The Influence of Architectural Design Decisions on the Formulation of MDAO Problems

Jasper H. Bussemaker, Sparsh Garg and Luca Boggero
Institute of System Architectures in Aeronautics, German Aerospace Center (DLR), Hamburg, Germany
Contact: jasper.bussemaker@dlr.de

Introduction

A system architecture defines how a system achieves its top-level functions, meets its requirements, and satisfies the stakeholder needs. The architecture design space, that is the space spanning all possible system architectures for a given design problem, suffers from a combinatorial explosion of alternatives, thereby making it infeasible to exhaustively search the complete design space to find the best architecture [1]. **System architecture optimization** techniques are currently being developed that enable the formulation of the design problem as a numerical optimization problem to enable design space exploration [2].

In an optimization process, architecture alternatives are compared using performance metrics, see Figure 1. To calculate these metrics, relevant engineering disciplines should be included, to prevent designs that favor one discipline (e.g. aerodynamics), and the behavior should be optimized at the system-level instead of the component-level. To achieve this, Multidisciplinary Design Analysis and Optimization **(MDAO)** techniques are used [3].

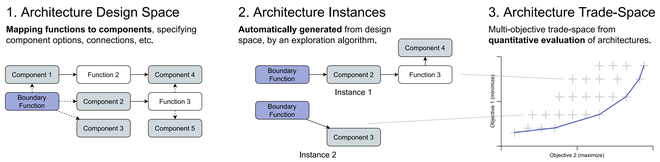

Figure 1: architecture optimization concept. Figure reproduced from [2].

Background

Architecture Optimization Problems

Formulating the system architecting process as an optimization problem brings several challenges. Architectural choices are often of discrete nature (e.g. the selection of aircraft engine type). Combined with continuous sizing parameters, this makes it a problem of **mixed-discrete** nature. Different architectural choices might lead to the inclusion or exclusion of other choices (e.g. the number of propeller blades is only relevant as a design variable if a propeller is chosen as propulsion instead of a turbofan): a **hierarchy** exists between design variables. Finally, the optimization problem can be **multi-objective** due to conflicting stakeholder needs.

Multidisciplinary Design Optimization (MDAO)

In MDAO problems, the inputs and outputs of different disciplinary tools are connected together and subsequently solved to end up with a dataset that is consistent between these tools. Two approaches to connecting data between tools in an MDAO problem can be distinguished [4]. In the **decentralized** approach, each tool follows their own naming convention for the data connections, and therefore it is the responsibility of the person setting up the MDAO problem to define connections between inputs and outputs of tools. This is the approach taken by MDAO platforms like OpenMDAO (not using data promotion) [5], and ModelCenter[1].

In **centralized** approaches, data connections can be discovered automatically because tools adhere to some centralized data protocol [6], either *tailor-made* or *predefined*. GEMSEO [7], WhatsOpt [8] and OpenMDAO (using data promotion) use tailor-made data mapping approaches. KADMOS [6] and MDAx [9] are based on the use of a centralized predefined data format. In aircraft design, an example of a predefined data format is CPACS [10].

[1] https://www.phoenix-int.com/product/modelcenter-integrate/, accessed September 2022.

Architecture-to-MDAO Influence Vectors

The MDAO problem can be influenced by generated architecture alternatives through various levels of influence vectors (see Figure 2).

L1 **Data values:** input values might be changed, for example when choices are represented using enumerations (e.g. choose between "turboprop" or "turbofan" propulsion) or integers (e.g. the number of ribs in a wing, the number of engines). No MDAO problem modification is needed.

L2 **Data structure (flat):** the multiplicity of elements can be affected, thereby influencing the length of data vectors to be communicated between tools (e.g. each rib might have a thickness and the number of ribs is a design variable). In this case the MDAO problem needs to support changing vector lengths or ignore parts of vectors that are not active.

L3 **Data structure (subtree):** more influential architecture choices might also influence the existence of certain data structure somewhere in the data hierarchy (e.g. the choice between a fixed- or rotary-wing aircraft might greatly influence what data is relevant for analysis: wing vs rotor attachment, flight controls assignment, etc.). In this case the MDAO problem needs to support changing data subtrees or ignoring larger subparts of the data structure.

