**Università degli Studi di Napoli Federico II**

**Scuola Politecnica e delle Scienze di Base**

Department of Industrial Engineering
Aerospace Division



DOCTORAL THESIS
IN
INDUSTRIAL ENGINEERING


# $3^{rd}$ generation of collaborative MDO for more efficient aircraft preliminary design. The turboprop case within the AGILE project


Coordinator:
**Prof. Michele Grassi**

Supervisors:
**Prof. Fabrizio Nicolosi**
**Prof. Pierluigi Della Vecchia**

Author:
**Luca Stingo**
**Matricola DR992751**



**January 24, 2020**

# List of Figures

# List of Tables

# Nomenclature

**Acronyms and Abbreviations**

| | |
|---|---|
| $ACSYNT$ | AirCraft SYNThesis |
| $BLI$ | Boundary Layer Ingestion |
| $BW$ | Box-Wing |
| $BWB$ | Blended Wing Body |
| $CA$ | Collaborative Architecture |
| $CAD$ | Computer Aided Design |
| $CFD$ | Computational Fluid Dynamics |
| $CMDOWS$ | Common MDO Workflow Schema |
| $CPACS$ | Common Parametric Aircraft Configuration Schema |
| $DAF$ | Design of Aircraft and Flight technologies |
| $DC$ | Design Campaign |
| $DLR$ | Deutschen Zentrums für Luft- und Raumfahrt |
| $DoE$ | Design of Experiments |
| $\epsilon - NSGAII$ | Epsilon-Dominance Nondominated Sorting Genetic Algorithm II |
| $EIS$ | Entry Into Service |
| $EU$ | European Union |
| $FPG$ | Fundamelntal Problem Graph |
| $GWP$ | Global Warming Potential |
| $IOSO$ | Indirect Optimization based on Self-Organization |
| $KA$ | Knowledge Architecture |
| $KADMOS$ | Knowledge- and graph-based Agile Design for Multidisci-plinary Optimization System |
| $KBE$ | Knowledge Based Engineering |
| $KBSE$ | Knowledge Based System Engineering |
| $MDG$ | MDAO Data Graph |
| $MPG$ | MDAO Process Graph |
| $MDAO$ | Multidisciplinary Design Analysis and Optimization |
| $MOB$ | Multidisciplinary Optimisation of a Blended Wing Body |
| $MOEA$ | MultiObjective Evolutionary Algorithms |
| $OAD$ | Overall Aircraft Design |
| $OMOPSO$ | Optimised Multiobjective Particle Swarm Optimisation |
| $PIDO$ | Process Integration and Design Optimization |

| | |
|---|---|
| $RCE$ | Remote Component Environment |
| $RCG$ | Repository Connectivity Graph |
| $RM$ | Rear Mounted |
| $SBW$ | Strut-Braced Wing |
| $SOTA$ | State Of The Art |
| $TLAR$ | Top Level Aircraft Requirements |
| $UAV$ | Unmanned Aircraft Vehicle |
| $UniNa$ | University of Naples |
| $VISTOMS$ | VISualization TOol for MDO Systems |
| $WM$ | Wing Mounted |
| $WP$ | Work Package |
| $XML$ | eXtensible Markup Language |
| $XDSM$ | eXtended Design Structure Matrix |

**Symbols**

| | |
|---|---|
| $A/C$ | Aircraft |
| $AR_w$ | Wing aspect ratio |
| $b_w$ | Wing span |
| $b_h$ | Horizontal tail span |
| $b_h$ | Vertical tail span |
| $BPR$ | Bypass Ratio |
| $c_r$ | Root chord |
| $c_k$ | Kink chord |
| $c_k$ | Tip chord |
| $CG$ | Center of Gravity |
| $C_{Dw}$ | Wing drag coefficient |
| $C_{Lmax_w}$ | Wing maximum lift coefficient |
| $COC$ | Cash Operating Costs |
| $d_F$ | Fuselage diameter |
| $DOC$ | Direct Operating Costs |
| $ISA$ | International Standard Atmosphere |
| $L_{max}$ | Maximum lift force |
| $\Lambda_{c/4}$ | Wing quarter chord sweep angle |
| $\lambda_w$ | Wing taper ratio |
| $L0/L1/L2/L3$ | Tools fidelity levels |
| $L_F$ | Fuselage length |
| $LFL$ | Landing Field Length |
| $mac$ | mean aerodynamic chord |
| $M_d$ | Dive Mach number |
| $MLM$ | Maximum Landing Mass (also MLW) |
| $MOEM$ | Maximum Operating Empty Mass |
| $M_{mo}$ | Max operation Mach number |
| $MTOM$ | Maximum Take-off Mass (also MTOW) |
| $q$ | Dynamic pressure |

| | |
|---|---|
| $S_h$ | Horizontal tail area |
| $S_h$ | Vertical tail area |
| $S_w$ | Wing area |
| $SL$ | Sea Level |
| $TOFL$ | Take-off Field Length |
| $VMC$ | Minimum Control Speed |
| $V_{mo}$ | Max operation speed |
| $X_{LE_w}$ | Wing leading edge position along the x axis |
| $W_w$ | Wing weight |

# Contents

# Abstract

One of the most important aspect within the aircraft design discipline is to be able to perform rapid and reliable analyses on different aircraft configurations. In this scenario has arisen the AGILE (Aircraft $3^{rd}$ Generation MDO for Innovative Collaboration of Heterogeneous Teams of Experts) project. This one was a successful European project, part of the *HORIZON 2020* programme, which gave a relevant contribution to the state of the art of Multidisciplinary Design Analysis and Optimization (MDAO) approach in aircraft design. The project aimed to create an evolution of MDO, promoting a novel approach based on collaborative remote design and knowledge dissemination among various and heterogeneous teams of experts. To achieve this goal, great importance was attached to massive collaboration among experts, to the development of technologies to allow remote and distributed design approach between partners worldwide dislocated and to the development of tool and/or new optimization methodologies.

The collaboration aspect was strongly faced by the author by setting up a whole aircraft design and optimization toolchain in terms of data exchanging, tools integration, workflow implementation and execution. Furthermore, thanks to the aircraft design tools and a new optimization technique which couples Nash game theory and genetich algorithm developed by the author, well-assessed AGILE technologies and setup of the whole framework, a complete MDAO task was performed on different disruptive aircraft configurations, with a main focus of the candidate on two innovative regional turboprop architectures.

This work can be divided in two main parts: the first is a more descriptive one, to best clarify the global scenario in which the research was conducted, and the second moves the focus on the major author's contributions to the AGILE project.

# Chapter 1

# Aircraft design approaches

## Introduction

In the past, around the twenties, the aircraft design time was very short since the market demand and technology about propulsion, structures, aerodynamics were very limited; there was only one kind of engine characterized by low power and so low speed. For these reasons it was possible to design and produce a new aircraft within six months; Anthony Fokker and his staff produced fourteen aircraft during a period of eighteen years [1].

Nowadays market requests are focused on increasing speed, range and number of passengers ensuring, at the same time, high level target of safety matching environmentally friendly solutions. Achieving these goals means to involve several disciplines and so needs heterogeneous teams of experts in the aircraft design process aiming to a deep and strong collaboration. This approach leads to lengthen the preliminary and conceptual design phase time inevitably. The focal point is that the collaboration among experts in several disciplines is often considered a simple way of working, but actually is not trivial even considering barriers like intellectual property, resources protection, licensing, security polices about data exchanging and so on. These obstacles can change a *simple* way of working to a *complex* one. AGILE European project [2] fits at that point aiming to develop an advanced and efficient MDO multi-site collaboration techniques to reduce convergence time and to face the lack of knowledge on how to setup analyses and optimization workflows involving multiple disciplines [3] [4].

## 1.1 Traditional aircraft design

Until few decades ago the aircraft design has been driven by a traditional approach explained by varoius important authors like Roskam, Perkins, Reymer, Torenbeek. In particular Egbert Torenbeek in *"Synthesis of subsonic airplane design" [1]* defines the aircraft design and says: *"'this is not an activity carried on in remote offices by specialists generating designs of any kind that may occur to their imaginations. There is close interaction between the development work for a new aircraft type and the other factors which together determine the growth of and/or changes in aeronautical activities."*
Daniel P. Raymer in [5] *"Aircraft Design: A Conceptual Approach"* adds: "*the aircraft design is a separate discipline of aeronautical engineering, different from the analytical disciplines such as aerodynamics, structures, controls, and propulsion. An aircraft designer needs to be well versed in these and many other specialties but will actually spend little time performing such analysis in all but the smallest companies. Instead, the designer's time is spent doing something called "design", creating the geometric description of a thing to be built."* Since the design involves specialists, customers and designers, it is hard to find a single starting point to begin the design process. For these reasons it is better to speak about the *"Design Wheel"*, shown in Fig. 1.1, to infuse the idea that design is an iterative process [5].

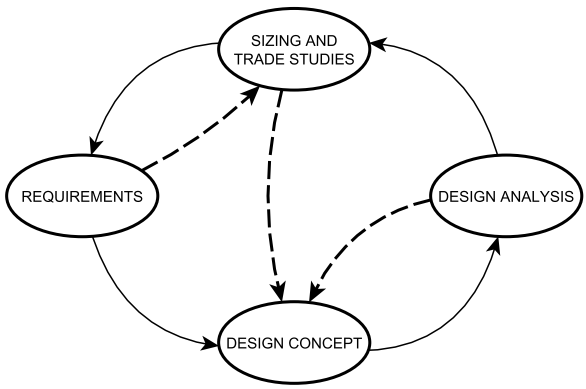



**Figure 1.1:** Aircraft Design Process: The "Design Wheel" [5].

According to all these authors the aircraft design process, also described in Fig. 1.2, is very articulate and complex but is possible to divide it in three big phases [5]:

- Conceptual Design
  In this phase the basic questions of configuration arrangement size and

weight, and performance are answered.
The first question is, "Can an affordable aircraft be built that meets the requirements?" If not, the customer may wish to relax the requirements.
Conceptual design is a very fluid process. New ideas and problems emerge as a design is investigated in ever-increasing detail. Each time the latest design is analyzed and sized, it must be redrawn to reflect the new gross weight, fuel weight, wing size, engine size, and other changes.

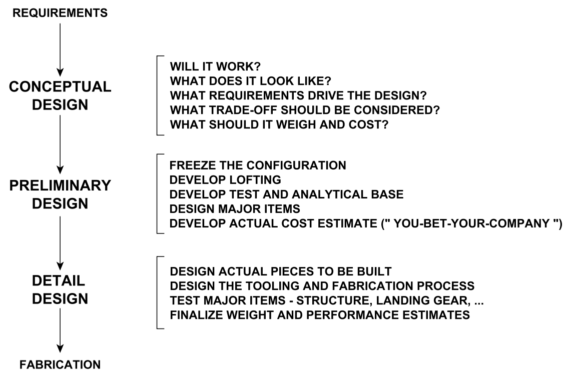



**Figure 1.2:** Aircraft design: three most important phases [5].

The final product is a conceptual layout of the aircraft configuration on paper or computer screen, to be reviewed by engineers and other designers.

- Preliminary design phase
  It can be said to begin when the major changes are over.
  The big questions such as whether to use a canard or an aft tail have been resolved. The configuration arrangement can be expected to remain about as shown on current drawings, although minor revisions may occur. At some point late in preliminary design, even minor changes are stopped when a decision is made to freeze the configuration.
  During preliminary design phase, the specialists in areas such as structures, landing gear, and control systems will design and analyse their portion of the aircraft. Testing is initiated in disciplines such as aerodynamics, propulsion, structures, and stability and control. A mock-up may be constructed at this point.
  A key activity during preliminary design is "lofting." Lofting is the mathematical modeling of the outside skin of the aircraft with sufficient accuracy to ensure proper fit between its different parts, even if they are designed by different designers and possibly fabricated in different locations.
  The ultimate objective during preliminary design is to ready the com-

pany for the detail design stage, also called full-scale development. Thus, the end of preliminary design usually involves a full-scale development proposal. In today's environment, this can result in a situation jokingly referred to as "you-bet-your-company." The possible loss on an overrun contract or from lack of sales can exceed the net worth of the company! Preliminary design must establish confidence that the airplane can be built on time and at the estimated cost.

- Detail design phase
  Assuming a favorable decision for entering full-scale development, the detail design phase begins when the actual pieces to be fabricated are designed. For example, during conceptual and preliminary design the wing box will be designed and analyzed as a whole. During detail design, that whole will be broken down into individual ribs, spars, and skins, each of which must be separately designed and analyzed. Another important part of detail design is called production design. Specialists determine how the airplane will be fabricated, starting with the smallest and simplest subassemblies and building up to the final assembly process. Production designers frequently wish to modify the design for ease of manufacture; that can have a major impact on performance or weight. Compromises are inevitable, but the design must still meet the original requirements.
  During detail design phase, the testing effort intensifies. Actual structure of the aircraft is fabricated and tested. Control laws for the flight control system are tested on an "iron-bird" simulator, a detailed working model of the actuators and flight control surfaces. Flight simulators are developed and flown by both company and customer test-pilots. Detail design ends with fabrication of the aircraft. Frequently the fabrication begins on part of the aircraft before the entire detail-design effort is completed. Hopefully, changes to already-fabricated pieces can be avoided. The further along a design progresses, the more people are involved. In fact, most of the engineers who go to work for a major aerospace company will work during preliminary or detail design phases.

A process like the described one is time consuming and expensive. Moreover, the steps of the whole process are often carried out by different teams and few, or even without, communication among them leading to lengthen the time to evaluate the right requirements and found the right answers. In particular, around the 80% of the total time is spent for repetitive work rather than for creative actions. The main goal is to invert or at least modify these percentages and reduce the time necessary for the entire process. So, it is necessary to work harder to develop new aircraft design methodologies and approaches. by encouraging the evolution from the *monolithic software* to *collaborative design.*

## 1.2 Origin of aircraft design: *the monolithic software*

Since the design of complex system, like aircraft, concerns a lot of disciplines and so a lot of specialists distributed in various groups, the main need is to report to a single chief designer or group well versed in all disciplines and in order to reduce problems related to organization and communication.
When this way of thinking is restricted to simple problems characterized by approximate analysis the results are satisfying; as the design problems become more complex the single group or chief designer is unable to monitor the whole process.
This kind of approach, called *Monolithic Design*, was used to carry out the conceptual design phase in the past but nowadays is difficult to maintain it [7].
The term monolithic is defined as: *"something created in one piece, resembling a monolith such as an obelisk. It mostly signifies artifacts without any subcomponents, i.e. a non-modularized, non-componentized, non-dismantlable building block."* [8]

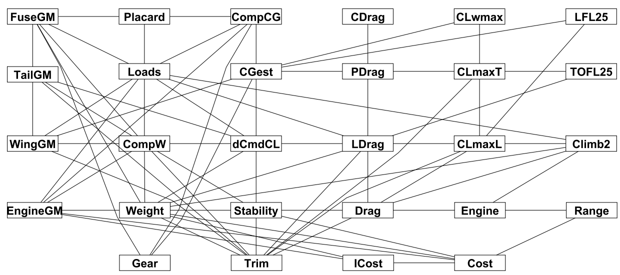



**Figure 1.3:** Conceptual design: disciplines connections [7].

Referring to aircraft design, as shown in Fig. 1.3, several modules are involved and multiple are the dependencies between them, but making use of the *monolithic design* it could be not possible to improve or modify and test one single module without running the whole system.
The main characteristics of *monolithic design* are [8]:

- Functionality implemented by part of the system cannot be reused without using the entire system
- Initialization of the system may be tricky or laborious
- Change to the control flow is impossible
- The only escape from *monolithic design* is to spend months refactoring and rewriting the system into independent modules

These abovementioned features are due to [8]:

- Procrastination of refactoring
- Premature optimization, especially a tendency to performance perfectionism or Puritanism
- Not writing for reuse
- Tunnel vision or attachment that limits your vision to one architecture, one flow paradigm, one memory management technique, etc

Possible approaches to prevent it are [8]:

- Not immediately think about the code perfect for one environment, since it will be difficult to change and tolerate another environment
- Examine the code and eliminate the most part of dependencies
- Take advantage of opportunities to work with a variety of paradigms and techniques

Thanks to these devices is possible to change from strongly dependent, not re-usable, time-consuming, possible to run only inside the whole process modules to a better and faster design approach characterized by mutually interacting independent modules.

