DR, Schonlau, M & Welch, WJ 1998, 'Efficient Global Optimization of Expensive Black - Box Functions', *Journal of Global Optimization*, vol. 13, pp. 455–492.

Judt, DM & Lawson, CP 2016, 'Development of an automated aircraft subsystem architecture generation and analysis tool', *Engineering Computations*, vol. 33, no. 5, pp. 1327–1352.

Mavris, D, Tenorio, C de & Armstrong, M 2008, 'Methodology for Aircraft System Architecture Definition', *46th AIAA aerospace sciences meeting and exhibit*, American Institute of Aeronautics; Astronautics, Reston, Virigina, pp. 1–14.

McClellan, JM 2021, 'Beechcraft C90GT', *FLYING Magazine*, viewed <https://www.flyingmag.com/pilot-reports/turboprops/beechcraft-c90gt/>.

McDonald, RA, German, BJ, Takahashi, T, Bil, C, Anemaat, W, Chaput, A, Vos, R & Harrison, N 2021, 'Future aircraft concepts and design methods', *The Aeronautical Journal*, vol. 126, no. 1295, Cambridge University Press (CUP), pp. 92–124.

NASA 2016, *NASA Systems Engineering Handbook*, NASA, p. 356.

'OpenConcept KingAirC90GT' n.d., *Openconcept KingAirC90GT*, viewed <https://github.com/mdolab/openconcept/blob/master/examples/aircraft_data/KingAirC90GT.py>.

Palaia, G, Zanetti, D, Salem, KA, Cipolla, V & Binante, V 2021, 'THEA-CODE: A design tool for the conceptual design of hybrid-electric aircraft with conventional or unconventional airframe configurations', J Morlier & M Kokkolaras (eds), *Mechanics &ampIndustry*, vol. 22, EDP Sciences, p. 19.

Palladino, V, Bartoli, N, Dubreuil, S, Bénard, E, Pommier-Budinger, V, Jordan, A, Schmollgruber, P & Defoort, S 2020, 'A comparative study of different propulsion models for hybrid electric aircraft', *3AF Aerospace Europe Conference 2020*, BORDEAUX, France, viewed <https://hal.science/hal-02904367>.

Palladino, V, Bartoli, N, Pommier-Budinger, V, Benard, E, Schmollgruber, P & Jordan, A 2023, 'Optimization of a hydrogen-based hybrid propulsion system under aircraft performance constraints', *Chinese Journal of Aeronautics*, Elsevier BV.

Pelamatti, J, Brevault, L, Balesdent, M, Talbi, E & Guerin, Y 2020, 'Bayesian optimization of variable-size design space problems', *Optimization and Engineering*, Springer Science; Business Media LLC.

Proesmans, P-J & Vos, R 2022, 'Airplane design optimization for minimal global warming impact', *Journal of Aircraft*, vol. 59, no. 5, American Institute of Aeronautics; Astronautics (AIAA), pp. 1363–1381.

Roelofs, M & Vos, R 2018, 'Uncertainty-based design optimization and technology evaluation: A review', *2018 AIAA aerospace sciences meeting*, American Institute of Aeronautics; Astronautics.

Rudenko, O & Schoenauer, M 2004, 'A Steady Performance Stopping Criterion for Pareto-based Evolutionary Algorithms', *6th international multi-objective programming and goal programming conference*, Hammamet, Tunisia.

Sahoo, S, Zhao, X & Kyprianidis, K 2020, 'A review of concepts, benefits, and challenges for future electrical propulsion-based aircraft', *Aerospace*, vol. 7, no. 4, MDPI AG, p. 44.

Saves, P, Diouane, Y, Bartoli, N, Lefebvre, T & Morlier, J 2022, *A mixed-categorical correlation kernel for gaussian process*, arXiv, viewed <https://arxiv.org/abs/2211.08262>.

Sgueglia, A, Schmollgruber, P, Bartoli, N, Benard, E, Morlier, J, Jasa, J, Martins, JRRA, Hwang, JT & Gray, JS 2020, 'Multidisciplinary design optimization framework with coupled derivative computation for hybrid aircraft', *Journal of Aircraft*, vol. 57, no. 4, American Institute of Aeronautics; Astronautics (AIAA), pp. 715–729.

Simmons, WL & Crawley, EF 2008, *A Framework for Decision Support in Systems Architecting*, PhD thesis, Massachusetts Institute of Technology, p. 198.

Sobieszczanski-Sobieski, J, Morris, A & Tooren, MJL van 2015, *Multidisciplinary design optimization supported by knowledge based engineering*, John Wiley & Sons, Ltd, pp. 1–378.

Vries, R de, Brown, MT & Vos, R 2018, 'A preliminary sizing method for hybrid-electric aircraft including aero-propulsive interaction effects', *2018 aviation technology, integration, and operations conference*, American Institute of Aeronautics; Astronautics.

Wöhler, S, Atanasov, G, Silberhorn, D, Fröhler, B & Zill, T 2020, 'Preliminary aircraft design within a multidisciplinary and multifidelity design environment', *Aerospace europe conference*, Bordeaux, France.

Biography

Jasper Bussemaker. Jasper received his MSc in Aerospace Engineering with a focus on aircraft design and MDO from Delft University of Technology in 2018. Currently he researches system architecture optimization and MDO at the DLR Institute of System Architectures in Aeronautics in Hamburg, Germany. He is developing methods and software for modeling architecture design spaces, for using MDO to evaluate candidate architectures, and for solving architecture optimization problems.

Raúl García Sanchéz. Raúl received his BSc in Aerospace engineering in Seville. As part of the MSc program on flight performance and propulsion at TU Delft, he performed his internship on system architecture optimization and MDO at the DLR Institute of System Architectures in Aeronautics in Hamburg, Germany, resulting in the work presented in this paper. Currently, he is researching the influence of system architecture on MDO problem formulation as part of his master thesis.

Mahmoud Fouda. Mahmoud obtained his BSc from the Aerospace Engineering Department of Middle East Technical University (METU) in 2019 and is currently pursuing his MSc from the same department. He joined the German Aerospace Center (DLR) Institute of System Architectures in Aeronautics in Hamburg in 2021 as a research student where he focused on architecture optimization application for electric and hybrid electric aircraft systems. He is currently working as a consultant engineer in the flight physics team at Capgemini in Hamburg.

Luca Boggero. Luca obtained in 2018 his PhD in Aerospace Engineering at Politecnico di Torino with a dissertation on Multidisciplinary Design and Optimization (MDO), Model Based Systems Engineering (MBSE) and design of aircraft subsystems. He now works as a Research Scientist at the DLR Institute of System Architectures in Aeronautics in Hamburg, and he leads the Digital Development Process Group. He coordinates and is involved in research projects within the context of MDO, Systems Engineering and MBSE.

Björn Nagel. Dr. Björn Nagel is director of the Institute of System Architectures in Aeronautics at the German Aerospace Center (DLR). His research is focused on digital engineering methods including MBSE and MDO enabling large and heterogeneous teams to model and optimize aviation as a system-of-systems. Pathways to climate-neutral air transport, co-design for industrialization and military air systems are major application.