L4 **Data connections:** component connection choices can influence the data flow between MDAO tools (e.g. landing forces are either an input to the wing- or fuselage-sizing tool, depending on their attachment). The MDAO problem needs to either support dynamically activating connections, or reformulation is needed.

L5 **Tool configuration:** disciplinary tools might be reused for analysis of different components (e.g. a structural sizing tool might be reused for wing and tailplane sizing). In this case the MDAO problem needs to support selecting different parts of the data structure for different uses of the same tool, or it should support automated redefinition of the input and output declaration of the tool.

L6 **Tool selection:** different technologies with different design procedures and engineering disciplines might be selected (e.g. designing composites or metallic structure might need different simulation tools). In this case the MDAO problem needs to either support dynamically skipping tools that are not needed for a given architecture, or it should be reformulated.

Note: L = level

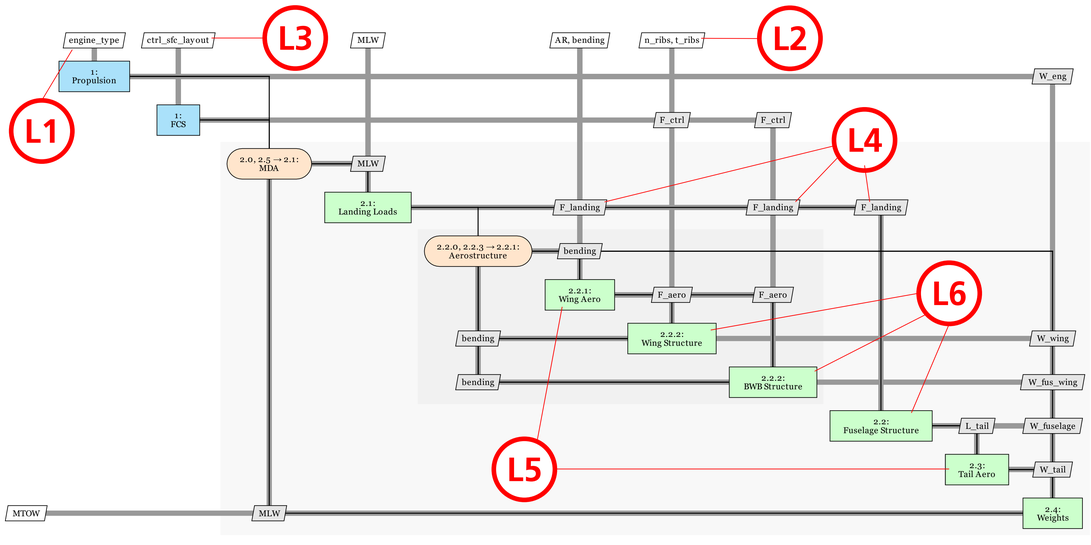

Figure 2: an example MDAO problem showing the different levels of architecture-to-MDAO influence vectors.

Mitigation I: Data Mapping Approach

A **decentralized** data mapping approach can work for dealing with L1 (values) and L2 (flat data structure) influence vectors, as these do not directly influence the MDAO problem formulation. Dealing with L3 (subtree data structure) is more challenging, because it requires relevant subtrees to be removable in the data model, which means that data connections can only be defined up to the root of the subtree and the MDAO problem needs to be agnostic to the contents of the subtree. Any higher-level influence vectors cannot be dealt with by a decentralized data mapping approach, as it prevents the automated redefinition of connections and tool inclusion.

Using **centralized** data mapping facilitates automatically establishing data connections between tools, and therefore allows any level of influence vector to be supported. For example, data connections can be automatically reestablished, and disciplinary tools can be automatically selected based on the availability of certain data elements.

Mitigation II: Tool Flexibility

Increasing the **area of validity** of a tool (e.g. the range of Mach numbers producing valid aerodynamics results) might make the tool more robust to changes in input values (L1; data values), especially when the same tool is reused for different parts of the architecture (L5; tool configuration).