## 1.3 Collaborative design

As explained in Sec. 1.1 and Sec. 1.2, conceptual and preliminary design phase are characterized by complex and time-consuming processes, so changing the way of thinking with respect to *monolithic software* it is an increasingly crucial need. In particular, to manage all involved disciplines, characterized by different decisions, analyses, methods and people, the only possibility is to build a system in which the product is designed thanks to collective efforts of experts with different background, moving to a *collaborative design* approach.
This one is typified by various participants, by each team of experts, capable to give their contribution facing design issues which concern their domain; in that way, many ideas and proposal could be submitted and evaluated. In Fig. 1.4 is shown a typical structure made of several cross-dependencies among the teams or design sub-spaces.

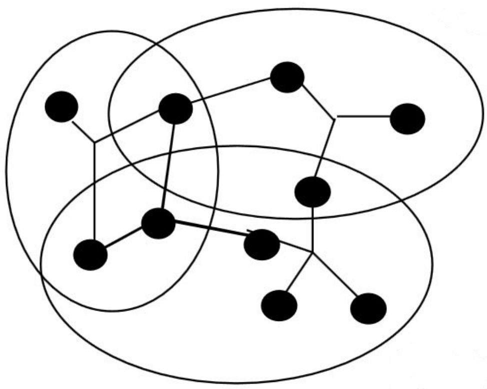


**Figure 1.4:** Collaborative design structure [9].

The small black circles represent design issues, the links between these ones represent design issue inter-dependencies, and the large ovals represent the design subspace associated with each design participant. It could be useful to underline that the design of commercial jet means may be millions of components and design issues, hundreds to thousands of participants, working on hundreds of distinct design subspaces, all collaborating to produce a complete design [9]. To face this amount of work in efficient and effective way, it is fundamental to distribute the design tasks to one or more people organized in teams; each of these ones should create one or more tools related to their own scientific domain such as aerodynamics, structures, costs and so on.
The innovative aspect lies in the possibility that teams with experience in a

specific area can use tools created by other teams to obtain output data usable as input data for their own tool. Following this approach, it is possible to talk about *collaborative* and *distributed* design. *Collaborative* because everyone within the system can admit to other teams' products; *distributed* because tasks and tools creation are assigned to peculiar and different teams.
The last evolution of this way to proceed is the *Collaborative Remote Design*. The main improvement is that teams can be geographically located in different parts of the world, such as Research Center, Universities or Companies, and can communicate and exchange their own tools, methods or results through a remote server connection or through a specific server thorough which tools are externally available.
This approach is the base for *Multidisciplinary Design Optimization* applications.

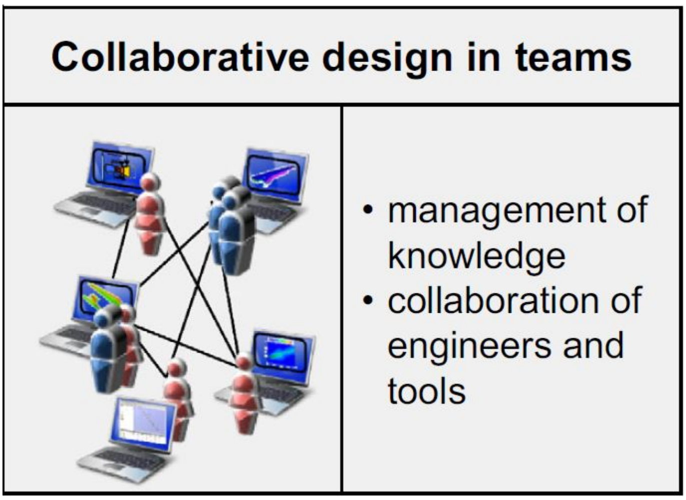



**Figure 1.5:** Collaborative design example [10].

## 1.4 *MDO*

A possible definition of MDO is contained in the *Preface* of the book entitled *"Multidisciplinary Design Optimization: State of the Art"* [11]; so : *" MDO may be defined as a methodology together with a set of tools for assisting in the design of complex coupled system, that is, systems whose behavior is governed by many distinct but interacting physical phenomena. Aircraft, spacecraft, automobiles, and engines are examples of such complex systems comprised of mutually interacting subsystems."*
The difference between conventional multidisciplinary optimization and the so-called *MDO* lies in two main characteristics [11]:

- it doesn't concern the entire design but is a complex of tools and methods that produce a complicated process that will never be completely automated.

- The potential change of one or more design variables in the subsystems will spread throughout the system; in this way the answer to the enduring question *"what if ?"* is indisputable and design cycle time is reduced.

Indeed, *MDO* incorporates various disciplines simultaneously such as aerodynamics, structural analyses, propulsion, control theory, economics and uses optimization methods to solve design problems; the innovative idea is to find the optimum by optimizing each discipline sequentially and exciting the interactions between teams. So, in this way, the optimum of the simultaneous problem is superior than the optimum of each team.
This methodology can be used in various design field, for instance in Large-Scale design such as aircraft design that involves many thousands of variables and hundreds of analyses and disciplines.
There are different levels of *MDO* in terms of how many disciplines, computer codes and teams are involved.
As reported in *" Multidisciplinary aerospace design optimization: survey of recent developments"* [15] paper, it is possible to identify three different classes of approaches to *MDO* problems.

1. Few disciplines, two or three, are involved and interact simultaneously such as aeroelasticity, structures, control system. In these cases, the designers work with a single computer code to reduce the difficulties and complexity.

2. The second class is characterized by the simplicity of the analysis tools and the interactions of various disciplines to carry out aircraft conceptual design phase. For instance, the *ACSYNT* [16] program.

3. The third category concerns all programs that aspire to contribute to design challenges and so complex computer codes and approximation techniques are involved to reduce the computational challenges.

In general the most difficult part of the whole process is the problem formulation in which the designers have to choose the design variables, such as wing span or thickness of ribs, constrains, such as the bounds on the validity of the analysis models, objectives or objective functions, such as find the minimum of a function that represent the aircraft weight, models of the disciplines, such as regression analysis applied to specific parameter.
Obviously to face the optimization of wing or fuselage shape, or in fluid flow control field or other kind of similar tasks, different optimization methods can be used:

1. Gradient-based methods
   - Adjoint equation
   - Newton's method
   - Steepest descent
   - Conjugate gradient
   - Sequential quadratic programming

2. Gradient-free methods
   - Hooke-Jeeves pattern search
   - Nelder-Mead method

3. Population-based methods
   - Genetic Algorithm (GA)
   - Memetic algorithm
   - Particle Swarm Optimization (PSO)
   - Harmony search

4. Other methods
   - Random search
   - Grid search
   - Simulated annealing
   - Direct search

- Response Surface Methodology (RSM)

Different methods can lead to different solutions achieved in a longer or shorter time and can be suited for specific application. For example, GA is a population-based optimization technique algorithms inspired by the principles of natural evolution of species through natural selection usually made of two processes. The first one is related to the selection of individual which will start the next generation and second one is the manipulation of the selected individual to form the next generation by crossover and mutation techniques. The more is the individual's adaptation the better is the individual and its chance of being parent. The process stops when the condition required by the user in terms of fitness function value is satisfied [12].

PSO is similar to GA in the initialization phase where the system is initialized by creating a population of random individuals (solutions), but each individual is characterized by its own velocity and position in the problem space. These quantities are changed at each time step according to the previous experience of each particle until the required criterion related to a certain fitness function is met [13].PSO is different than GA because it does not make use of mutation and evolution and keeps the same particles for the entire run but velocity and position of each one are also influenced by the experience of other particles.

Although both GA and PSO are part of the same optimization category, GA presents high global searching ability but is characterized by a poor computation efficiency and poor optimization speed compared to Particle Swarm optimization technique and the convergence is not guaranteed. On the other side, PSO presents good advantages in convergence speed, in finding global optimal and has a simplest implementation than GA, but sometimes can converge prematurely and can be trapped into a local minimal especially with complex problems [14].

As can be noticed, tackling an optimization problem means conduct a careful analysis of pros and cons of the available methodologies also considering how many variables are involved and that computational time spent to reach the convergence is considerable as more as the complexity increases. So, developing new optimization techniques, being confident in choosing the specific low or high-fidelity methodology suited for each task and decomposing the whole process in multiple sub-processes could be the way to significantly reduce the time spent in performing analysis and optimization design tasks.
Indeed, nowadays the computational routines about each discipline are not executed serially but the whole problem is broken down in sub-problems that work in parallel way as depict in the Fig. 1.6.

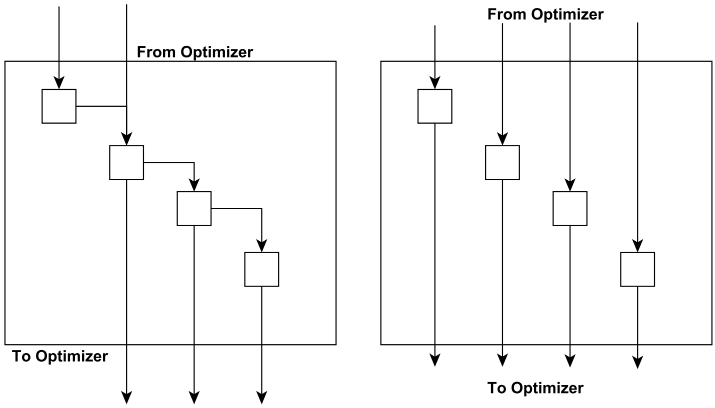



**Figure 1.6:** Problem decomposition: from serial to parallel iteration [7].

Despite multiple optimization methodologies can be applied to carry out an optimization task, the same one can be solved by using different *architecture* or *problem formulation* improving the system efficiency. The organization of the discipline models and optimization software must be suited with the problem formulation in order to achieve an optimal design. In particular, as deeply explained by Martins and Lambe [17], a MDO problem that can take advantage of pre-existing discipline tools can be performed by means of IDF (Individual Discipline Feasible), where all the coupling variables (output) are implicit functions of the design variables or MDF (Multidisciplinary Feasible), where more coupled disciplines are involved and solved in turn reaching the analysis convergence and then the optimization process can start. The two architectures are represented in Fig. 1.7 and Fig. 1.8 respectively.

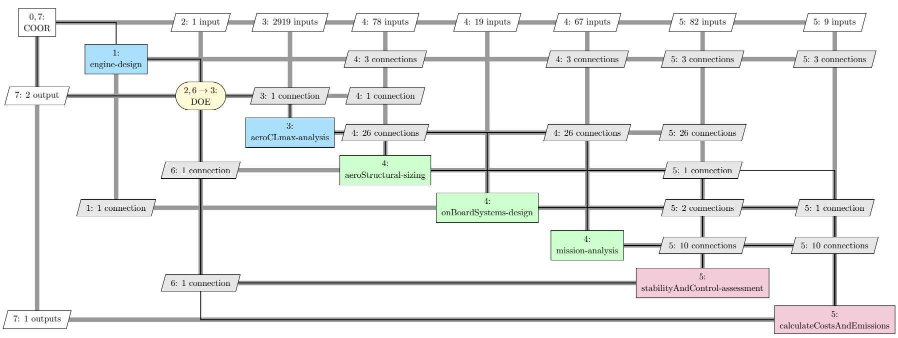



**Figure 1.7:** MDO problem formulation: IDF architecture

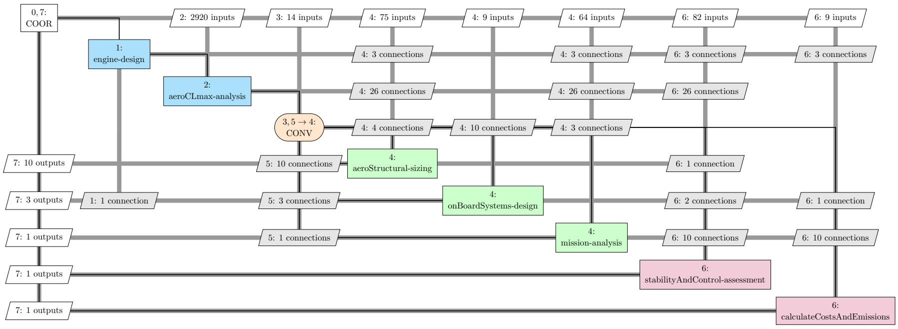



**Figure 1.8:** MDO problem formulation: MDF architecture

The abovementioned architecture can be employed both for single and for multiple optimization problems. The latter case concern situations where a complex problem can be decomposed in a set of subproblems which can be optimized separately. In this case it is possible to speak about *distributed* MDO architectures [17]. Examples of distributed IDF architecture can be the Collaborative Optimization (CO) or the Quasiseparable Decomposition (QSD) methods, while MDO of Independent Subspaces (MDOIS) or the Asymmetric Subspace Optimization (ASO) are labeled as distributed MDF problem formulations. A concrete application was proposed in [18] where an aircraft MDO focused on high-fidelity aerostructural optimization was performed. In particular, an aerodynamic shape optimization framework was assembled involving several disciplines such as aerodynamics, mesh and geometry model creator and structures and the optimization loop was managed thank to a gradient-based optimization algorithm in order to make a decision on a new set of hundreds or thousands design variables. In this case, a complex problem is split in subproblems addressed to tool or algorithms corresponding to a specific discipline. In this scenario a distributed architecture was employed approaching to the complex problem by decomposing it in a set of separate problems which need to be solved or optimized in order to reach the optimization of the entire complex problem in a more efficient way. Another significant way to proceed has been proposed in [19]. Here, the authors deal with a complex MDAO problem involving a large number of disciplinary tools with thousands of cross-connections and characterized by a different level of fidelity. The main target is formulating a complex MDAO problem by means of theoretic graph approach starting from a chain composed by all the disciplines to a chain obtained by keeping only the disciplines necessary to solve a specific design problem. The described procedure is one of the procedures chosen to provide a possible response to the MDAO community that is deeply interested in learn how to approach in formalizing and solving such complex problems.

However, reaching an optimal design means making suited choices in terms of objective functions, constraints, design and state variables and MDO architecture targeting to the most efficient compromise. Matching this goal is not trivial, so the following survey of requirements, proposed by *G. La Rocca* in *Knowledge Based Engineering Techniques to Support Aircraft Design and Optimization* [20], should be followed to achieve a good result.

**Overall system architecture**

1. The system should have a *loosely coupled* modular structure to adapt, i.e. allowing reconfiguration and scalability, to different design cases and to the specific needs of the various design process phases and it should be able to support *closely coupled* analysis to fulfill high computation speed requirements.

2. The system should be able to integrate both of-the shelf and in house developed design, analysis and optimization tools, as well as data sharing and communication systems.

3. The data exchange among the various MDO system components should be based on standard data representation formats.

4. The system should support automation of all the repetitive activities related to the iterative nature of the MDO approach. This means to include pre-processing of data and models as required to feed different design and analysis tools, post-processing of the data generated by the various design and analysis tools and the transfer and storage of data between the various design and analysis tools.

5. The system should make use of dedicated software frameworks for the integration of the various analysis and design tools involved, i.e., to support process coordination and communication among the various design, analysis and optimization tools.

**Analysis capability**

1. The system should not have any limit on the number of disciplines that can be integrated.

2. The system should allow the use of analysis tools with different levels of fidelity, with the possibility to switch level (possibly automatically, based on the results of some accuracy sensitivity analysis).

**Geometry modeling**

1. The geometry model should support the use of both low and high fidelity analysis tools.

2. The geometry modeling system should not constrain the user to conventional aircraft configuration.

3. The geometry model should support the level of automation and robustness required for the use in an MDO framework.
4. The parameterized geometry description should be compatible with current CAD systems and transferable through standard data exchange formats Optimization.
5. It should be possible to provide designers with visualizations of the design space and not only with single optimum points, to facilitate them judging the robustness and the sensitivity of the reached design point.

In the past, numerous projects moved to this direction to propose an innovative product requesting a deep work in terms of research and MDO applications; one of the most significant is the MOB project [21], focused on developing a multidisciplinary optimization process for blended wing body configurations.

Within this project analysis modules of high fidelity, which were interlinked in a distributed network of computers, have been used. A step in the direction proposed by [20] is represented by the work discharged by DLR's team [22]. This application was focused on developing a distributed collaborative MDO environment by using a multi-level approach combining a high-fidelity gradient-free and a gradient-based optimization with conceptual aircraft design methodologies to minimize the fuel burn and optimize the aerostructural aircraft behaviour.

Each discipline's tasks were solved by a set of tools stored in a sort of black-boxes part of the whole complex MDO chain, the efficient data exchanging was ensured by using the CPACS schema and the chosen architecture,a distributed MDF, was implemented in RCE. A further step could be creating a cluster for each discipline and producing a response surface for each relevant variable in the optimization process. Then, making use of the RSM, it could be possible to query each response surface through a certain interpolation methodology speeding up the entire optimization flow.

The way of working proposed in [22] is the same that AGILE project wants to continue and improve working on design and optimization tasks focusing on the development of techniques suited for teams distributed all over the world enabling the remote collaboration.

In Sec. 1.5 *AGILE* project will be explained in detail.

## 1.5 European project: *AGILE* overview

AGILE (*Aircraft 3rd Generation MDO for Innovative Collaboration of Heterogeneous Teams of Experts*) is an EU funded project under the research schema Horizon 2020 and coordinated by the DLR. Within AGILE, a team of 19 industry, research and academia partners from Europe, Canada and Russia are collaborating together to develop the next generation of aircraft Multidisciplinary Design and Optimization processes, which targets significant reductions in aircraft development costs and time to market, leading to cheaper and greener aircraft solutions.

The proposal is to introduce and create a new MDO aircraft framework promoting a new approach in terms of collaborative design, knowledge dissemination among various teams of experts and innovative MDO approaches and applications. Implementing such a complex system means making several assumptions such as how things should be done and how to use different existing technologies. This set of assumptions and technologies is labeled the 'AGILE Paradigm' of which the two enabling layers are the *Collaborative Architecture* (CA) and the *Knowledge Architecture* (KA) as depicted in Fig. 1.9.

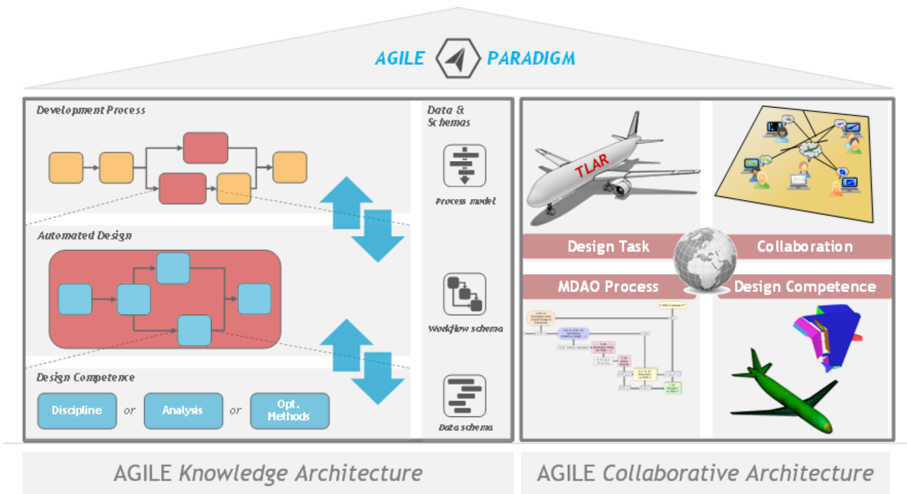



**Figure 1.9:** AGILE Paradigm.

The first one (Collaborative Architecture) targets the development of the technologies enabling distributed collaboration, comprising the process of collaboration between involved specialists, collaborative pre- and post-processing, visualization and the 80 enhancement of existing framework. The second enabling layer (Knowledge enabled technologies) develops the information technologies, which support the management and the formalization of knowledge within an MDO process. Both CA and KA are essential to support large, heterogeneous teams of experts in performing collaborative development in a efficient and time-effective way.

As reported in [3], four are the main technical objectives of AGILE project:

1. The development of advanced multidisciplinary optimization techniques and their integration, reducing the convergence time in aircraft optimization to face the lack of knowledge about how optimization workflows involving a lot of disciplines
2. The development of processes and techniques for efficient multi-site collaboration in the overall design teams, since there are a lot of tools of specific disciplines and the results are hard to interpret without specialists; so the need of collaboration
3. The development of knowledge enabled information technologies to support interdisciplinary design campaigns
4. Develop and publish an *Open MDO Test Suite*, allowing the access to the project technologies by other research activities, and providing a reference database for future aircraft configurations research

Reaching these objectives means to bring a great improvement concerning the aircraft design and optimization process achieving a

- 20% reduction of time needed to converge the optimization of an aircraft configuration thanks to AGILE optimization techniques.
- 40% reduction of time needed to solve an MDO problem in a heterogeneous team of specialists thanks to AGILE collaboration processes and AGILE optimization techniques

As mentioned in Sec. 1.4, building an MDO system means to build up a complex structure made of some main instruments. In particular, three of these ones are crucial: the **framework** to control the entire process and optimization tasks, a **common standard file** to exachange aircraft data and tools information, **several analysis modules**. With regard to the *framework* and the *common standard file*, they have been both developed and provided by DLR to the consortium and named RCE and CPACS respectively. A detailed description is provided in Sec. 2.4.

AGILE started in June 2015 and finished in November 2018 for a total duration of 3 years and a six months extension. The project is structured into three sequential phases, distinguished by three design campaigns with increasing levels of complexity, addressing different aircraft configurations and suited MDO techniques. The whole structure is shown in Fig. 1.10.

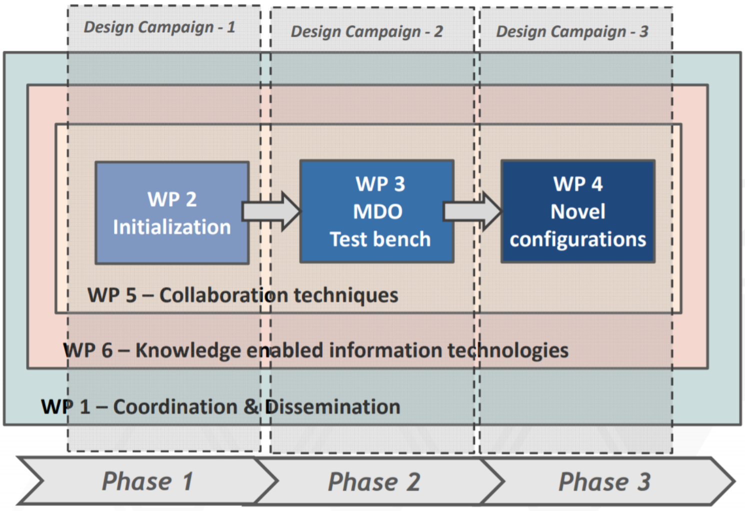



**Figure 1.10:** AGILE overall structures. [23]

The first stage, the **Inizialization** phase, focused on optimizing a reference conventional aircraft configuration using the state-of-the-art techniques provided by the consortium.

Studies and MDO analyses were performed on the reference use case and results and best practice were then used to examine and benchmark novel optimization techniques individually and later in smart combinations (MDO test bench). Finally, the most successful methods and approaches were applied to significantly different and disruptive aircraft configurations (Novel Configurations).

The entire project structure is made of six work packages. The three sequential WPs are included within two enabling layers. The first one (Collaboration techniques) is focused on the development of the technologies enabling distributed collaboration, including the process of collaboration between involved specialists, collaborative pre- and post-processing, visualization and the enhancement of existing framework. The second one (Knowledge enabled technologies) aimed to develop the information technologies, in order to support the management and the formalization of knowledge within an MDO process.
Each of the sequential design campaigns, which also comprise some parallel activities, focused on the solution of the use cases, which are setup to develop specific collaborative and knowledge-based technologies. Sequential DCs were characterized by an increasing complexity both as regards use case perspective, progressing from conventional aircraft to novel configurations, and MDO environment perspective, from the current state-of-the-art MDO system to the 3rd generation system. Thereby, the most of the DC-1, was spent deploying SOTA techniques and distributed MDO system, using pre-existing analysis codes, interfaces, framework and optimization software. Subsequently, an optimization

process, according to today's best-practice methods, was formulated to optimize a large regional jet aircraft designed starting from TLAR assigned to the consortium by Bombardier as AGILE partner.

During DC-2, as consequence of DC-1 achievements, the "MDO Test Bench" activities was performed aiming at quantifying the benefits of different optimization techniques and their combinations for large scale optimization problems. Methodological improvements and applications to specific use cases were achieved also considering extension to the SOTA of MDO approach implemented in DC-1. In order to obtain a process characterized by a more rapid convergence to the best solution, some investigations were performed on the capabilities and differences among all the optimization approaches. Finally, within DC-3, all improvements and the best and suited optimization approaches were applied to design and optimize seven different novel and disruptive aircraft configurations such as *Blended Wing Body, Box Wing, Strut-braced, Medium-Altitude-Long-Endurance (MALE), Wing and Rear Mounted turboprop engine.*

The proposal is not only to integrate **more disciplines**, **higher fidelity** and **certification constraints** in the optimization process, but also to reach a European transport system that is resource-efficient, climate- and environmentally-friendly, safe and seamless for the benefit of all citizens, the economy and society through the respect of these needs:

- resource efficient transport that respects the environment
- better mobility, less congestion, more safety and security
- global leadership for the European transport industry
- socio-economic and behavioural research and forward-looking activities for policy making

Gaining these targets means to introduce and spread knowledge and skills through the setup of the $3^{rd}$ MDO generation characterized by team of experts which can perform each specific analysis and interpret and disseminate technical results. These characteristics make the difference with respect to previous MDO generation in which a single supervisor was responsible for tools and related analyses. In Fig. 1.11 the first MDO generation approach is presented. Here a single person oversees the entire process, several analyses are performed in the same algorithm and corresponding results flow into a huge database and convergence loop results are then used to perform the optimization task. In Fig. 1.12 a second MDO generation is proposed. The difference with respect to the first generation lies in using a central data model (CPACS) as data exchange platform and each analysis is performed by a dedicated computer code. The $3^{rd}$ MDO generation shown in Fig. 1.13 perfectly embody the concept of remote, distributed and collaborative MDO. In this scenario there is a single supervisor which is responsible for the optimization process, but each discipline is entrusted to a certain specialist which is responsible for its own competence made available in the optimization framework. In this way the specialists can check their algorithm and the results provided to other specialists guaranteeing a higher level of reliability to the entire system than the one ensured by the previous MDO generations.