Another aspect is related to **input parsing**. If a tool is agnostic with respect to data repetition and robust to changes in the subtree data structures, it can deal with L2 (flat data structure) and L3 (subtree data structure), respectively. An example of this would be a flight dynamics tool that can deal with different wing control surface configurations. Robustness to data structure can also aid in supporting L4 (data connections), by adding or removing data variables related to different connections.

The reuse of tools for different parts of the architecture can be implemented using a tool-level **global-to-local data wrapper** (see L5; tool configuration). The conversion logic can either result in multiple tool definitions, or be implemented in the MDAO problem itself.

Finally, a shift of **tool purpose** from component- (e.g. propeller, fuel tank) to discipline-oriented (e.g. aerodynamics, mission analysis) can make the tool more robust to data structures changes (L2 and L3) and aid in reusability (L5). However, it needs to be ensured that the discipline-oriented tool itself is flexible enough to be able to construct internal models of (parts of) the architecture to be analyzed.

Mitigation III: Problem Flexibility

Problem-level flexibility can be implemented in a **single** MDAO problem. An advantage is that this can be done before running the optimization, thereby giving the user more control over problem execution. If tools are robust enough, the single problem can deal with L1 to L4 influence vectors. Support for L5 (tool configuration) and L6 (tool selection) can be enabled using skipping logic: the problem itself contains the union of all available tools, which are then selectively skipped based on data values and availability. For example, if aluminum is chosen as the material, the composites tool is skipped. The single problem approach, however, cannot deal with changing optimal tool sequences or MDAO architectures.

Another way is by defining **multiple** MDAO problems and then delegating analysis of architecture alternatives to the corresponding problem. The advantage over the single MDAO problem approach is that here each problem definition can have different tool sequences or MDAO architectures applied for dealing with L5 and L6 influence vectors.

Finally, the most flexible approach is **automatically creating MDAO problems** for each new architecture alternative within the optimization loop. This ensures that each MDAO problem is optimally configured to analyze each specific architecture. To use the automated approach, there should be no additional user interaction needed for problem formulation and execution. This means it will be difficult to use GUI-based integration platforms like RCE[2]. Scripting-based frameworks like OpenMDAO [5] can alleviate this.

[2] https://rcenvironment.de/, accessed September 2022.

Bibliography

[1] J.H. Bussemaker, et al., "System Architecture Design Space Exploration: An Approach to Modeling and Optimization," in AIAA Aviation 2020 Forum.
[2] J.H. Bussemaker and P.D. Ciampa, "MBSE in Architecture Design Space Exploration," in Handbook of Model-Based Systems Engineering, Springer, 2022.
[3] P.D. Ciampa and B. Nagel, "Accelerating the Development of Complex Systems in Aeronautics via MBSE and MDAO: a Roadmap to Agility," in AIAA Aviation 2021 Forum.
[4] I. van Gent, "Agile MDAO Systems: A Graph-based Methodology to Enhance Collaborative Multidisciplinary Design," 2019.
[5] J.S. Gray, et al., "OpenMDAO: an open-source framework for multidisciplinary design, analysis, and optimization," Structural and Multidisciplinary Optimization, March 2019.
[6] I. van Gent, et al., "Composing MDAO symphonies: graph-based generation and manipulation of large multidisciplinary systems," 18th AIAA/ISSMO MDAO Conference, 2017.
[7] F. Gallard, et al., "GEMS, a Generic Engine for MDO Scenarios : Key Features in Application," in AIAA Aviation 2019 Forum.
[8] R. Lafage, et al., "WhatsOpt: a web application for multidisciplinary design analysis and optimization," in AIAA Aviation 2019 Forum.
[9] A. Page-Risueño, et al., "MDAx: Agile Generation of Collaborative MDAO Workflows for Complex Systems," in AIAA Aviation 2020 Forum.
[10] M. Alder, et al., "Recent Advances in Establishing a Common Language for Aircraft Design with CPACS," in Aerospace Europe Conference, 2020.

Deutsches Zentrum für Luft- und Raumfahrt
German Aerospace Center

AGILE 4.0

European Commission | Innovation and Networks Executive Agency

3rd European Workshop on MDO
20-21 September 2022, Paris, FR

This research has received funding from the European Union Horizon 2020 Programme under grant agreement n. 815122.