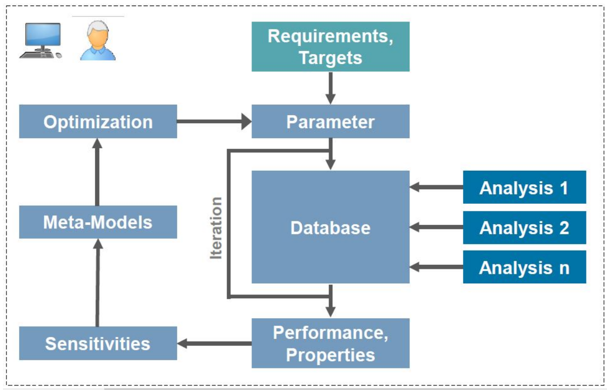



**Figure 1.11:** First MDO generation [24].

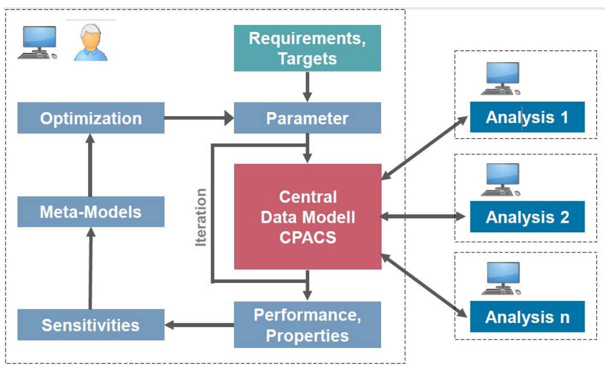



**Figure 1.12:** Second MDO generation [24].

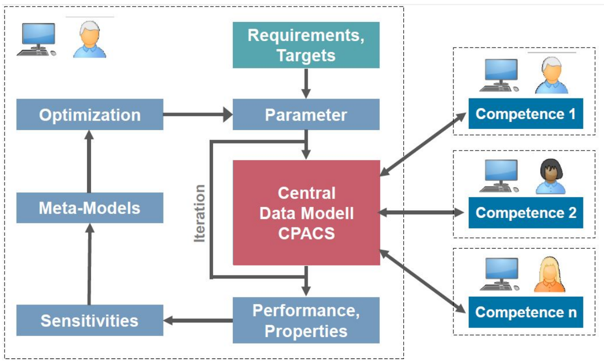



**Figure 1.13:** Third MDO generation [24].

## Synthesis of the chapter

- Traditional aircraft design steps have been presented.
- Obstacles and drawbacks of the *monolithic* design approach have been emphasized.
- Aircraft design procedure needs to be changed moving towards a collaborative approach fostering the exchange of knowledge among teams and people
- MDO approach could be the way of facing aircraft design activities limitations. A complex design problem can be divided in multiple sub-problems, each addressed to a specific discipline, and an MDO problem strategy must be formulated to reach the solution efficiently.
- AGILE project is a reliable solution to make a step forward in MDO state-of-the-art by creating the next generation of Multidisciplinary Design Analysis and Optimization approach. The main goals of the project are:
  - The development of advanced multidisciplinary optimization techniques and their integration in a complex framework.
  - The development of processes and techniques for efficient multi-site collaboration in the overall design teams.
  - The development of knowledge enabled information technologies to support interdisciplinary design campaigns.
  - To develop and publish an Open MDO Test Suite allowing the access to the project technologies by other research activities.
  - To reduce by 20% the convergence time needed for aircraft optimization tasks.
  - To reduce by 40% the time needed to solve an MDO problem in a heterogeneous team of specialists thanks to AGILE technologies and approach.

# Chapter 2

# Instruments for Aircraft $3^{rd}$ MDO Generation

## Introduction

One of the main achievements of the AGILE paradigm, is the possibility to distribute the workload among partners worldwide distributed allowing the availability of the most up-to-date and state of the art competences within a MDO process.

Competences can be developed and integrated with different level of fidelity, different environment, different programming languages, locally or remotely executed, enabling specialists to be focused on their own tasks.

In this context and embracing this way of working, the author, as part of UniNa partner, gave a great contribution by developing and providing to the consortium several tools related to several disciplines created using various programming language such as JAVA, Python and Matlab and taking advantage of CFD database. These tools and other specialists' tools were then assembled through a dedicated platform developed during the project.

Each of the developed tools was made available to other partners through a remote request. A partner can execute a toolchain where author's tool performs specific calculations and the author can run his own tool on his machine and share the results with the partner which made the request. The paradigm is well described in Fig. 2.2, Fig. 2.3 and Fig. 2.4. In particular, the main workflow labeled as *Master Workflow* is managed by a single supervisor which sends a request to one or more specialists querying for a remote service provided by running the sub-workflows (*Slave*). The results obtained are then sent again to the *Master Workflow* where all the results are collected in a single CPACS file.

In this chapter basic instrument to develop the aircraft $3^{rd}$ MDO generation are presented.

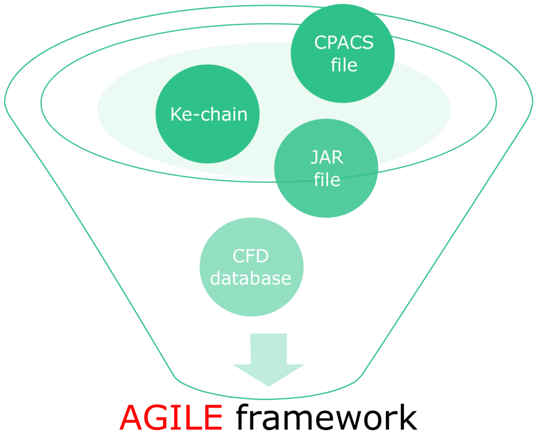



**Figure 2.1:** Useful software packages and files for *UniNa* tools developed by the candidate.

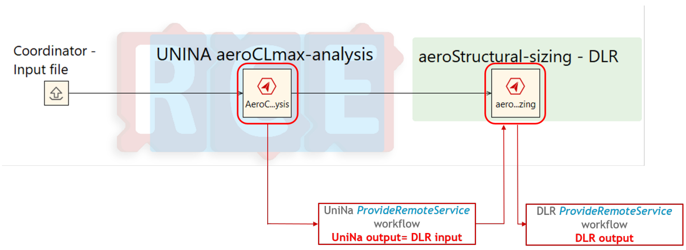



**Figure 2.2:** *Master Workflow* implementation involving two partners.

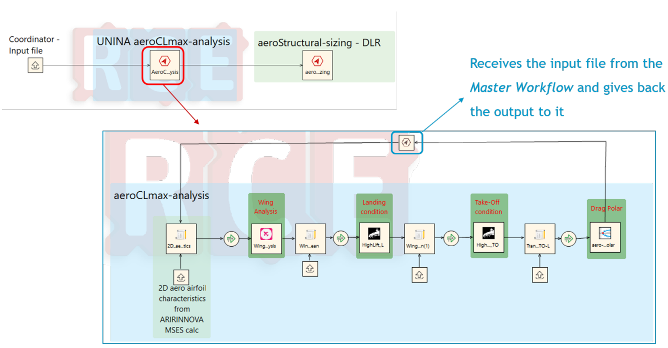



**Figure 2.3:** *Slave Workflow* assembled by the author concerning aerodynamic calculation.

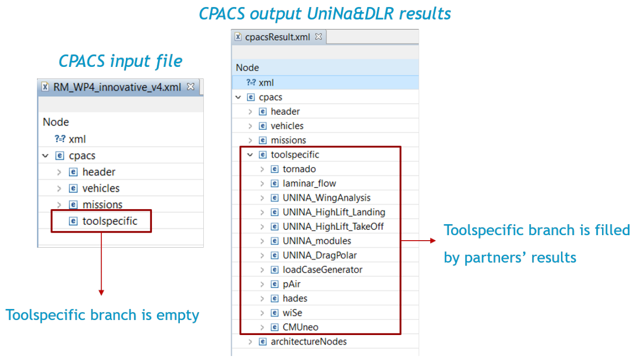



**Figure 2.4:** Results coming from partners are stored in the *toolspecific* tag within the CPACS file.

## 2.1 CPACS standard format

The ***C****ommon* ***P****arametric* ***A****ircraft* ***C****onfiguration* ***S****chema* is an open source project and has been created by DLR since 2005 [25–27]. It is characterized by a standard schema and it is based on XML technology; so, every kind of data or contents are positioned underneath a specific tag and the amount of these assemble a typical XML tree in which the root element is named *'/cpacs'*.

The CPACS can contain a parametric description of one or more aircraft configurations in terms of geometry, mission, airport, engine performance and so on as shown in Fig. 2.5 and Fig. 2.6.

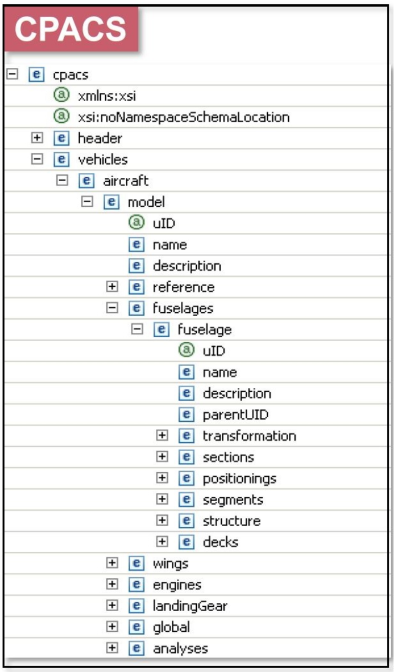



**Figure 2.5:** CPACS schema [25].

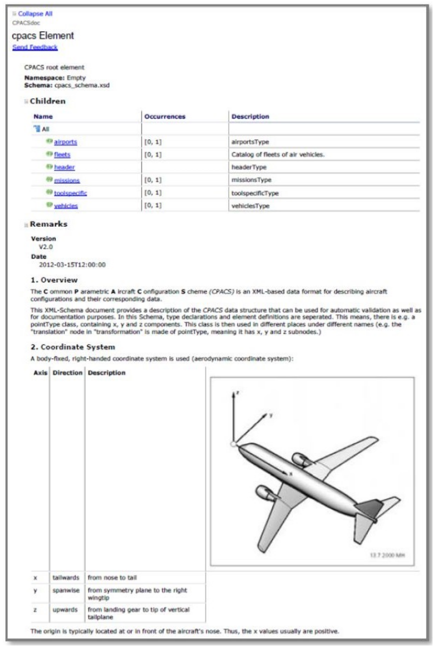



**Figure 2.6:** CPACS documentation [25].

This file is not just a trivial data storage but it is useful for documentation and it is fundamental to exchange information between partners specialized in each different discipline. In particular, it contains [28]:

1. Only “exchangeable” information
2. No redundant information
3. Product information
   - Aircraft (geometry, aerodynamic...)
   - Airport
   - Mission
4. Toolspecific information
   - Options
   - Runtime information
   - Partner tool results

- Placeholder

In this way data exchanging between partners is assisted and everyone have to interact to each other only through CPACS file either adding a calculation under a specific tag, for example *mission* or *aircraft*, or writing its own results in the *toolspecific* section. A *description* tag can be added to help other specialists to understand in details output data coming from other disciplines. Furthermore, a consortium of a European project is commonly made out of three or more partners and so the number of interactions to exchange data between them could be prohibitive. Using the CPACS technology, the number of interactions is strongly reduced because there are no more *partner-to-partner* relation but only *partner-CPACS-partner* one; so, as shown in Fig. 2.7, it is possible to reduce connections amount from *n(n-1)* to *2n*, where *'n'* is the number of entities (partners/tools) that want to interact.

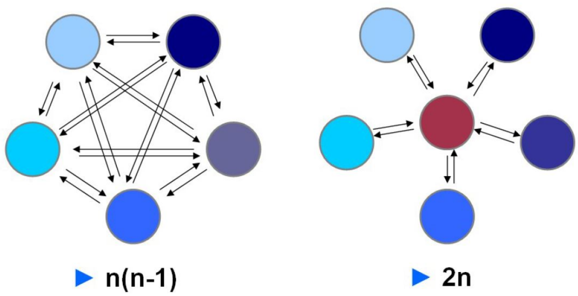



**Figure 2.7:** Number of interactions [25].

The toolspecific data can be transferred along with the dataset and carries further information.So the interaction among the analysis modules is sequential and, in addition, the framework allows for splitting and/or merging CPACS datasets. For instance is possible to add the toolspecific section that contains characteristic input useful to use the module that you want run and after is possible to create a new toolspecific dataset with the module results and added these ones to the CPACS that will be create after the run. The following figure explain this concept. The mapping file is used to choose the CPACS file source sections to add in a specific position of the CPACS results file (target), for instance under the toolspecific tag.

It is possible to understand that this technology can be used to assist the optimization and the Design of Experiments and so to foster the MDO applications, because the results inside the file may come from different disciplines, for each of these ones from high or low fidelity analysis level applied to the whole aircraft or to a single component as show in Fig. 2.8 [25].

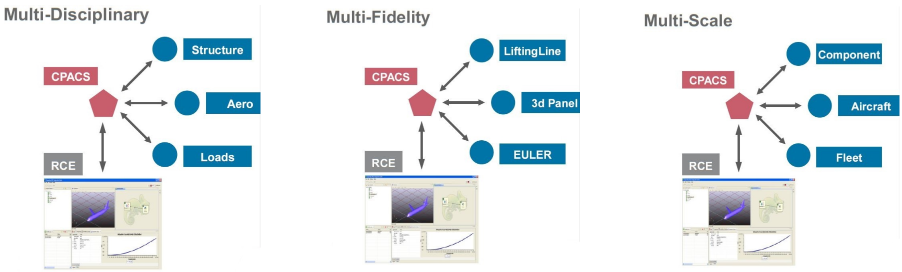



**Figure 2.8:** CPACS file capabilities.

For example, under the tag *'aeroPerformanceMap'* is possible to find vectors with Mach and Reynolds number combined to provide array for force and moment coefficients for different attack and yaw angles. The same structure there is for engine analysis module thanks to global engine input parameters, flight altitude, mach number and thrust to provide a performance map for the thrust-specific fuel consumption [25]. In this way the need of automation is satisfied allowing to reduce time and costs in the conceptual design phase. In Fig. 2.9 a typical CPACS structure is shown.

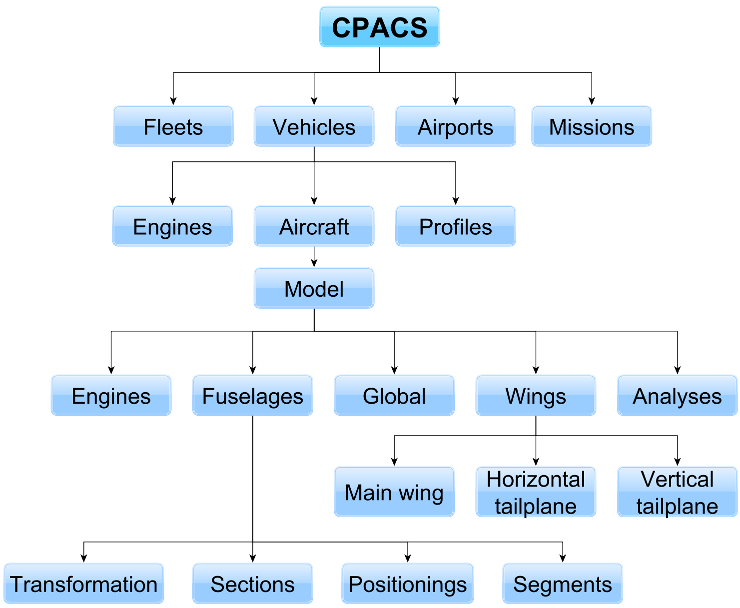



**Figure 2.9:** General CPACS structure.

## 2.2 RCE

The **R**emote **C**omponent **E**nvironment is an open-source software developed by the German Aerospace Center (DLR) to help engineers or teams of scientists to manage, run, control complex analysis and simulations [29]. Other PIDO (Process Integration and Design Optimization) framework characterized by similar capabilities are also available such as OPTIMUS [30] or OpenMDAO [31]. The first one, developed by Noesis Solutions, is an industry-leading software platform which enables the communication with any engineering software and with any file syntax allowing the implementation of several MDO architecture also suited for real-world problems. The second one is an open-source MDO framework which implements the state-of-the-art algorithms to solve coupled models achieving high computational efficiency.

Actually, the DLR provided RCE to all the partners of the AGILE consortium giving also support for any issue or software updates. Furthermore, the software is user-friendly, open-source and fully compliant with CPACS technology. These advantages led UniNa, and so the author, to choose RCE as the suited PIDO for the activity to carry out over the project. In RCE environment the components of a specific run are represented by default tool, like *'Optimizer'* or *'Parametric Study'*, combined with tools manually integrated tools by partners and connected each other thanks to data flows instruments available in RCE environment.

A set of default and/or integrated tools linked each other to accomplish a specific task, is called *Workflow* and each component is like a 'black box' that receives one or more data (float element, files, short text,...) in input and provides one or more outputs. In Fig. 2.10 a *Workflow* example in RCE environment is depicted.

Obviously, following the best practice means to know the input and output type of data and the right XML path in order to avoid errors coming from a unit mismatch or missing path in the CPACS file.

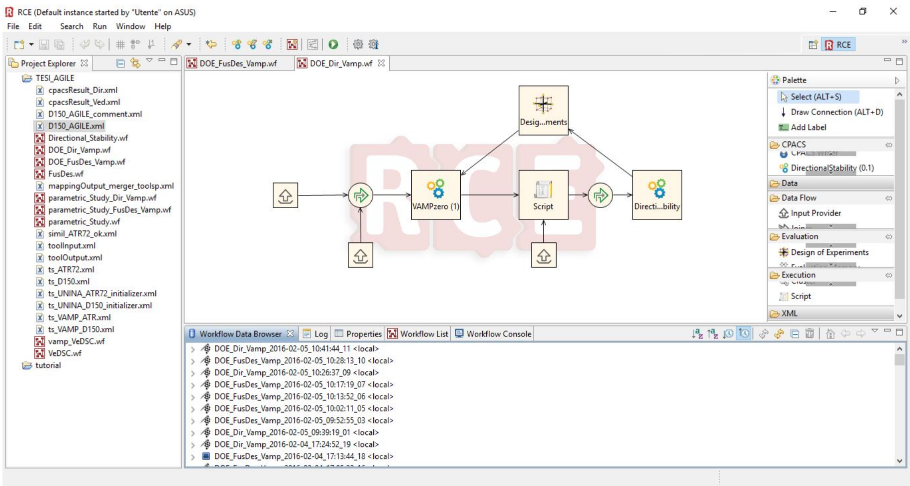



**Figure 2.10:** RCE workbench.

Referring to Fig. 2.10 it is possible to provide a detailed description of RCE. Left hand-side:

- *Project Explorer*: contains the project folders in which it is possible to access to saved workflows (file '.wf') and other types of files such as CPACS, text files and so on. From here it is possible to open, read and edit files and workflows, within RCE, selecting it from a specific project folder.

Right hand-side:

- *Palette list*: components can be default ones like *'Draw Connection'*, *'Input Provider'* in **Data Flow** section, *'DoE'* in **Evaluation** section or *'Script'* in **Execution** section. In **CPACS** section, manually integrated CPACS tools can be chosen or modules shared on the cloud by partners can be used through remote connection access.

Center:

- *Workflow Editor*: is the main view in which is possible to assemble a simple or complex workflow using different components. This panel allows to access to the component properties and to choose the needed files, such as CPACS input file, *'mappingInput.xml'* and *'mappingOutput.xml'* to select the desired CPACS path, to run the workflow. Furthermore, it is required to connect input and output variables or files of each component and/or tool by using the *'Draw Connection'* feature.

- *Workflow Data Browser*: history collection of all finished or stopped workflows. This panel allows to check every kind of input and output of each component in the workflow following the time history. In particular, it is possible to dive into the *Timeline* or *Timeline by component* tag which contain all the actions, performed by each component, stored chronologically as shown in Fig. 2.11. In addition, it is also possible to export and save workflow results.

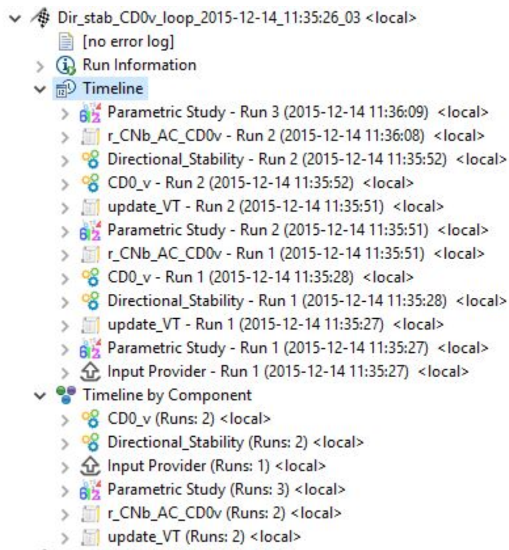



**Figure 2.11:** Workflow Data Browser

- *Properties*: workflow components properties are shown; in particular all connections between input and output data are reported, as shown in Fig. 2.12. It is also allowed to edit properties and connections from this panel directly.

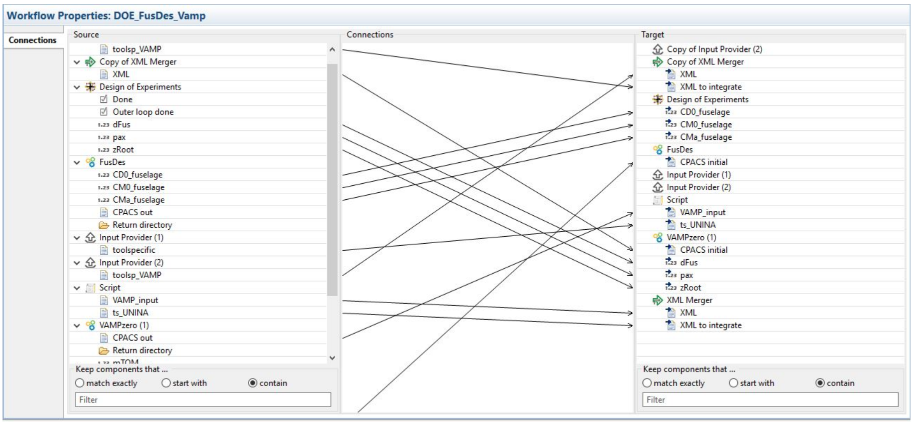



**Figure 2.12:** Workflow properties and connections

- *Workflow Console* : it allows to check workflow run step by step by showing all the details about the specific toolchain such as the current directory, the used input and output file, text, vectors or everything user wants to *'print'* through the main algorithm. In Fig. 2.13 an example is depicted.

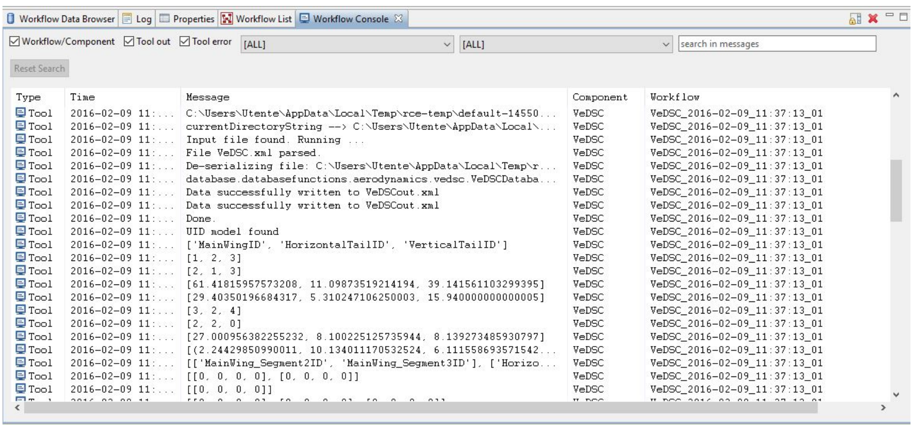



**Figure 2.13:** Workflow console

### 2.2.1 New tool creation

Within RCE framework it is possible to create a new tool pressing on *'Integrate Tool...'* button; then a wizard will open (see Fig. 2.14), and it is possible to choose among several tool configurations. Following, the best practice is described:

- Select *Create a new tool configuration from a template* and choose *CPACS Tool with return directory (Type: CPACS)* and then go *Next*.

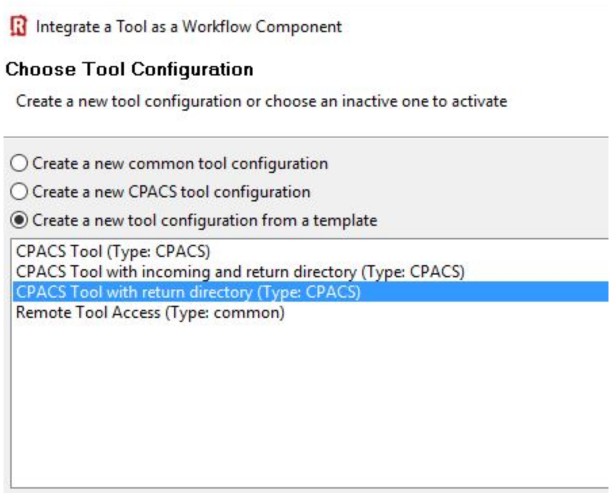



**Figure 2.14:** New CPACS tool within RCE. First step.

- Insert an arbitrary name, as shown in Fig. 2.15, and click *Next*.

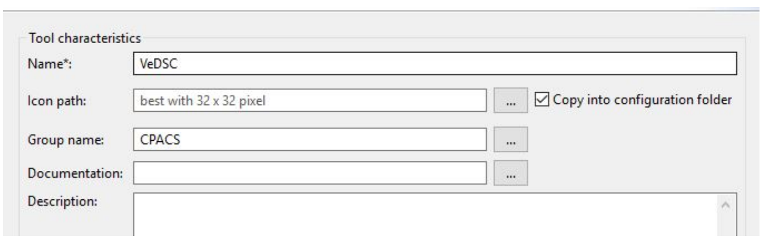



**Figure 2.15:** New CPACS tool within RCE. Second step.

- Click *Next* two more times.
- As shown in Fig. 2.16, press on *'Add'* button in the right upper corner. Then, browse to choose an arbitrary tool directory and specify a version

tool number (e.g. '0.1'). Finally, select *'Create an arbitrary directory in RCE temp directory'* and go to *'OK'*.

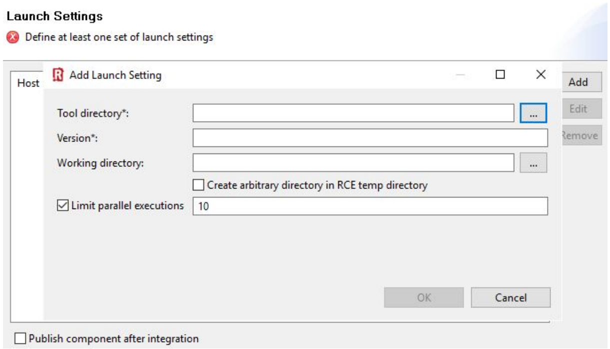



**Figure 2.16:** New CPACS tool within RCE. Third and fourth step.

- All gaps are automatically filled correctly using default filename, as illustrated in Fig. 2.17. If a file has been customized and its name is different than default one, it is necessary to browse until the tool folder and select the specific file. To uncheck *'Merge static tool specific input'* box and go *Next*.

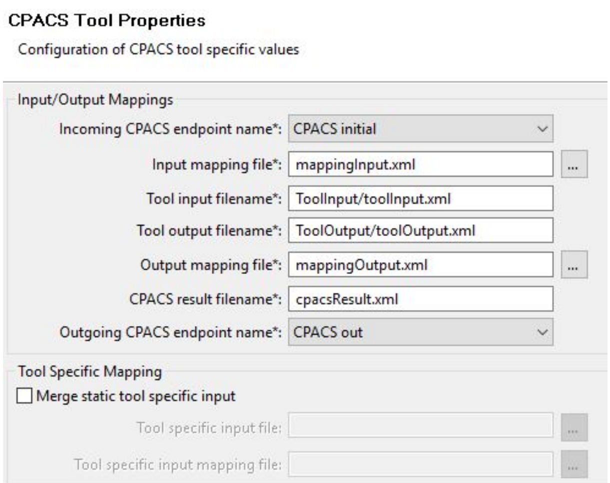



**Figure 2.17:** New CPACS tool within RCE. Fifth step.

- For *Windows OS* check *'Command(s) for Windows* box and write the type of file and the filename (e.g. 'python file_name.py') that represent the main script that allows workflow running correctly. Finally, press on *'Save*

*and activate'* button to end the process.
At this stage, the new tool is added to the *'CPACS'* sub-section folder under *Palette* section.

- **N.B.**: To avoid error arising, an empty folder named *'ReturnDirectory'* must be created within the tool folder. *'ReturnDirectory'* can be used to store results, output data, plots, files and so on.

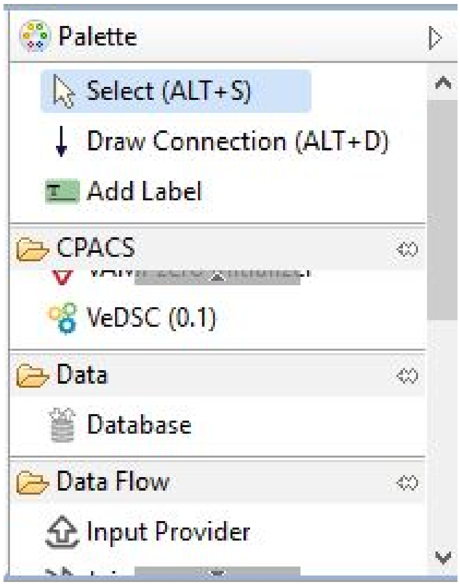



**Figure 2.18:** New CPACS tool available

Moreover, RCE allows to integrate a tool created on another machine following few steps described in Sec. 2.3.

## 2.3 Tools Integration

To integrate an already existing CPACS tool on another machine, the tool directory must be copied in the default tools directory of RCE software ( *'\..\.rce\default\integration\tools\cpacs'*). Moreover, it is also necessary to copy and paste the *'configuration.json'* file from the old machine to the new one in the same default tools directory path.
In this way RCE will automatically recognize all tool settings and will import the new tool into the *'CPACS'* sub-section under *Palette* section. Within the new folder the following files must be stored to ensure a proper way of working of the tool:

- *'python_algorithm.py'*, that is the main script
- directory called *'shared'*, that contains *'file.dll'*
- *'mapping_input_file.xml'*
- *'mapping_output_file.xml'*
- *'file.jar'*, that is the core of a certain module developed by UniNa
- *'executable_file.bat'* , a *'Windows'* executable file to run the *'file.jar'*
- *'input_file.xml'*, that contains needed input data to allows output data calculation
- *'output_file.xml'*, in which output data are stored

To finalize the integration procedure, it is necessary to point to the right path using RCE.
In particular, opening RCE homepage it is possible to find the new tool within *'Palette'* section, under *'CPACS'* sub-section. By selecting the new tool and clicking the mouse right button, *'Edit Tool'* feature mast be chosen. After that a wizard will open and the following procedure must be carried out:

- Choose a tool configuration to edit and go *'Next'*
- Go *'Next'* three more times
- Choose a new tool directory and go *'Next'*
- Select *'mapping_input'* and *'mapping_output'* files and go *'Next'*
- *'Save and update'*
- Enjoy your tool

The algorithm that allows the tool running can be coded in any programming language but paying attention to the necessity of coding in Python or Jython if internal RCE script needs to be created. A tool integrated in RCE can be run in local (user's machine) or in remote manner.

Usually, author's tools consist of a main script coded in Python language to manage all the information coming from the CPACS file and takes advantage of suited JAVA executable file in-house created to perform all the analyses. Such a tool must be compliant with any CPACS file being able to grasp all the inputs from it and to save all the outputs in specific tags contained in the CPACS file. A detailed description and some examples are described in Sec. 2.3.1.

### 2.3.1 JAVA Executable File

In order to accomplish several tasks scheduled within AGILE project, developing and implementing different aircraft preliminary design phase methodologies is of primary importance. At the same time, the possibility to operate with methodologies characterized by different levels of fidelity or to implement new semi-empirical models could ensure longevity and enriching future exploitation capabilities. Aiming at achieving the abovementioned targets, the DAF research group [34] of UniNa has been working since 2005 to the development of software and frameworks for aircraft conceptual and preliminary design stages [35]. Currently, DAF group has been involved in the development of a complex open-source Java library named Java toolchain of Programs for Aircraft Design (JPAD) [36–38], built as a modular framework. JPAD and all its executable modules are completely written in JAVA which is a class based and object-oriented programming language that ensuring modularity, code reusing through inheritance, flexibility and portability on any combination of hardware and operating system. The modularity guaranteed by JAVA language allowed the author to use just some pieces of JPAD software stored in several '.jar' (**J**ava **Ar**chive) file as the core of a tool where all the analyses are performed. It is possible by running a simple executable file. While, the implementation of the main script and functions used to deal with all the information concerning the CPACS schema and the creation of suited XML input files needed to run the '.jar' core were implemented in Python language. In Fig. 2.19 the flowchart of the operations actuated when a tool works is depicted. In particular, starting from a CPACS baseline file, a Python script is in charge of extracting all the useful information to accomplish a certain analysis and storing them in a customized input XML file to run the executable JAVA file. After the execution of the '.jar' file, an output XML file is created and the data are automatically transferred into the initial CPACS baseline which will contain all the results of the tool. In this way, at the end of a toolchain the CPACS file will store the outputs of all the tools involved in the task.

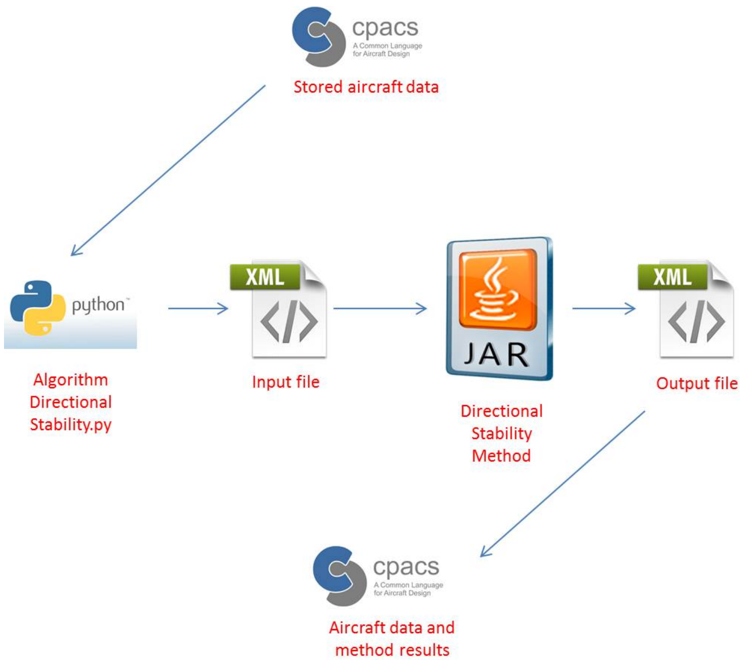



**Figure 2.19:** Operation flowchart of a author's tool.

Going deeply in details, a '.jar' file is a package file format typically used to aggregate many *JAVA* class files and associated metadata and resources (text, images, etc.) into one file to distribute application software or libraries on a JAVA platform. JAR files are archive files with '.jar' extension created using *Eclipse* platform.
Following the best practice, JAR file are obtained by encapsulating a JAVA *Main Class* that is a class which manages other dependent classes. In particular, classes can be mainly divided into *Utility Class* containing all the equations of the implemented methods and the *Calculator Class* which uses the *Utility Class* to perform all the required analyses on a specific object.
To give an example of what is the best practice followed to develop tools provided by the author in the AGILE project [39] [40], a brief description of *Directional Stability* tool is now presented. In Fig. 2.20 the *Main Class* of the Directional Stability tool is shown.

```java
public static void main(String[] args) {

    if (args.length == 0) {
        System.out.println("No input file name given. Terminating.");
        return;
    }

    String fileNameWithPathAndExt = args[0];

    // Set the folders tree
    MyConfiguration.initWorkingDirectoryTree(MyConfiguration.currentDirectoryString,
                                            MyConfiguration.inputDirectory,
                                            MyConfiguration.outputDirectory,
                                            MyConfiguration.databaseDirectory);

    // Set database name
    String veDSCDatabaseFileName = "VeDSC_database.h5";
    String fusDesDatabaseFileName = "FusDes_database.h5";

    File inputFile = new File(fileNameWithPathAndExt);
    String inputFileName = new String(inputFile.getName());

    if (!inputFile.exists()) {
        System.out.println("Input file " + fileNameWithPathAndExt + " not found! Terminating.");
        return;
    }
    if (inputFile.isDirectory()) {
        System.out.println("Input string " + fileNameWithPathAndExt + " is not a file. Terminating.");
        return;
    }

    System.out.println("Input file found. Running ...");

    String outputFileNameWithPathAndExt = MyConfiguration.getDir(FoldersEnum.OUTPUT_DIR) + File.separator +
            Files.getNameWithoutExtension(inputFileName) + "Out.xml";

    DirStabCalc.executeStandaloneDirStab(veDSCDatabaseFileName,
            fusDesDatabaseFileName,
            fileNameWithPathAndExt,
            outputFileNameWithPathAndExt);

    System.out.println("Done.");

}
```

**Figure 2.20:** *JAVA Main Class* example.

In Fig. 2.20 it is possible to notice two dependent classes:

- *MyConfiguration* where the method *initWorkingDirectory()* creates the necessary folders.
- *DirStabCalc* where, through the method *executeStandaloneDirStab()*, calculations are performed. Moreover, *DirStabCalc* class initializes the input and output file, and implements some low-level methods concerning reading and writing operations.

The module estimates the directional stability of an aircraft computing the yawing moment coefficient derivative with respect the sideslip angle, $C_{N_\beta}$, as the sum of 4 contributions due to vertical tailplane, fuselage, wing and horizontal tailplane respectively.

$$C_{N_\beta AC} = C_{N_\beta verticaltail} + C_{N_\beta fuselage} + C_{N_\beta wing} + C_{N_\beta horizontaltail}$$

The vertical tail contribution is obtained from VeDSC method [41].

$$C_{N_\beta vaerticaltail} = K_{Fv}K_{Wv}K_{Hv}C_{L_\alpha V}\left(\frac{l_v S_v}{b_w S_w}\right)$$

The fuselage contribution is estimated using at the same time the *VeDSC* [41] and *FusDes* methods [42] both implemented by the DAF Research Group.

$$C_{N_\beta fuselage} = K_{Vf}K_{Wf}K_{Hf}(C_{N_\beta Fus})$$

The wing contribution is estimated with the following equation [43].

$$\frac{C_{N_\beta}}{C_L^2} = \frac{1}{57.3}\left[\frac{1}{4\pi A} - \frac{tan\Lambda_{c/4}}{\pi A(A+4cos\Lambda_{c/4})}\left(cos\Lambda_{c/4} - \frac{A}{2} - \frac{A^2}{8cos\Lambda_{c/4}} + 6\frac{x}{\bar{c}}\frac{sin\Lambda_{c/4}}{A}\right)\right.$$

The horizontal tail contribution can be calculated following the same procedure of the wing or it can be neglected.
In order to handle these equations some dependent classes (and lots of methods) have been created. As regard the *VeDSC* and *FusDes* methods, it has been necessary manage databases as well. The databases are '.h5' files have been generated by a *Matlab* script starting from a survey of results obtained running several CFD analyses on several different aircraft geometries. Those data have then been stored in a '.h5' archive as shown in Fig. 2.21.
The module is able to interpolate the data through a low-level class *'MyInterpolatingFunction'*.

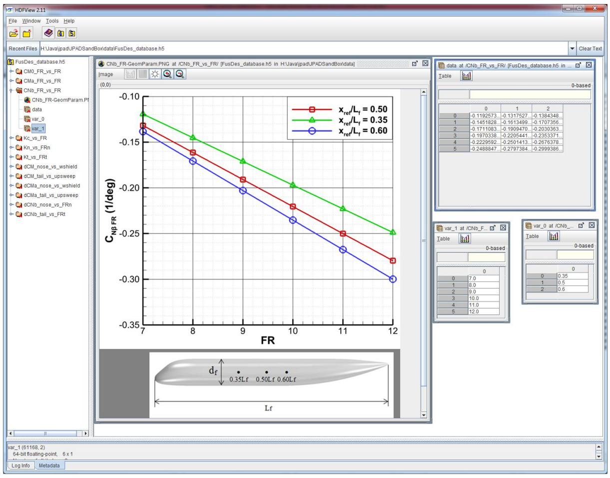



**Figure 2.21:** Database of fuselage's $CN_{\beta}$ values.

To obtain the .jar file, the following steps have to do.

1. To export the main class as .jar file. In Eclipse: File > Export> JAVA> Runnable Jar file> Runnable Jar file Specification: choose **Package required libraries into generate JAR** under **Library handling**.

2. To write a batch file, as shown in Fig. 2.22, which contains the instructions to run the .jar file.

```
setlocal
set PATH=%PATH%; .\l i b ;
java –jar DirectionalStability. jar DirectionalStability. xml
```

**Figure 2.22:** Batch file to run *Directional stability tool.*

   These three simple rows allow to add necessary libraries to the classpath (second row) and to launch the .jar file reading the input file (third row).

3. To store the input and output files, the database and the libraries in the appropriate folder. The '.h5' files are read only once thanks to initial data serialization process and then are stored in *'serializedDatabase folder'*.

Input and output files are XML files characterized by specific *tags*. The input file contains all necessary data used to perform calculations through certain equations and database, while within the output file input data are summarized and the corresponding output data are stored. Each tool has its peculiar set of files stored in specific folders automatically.

### 2.3.2 TIXI and TiGL library

Since the CPACS data format is based on XML technology, it is necessary to make use of a library capable to read, parse and edit it.
With this aim, TIXI library has been provided to the consortium as XML interface [32]. It has been developed by DLR and coded in C language, but also wrapper for C++, Fortran, Python, JAVA and Matlab has been developed. TIXI is really useful to create documents, create and delete nodes, add and remove element, attributes and so on. In this way it is possible to extract from CPACS, thanks to specific routines, every type of data such as vectors, array, boolean, integer, text or float elements. In Table 2.1 the most used routines are summarized.

| **Routine** | **Description** |
|---|---|
| open | open an XML file |
| getTextElement | Extract a text element from a specific CPACS (XML) path |
| getIntegerElement | Extract an integer element from a specific CPACS (XML) path |
| getDoubleElement | Extract a double element from a specific CPACS (XML) path |
| updateDoubleElement | Update (replace) a double element pointing to specific CPACS (XML) path |
| save | Save the specific document |

**Table 2.1:** Some TIXI routine for Python programming language

Since CPACS technology is capable to describe the entire aircraft geometry, it is also necessary to have another library that allows to take action on the geometry directly. In order to fill this need, DLR provided **Ti**GL **G**eometry **L**ibrary [33], which has been developed using *OpenCASCADE*, and is able to represent the aircraft geometry by B-spline curves and surfaces. Several external interfaces for C/C++, Python, Matlab and FORTRAN are also available.
Following TiGL capabilities are listed:

- Reading and processing data and the information stored in CPACS file with regard to main aircraft components such as wings (main wing, vertical and horizontal tailplane) and fuselages.
- Building up the 3D airplane geometry for further processing.
- Computing, thanks to specific function, area and volume of each component
- Computing surface points in Cartesian coordinates by using common aircraft parameters

A future goal is to extend these capabilities to engine pods, landing gear and other components.

Furthermore, TiGL provides some useful functions useful to go deep in details about geometry description. For instance, some features can be used to evaluate how many wings or fuselages characterize the aircraft, and for each one to detect the number of sections and segments and the corresponding 'Uid'.
In order to visualize the 3D geometry of a certain airplane by using the data stored in the CPACS file, a visualization tool named *TiGL Viewer* have been developed. This tool can be used either as executable standalone software in which a CPACS file can be loaded or as a tool already integrated within RCE environment. The CPACS input file can be provided by using the *Input Provider* default feature available in RCE.
Finally, it is also possible to export the geometry in different format such as STEP, B-Rep, VTK, STL and IGES as shown in Fig. 2.23.

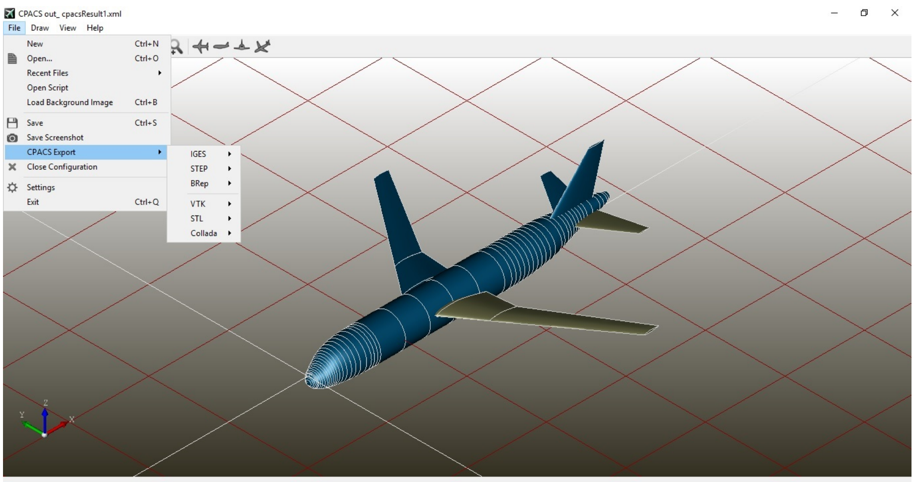



**Figure 2.23:** TiGL Viewer

## 2.4 Software to support AGILE MDO

To accomplish AGILE objectives and to manage all the tasks that involve the great amount of partner in the consortium, developing different software over the project duration was deemed necessary. In the next sub-section KADMOS, CMDOWS, VISTOMS, Brics and KE-chain (Sec. 2.4.1) will be described.

### 2.4.1 KADMOS, CMDOWS, VISTOMS, Brics and KE-chain

A Python software implementation, called KADMOS (Knowledge- and graph-based Agile Design for Multidisciplinary Optimization System), was proposed in order to increase the agility of aircraft design teams in performing collaborative multidisciplinary design analyses and optimization (MDAO) tasks by means of graph manipulation techniques.

This novel methodology [44], based on graph manipulation techniques, is capable to improve the agility of the team of expert by i) reducing the set-up time required to compose large and complex MDAO models, ii) enabling the systematic inspection and debugging of this model, and iii) allowing an automated creation and reconfiguration of optimization strategies, including the accompanying executable workflow. These advantages have been made possible thanks to pre-existing techniques suited for performing MDAO such as the CPACS standard data schema, that contains aircraft data and tools meta-data, and RCE environment which enables complex MDAO workflow simulation. Graphs used by KADMOS are usually composed by *nodes* and *edges*, each one with a specific duty. Nodes representing partners' tools or functions addressed to manage the workflow process are named *functions nodes*, while elements coming from the CPACS schema are represented by *variables nodes*. Moreover, edges can be grouped according their peculiar function: *input edges*, *process edges* and *output edges*. Key data graphs are the Repository Connectivity Graph (RCG), the Fundamental Problem Graph (FPG) and the MDAO Data Graph (MDG).

RCG contains all the available information, while FPG only contains partners' tools and variables needed to solve the specific MDAO task. FPG is structured in such a way KADMOS algorithm can manipulate it by imposing a certain MDAO architecture such as Multidisciplinary Feasible (MDF) or Individual Discipline Feasible (IDF) ensuring enough operational flexibility. Moreover, it is also possible to choose to perform a MDA, a DOE or optimization task obtaining a new toolchain and executable workflow automatically. When a specific MDAO architecture is imposed, FPG changes to MDG.

Graphs representation by means of data flow aspect of the eXtended Design Structure Matrix (XDSM) is shown in Fig. 2.24, in Fig. 2.25 and in Fig. 2.26.

Due to the huge amount of disciplines involved, defining and setting up an MDO chain requires an accurate MDO system assembling in terms of managing tools data and all the process connections. Aiming at support MDO system definition, an open-source standard workflow schema, called CMDOWS (Com-

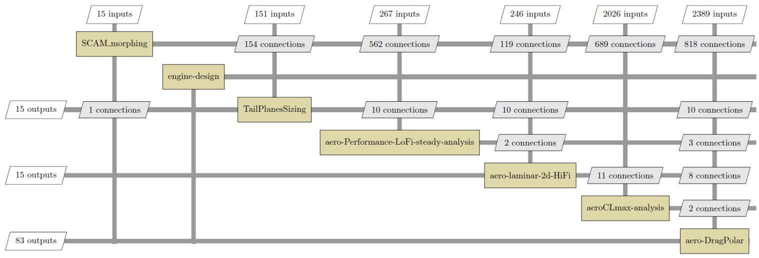



**Figure 2.24:** Repository Connectivity Graph example.

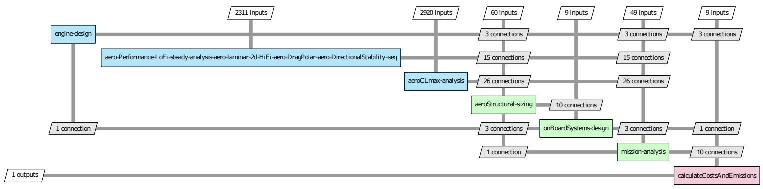



**Figure 2.25:** Fundamental Problem Graph example. Boxes can be divided in three different categories. The blue one represents the *pre-coupling functions*, the green one the *coupled functions* and the red one the *post-coupling functions*.

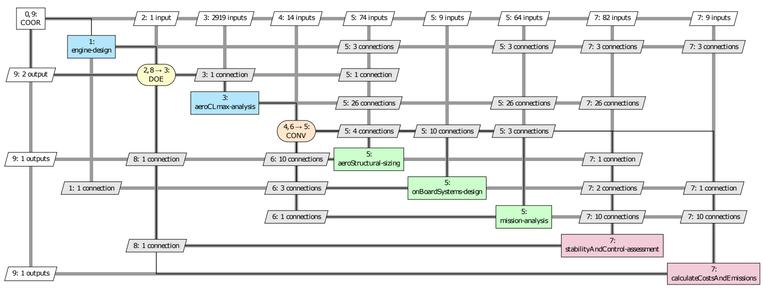
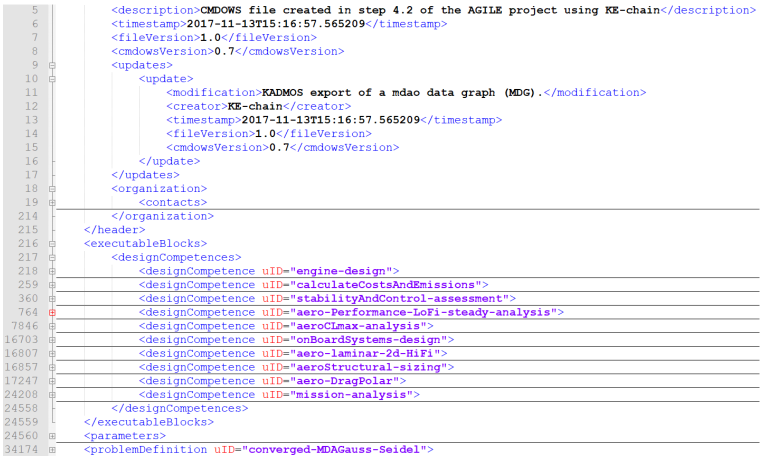



**Figure 2.26:** MDAO Data Graph example. Converged DOE architecture applied to a FPG obtaining a MDG.



```
<description>CMDOWS file created in step 4.2 of the AGILE project using KE-chain</description>
<timestamp>2017-11-13T15:16:57.565209</timestamp>
<fileVersion>1.0</fileVersion>
<cmdowsVersion>0.7</cmdowsVersion>
<updates>
    <update>
        <modification>KADMOS export of a mdao data graph (MDG).</modification>
        <creator>KE-chain</creator>
        <timestamp>2017-11-13T15:16:57.565209</timestamp>
        <fileVersion>1.0</fileVersion>
        <cmdowsVersion>0.7</cmdowsVersion>
    </update>
</updates>
<organization>
    <contacts>
</organization>
</header>
<executableBlocks>
    <designCompetences>
        <designCompetence uID="engine-design">
        <designCompetence uID="calculateCostsAndEmissions">
        <designCompetence uID="stabilityAndControl-assessment">
        <designCompetence uID="aero-Performance-LoFi-steady-analysis">
        <designCompetence uID="aeroCLmax-analysis">
        <designCompetence uID="onBoardSystems-design">
        <designCompetence uID="aero-laminar-2d-HiFi">
        <designCompetence uID="aeroStructural-sizing">
        <designCompetence uID="aero-DragPolar">
        <designCompetence uID="mission-analysis">
    </designCompetences>
</executableBlocks>
<parameters>
<problemDefinition uID="converged-MDAGauss-Seidel">
```

**Figure 2.27:** CMDOWS file example. XML visualization of the whole MDAO formulation.

mon MDO Workflow Schema), has been proposed, developed and provided to AGILE consortium by TU Delft University [45]. This schema, used to store results coming from KADMOS, is based on XML technology and allows an easy data exchanging between team members and their own tools supporting teams to setup the MDO system and its flexibility enables the creation of a versatile MDO architectures. Moreover, CMDOWS is also able to translate simple or complex MDO system formulation into an executable workflow ready to be run in any PIDO environment such as RCE or OPTIMUS. To accomplish all these tasks and improve MDO formulation process, CMDOWS was designed to be i) machine-interpretable and human-readable allowing user inspections at any level, ii)neutral because it is not specific to any product or project guaranteeing great flexibility, iii)adaptable to any updates or enrichment, iv)balance of redundant information aiming at minimum redundancy, v)support all MDO framework categories and system stages from the tool repository step to the optimized design, vi)support tool heterogeneity from simple mathematical expressions to advanced analysis tool or surrogate model.

In Fig. 2.27 the XML visualization of a CMDOWS file is shown. As can be noticed, even if this standard schema has been designed to be human readable, the XML format readability is quite poor also for MDO experts, especially for large MDO systems involving thousands of variables. Aiming at improving CMDOWS readability and MDAO system inspection, a dynamic visualization package, named VISTOMS (VISualization TOol for MDO Systems), was developed by RWTH Aachen University [46]. Thanks to VISTOMS package it is possible to visualize a CMDOWS file choosing between three different techniques:

- XDSM

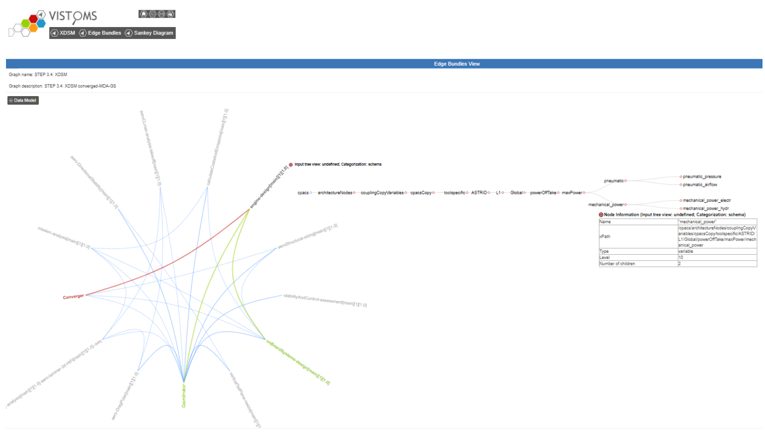



**Figure 2.28:** VISTOMS visualization of a CMDOWS file through Edge Bundling View technique.

- Edge Bundling View
- Sankey Diagram

*XDSM* format, already presented in Figures 2.24, 2.25 and 2.26, allows user to inspect an MDO system and to access to all the embedded information in a human intelligible way via browser. By clicking on a specific box it is possible to access to information such as number of input or output data, number of connections, tool's owner and so on.Moreover, it is also possible to download information as an XML file and to customize the data set or simply extract useful data.

*Edge Bundling View* visualization is a circular shaped layout in which element, such as variables and tools, are interconnected. Blue lines indicate a general dependence between two elements, while selecting one element some blue lines change to red and some others to green as shown in Fig. 2.28. The red lines indicate input data and green ones the output data. Furthermore, right click on a specific element allow to access to detailed information, as explained for XDSM, represented by a hierarchical tree and tabular format. The main difference with respect to XDSM is that in this case it is possible to access to data processed by different tools and to their information, but it is not possible to check workflow process details.

*Sankey Diagram* is the third available visualization technique. In Fig. 2.29 an example, referred to a complex MDO system formulation, is depicted. As described for the Edge Bundling View, red and green arrows indicate input and output data for a specific element. The width of each arrow is directly connected to the amount of data transferred between elements. This kind of visualization is characterized by the same capabilities described for XDSM and Edge Bundling techniques in terms of dynamic and interactive inspection.

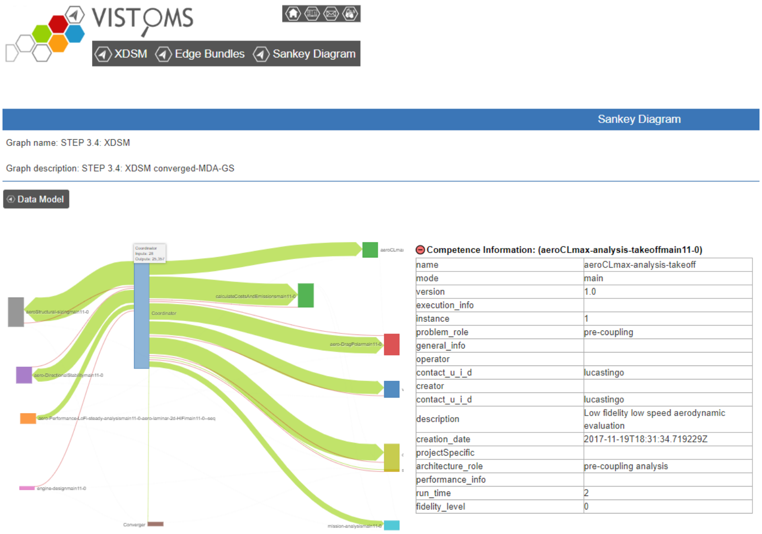



**Figure 2.29:** VISTOMS visualization of a CMDOWS file through Sankey Diagram technique.

The abovementioned software and instruments are fundamental in order to orchestrate a huge amount of data each other connected and coming from different disciplines and partners worldwide distributed. So, a crucial aspect is how to enable heterogeneous teams of experts exchanging information guaranteeing IT security for their own institutions. Within AGILE, NLR was in charge of developing and providing a software able to respect the described requirements, with a simple PIDO integration procedure and suitable for company as well as Universities or research centers. To achieve this aim, NLR developed and released Brics software [47]. Brics is based on a 'single-task' protocol that set the execution and data flow between an orchestrating (*master*) processing one organization and a remote ("slave") sub-processing another organization under control of a specialist who is notified to start the sub-processes. A further Brics capability is the possibility to support the notion of a 'multi-task' protocol, enabling the *master* to run a workflow with a convergence loop or a DOE and the remote specialist to easily deal with multiple series of sub-processes. In particular, the remote specialist needs to accept just once the 'multi-task' request and then the process is able to continue without any other action. In this way, Brics supports easy experimentation with different set-ups of collaborative scenarios to support the Design Campaigns and configuration of services involved. Moreover, Brics functionalities have been embedded by the PIDO providers using easy-to-use and modular interfaces in order to allow their inclusion in the workflows with minimal efforts and without need of programming skills. These front-ends building blocks expose to the user the basic information that Brics needs to operate. The AGILE integrator is only required to specify the file(s) that will be transferred, the name of the tool and the email address of the remote specialist. In this way, the specialists are notified through an e-mail message of the pending task, whose name and identification number are stored in the message and are the only data that specialists need to run the sub-process accessing to the inputs. For each Brics call it is necessary to store the input file(s) chosen by the *master* and used by the *slave* and the output data coming from the specialist at the end of the sub-process. For this scope a Neutral Data exchange Domain,a dedicated Microsoft SharePoint server the AGILE Teamserver, has been setup to serve as data server for the exchange of data of the collaborative workflows in the various AGILE Design Campaigns.

Aiming to support the entire MDO process, it is crucial to make KADMOS, CMDOWS, Brics and VISTOMS cooperate enabling an automated design and optimization process characterized by a fast setup phase, good flexibility, controlled and redundancy free data flow between elements. These aims were achieved, thanks to the collaboration of TU Delft, RWTH Aachen University and KE-works company, by implementing a browser platform named KE-chain which provide a way for the definition of the Knowledge Architecture mentioned in Sec. 1.5.

Usually, any design and optimization process is characterized by three main phases:

1. setup
2. operation
3. solution

The first one is related to the multidisciplinarity because the challenge is to integrate multiple and different design competences, coming from different partners, into a coherent and consistent design process targeting assembling a large, heterogeneous, distributed and automated design process. Furthermore, a portion of time of this step is addressed to establish conceptual constraints and design requirements.

Within such a complex system, wrong interpretation of the results, IT security trouble, intellectual property violation and time-consuming setup operations can be likely issues.

This time-consuming phase [48] needs to be accelerated, as proposed by AGILE project objectives in 1.5 and shown in Fig. 2.30, and carefully planned at the same time. In particular, reach the solution phase in a reduced time lap means to address less time to abstraction activities and more time in enriching the knowledge checking and analyzing the process and results.

In the operational phase, challenges are related to the missing overview of the entire MDO formulation that can lead to difficulties in finding possible inconsistencies in the process and the lack of operational flexibility when it is necessary to add another requirements, constraint or competence into an already assembled process.

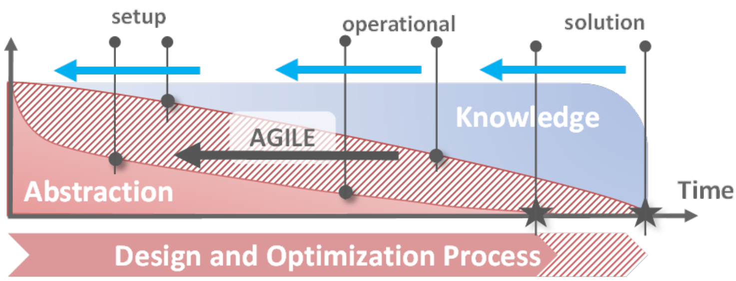



**Figure 2.30:** AGILE time reduction objectives.

In the final phase the optimal and robust solution needs to be found within an allocated time.

In order to face these challenging phases, by means of building an automated design process, KE-chain platform was developed and employed over the project. Firstly, five main agents responsible for specific tasks must be identified as reported in [48]:

- **Customer**: Customer and primary user of the framework. Responsible for defining the design task, top-level requirements, and available development lead-time. It includes the retrieval of results from the AGILE framework app.
- **Architect**: Responsible for specification of the design case in the AGILE framework, such as collecting the required competences, defining the design phases and the of the design space size to be explored.
- **Integrator**: Responsible for the deployment of the design and optimization (sub-)processes, and for the management of such processes within the AGILE framework. Intellectual property protection is also administrated.
- **Competence specialist**: Responsible for providing design competence within the framework, such as a simulation for a specific domain, or an optimization service. Specifications of the competences are managed in the AGILE framework app.
- **Collaborative engineer**: Responsible for providing the integration within the framework, necessary to connect the various competences and making them accessible to the framework. It includes the secure integration of software apps in different networks.

Then, in order to carry out the AGILE development process, which is part of the KA building block, it is necessary to start from the design case and requirements definition achieving the executable MDO workflow. To do that, operations in KE-chain are divided in five main steps:

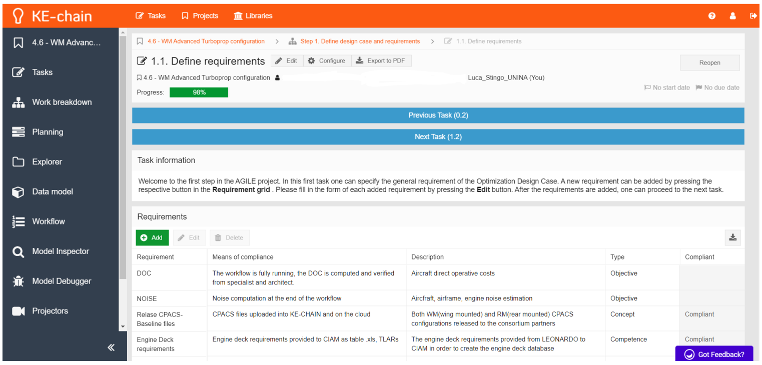



**Figure 2.31:** KE-chain framework step 1.1. Define design case and requirements.

1. Define design case and requirements
2. Specify complete and consistent data model and competences
3. Formulate design optimization problem and solution strategy
4. Implement and verify AGILE framework app or, in general, a collaborative workflow
5. Create executable collaborative workflow and select design solution

On KE-chain browser, after a project has been selected, it is possible to go to *Work breakdown* section and begin the entire process setup starting from step 1.1 *Define requirements*. Here, as shown in Fig. 2.31, aircraft requirements such as noise, maximum range, cruise mach and so on should be listed according to directions given by the *Customer* and *Architect* together with *Competence specialists*. Moreover, in step 1.2 *Define Competences & Parameters*, main parameters involved in the optimization task and the needed competences should be described as depicted in Fig. 2.32.

When step 1.1 and 1.2 are concluded, the CPACS base model must be uploaded in step 2.1. There is also the possibility of importing and uploading more than one XML file and KE-chain is able to merge these ones obtaining a single XML file, named "merged_file.xml" automatically, which will be selected as the CPACS reference one. Once this file has been correctly uploaded, a VISTOMS data model and a 3D ".stl" format CAD file are generated automatically. This step and related features are shown in Fig. 2.33, Fig. 2.34 and Fig. 2.35.

In step 2.2 it is necessary to *Import CPACSized competences into repository* uploading input and output XML files for each design competence. These files must be assembled following a standard procedure in terms of XML root tag name and child-node names. In this way, all available information can be

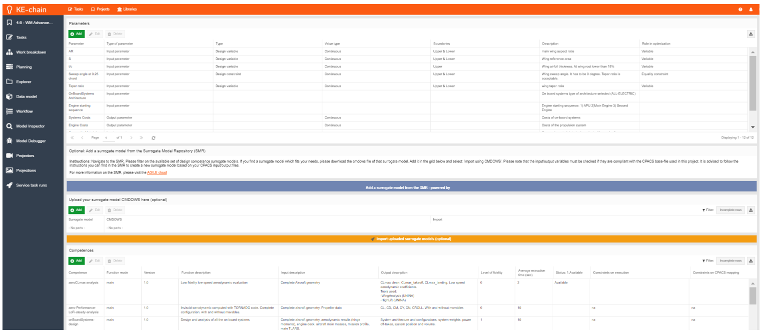



**Figure 2.32:** KE-chain framework step 1.2. Define parameters and competences.

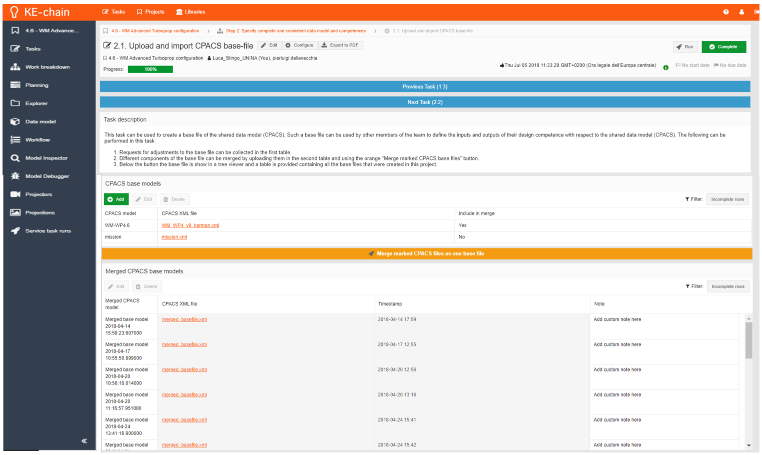



**Figure 2.33:** KE-chain framework step 2.1. Upload and import CPACS base-file.

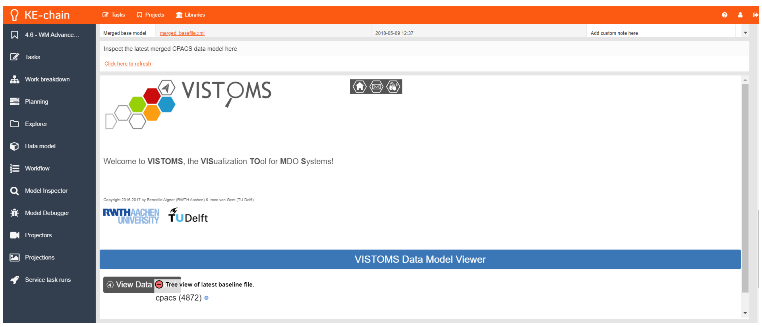



**Figure 2.34:** KE-chain framework step 2.1. Data model creation and VISTOMS visualization based on uploaded CPACS base-file.

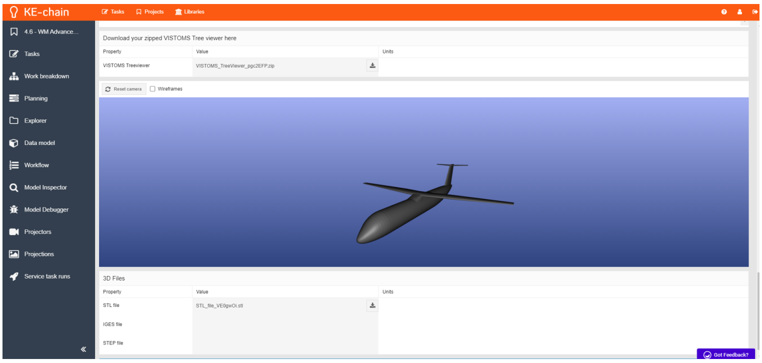



**Figure 2.35:** KE-chain framework step 2.1. CAD model creation based on the specific data model.

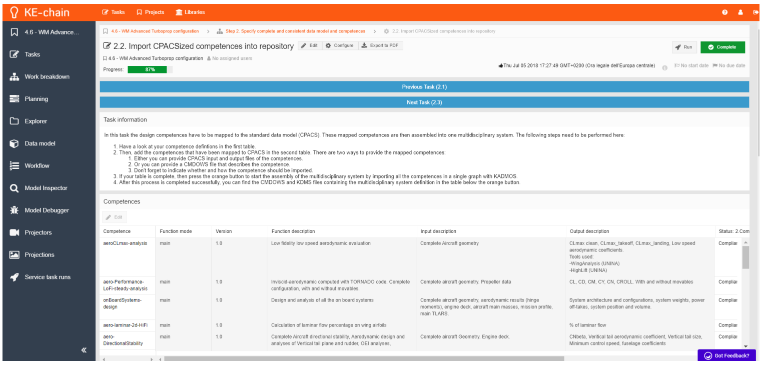



**Figure 2.36:** KE-chain framework step 2.2. Competences input and output data description.

converted into exportable KADMOS and CMDOWS files which are able to create connections between tools and to check for any kind of redundancy. The described step is illustrated in Fig. 2.36 and Fig. 2.37.

Thanks to the information stored in the competences' files, RCG is created in step 2.3 and the embedded VISTOMS package enables to *Inspect competence repository* as shown in Fig 2.38.

At the end of step 2, the first setup phase can be considered closed and the user can directly move to step 3.1. Within this step all data and information related to competences are available and stored in KADMOS, CMDOWS and RCG file. Since KE-chain is characterized by a deep flexibility, it allows to make modifications to the already accomplished actions keeping in memory the time history of the different version of output files (KADMOS, CMDOWS and RCG) created for each step. In this way, in step 3.1, the user can *Import CMDOWS and inspect RCG* choosing between the different version of the created files (see

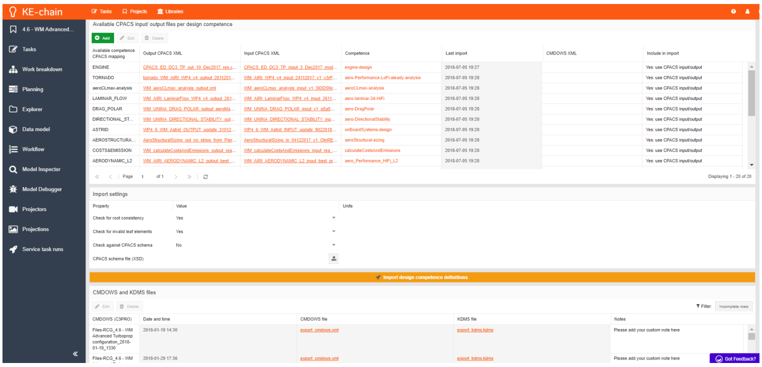



**Figure 2.37:** KE-chain framework step 2.2. Competences input and output XML files.

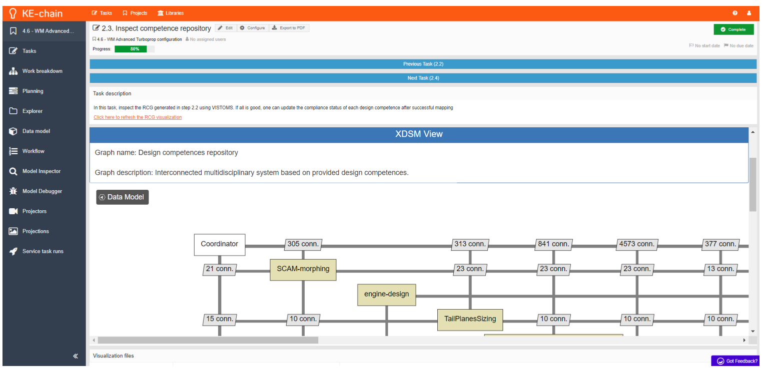



**Figure 2.38:** KE-chain framework step 2.3. RCG file creation and inspection through VISTOMS embedded package.

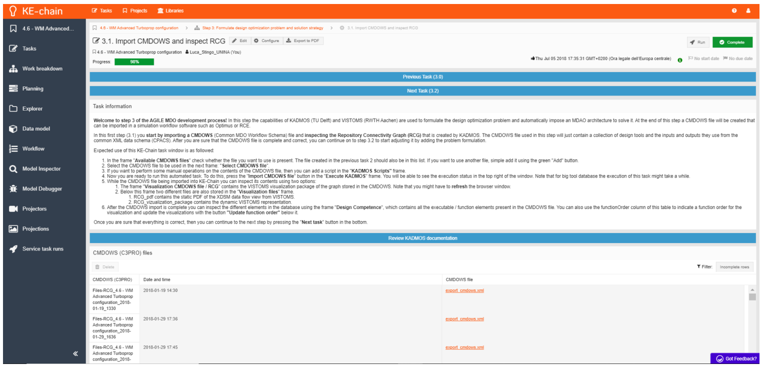



**Figure 2.39:** KE-chain framework step 3.1. User can choose between different CMDOWS files creating a specific RCG.

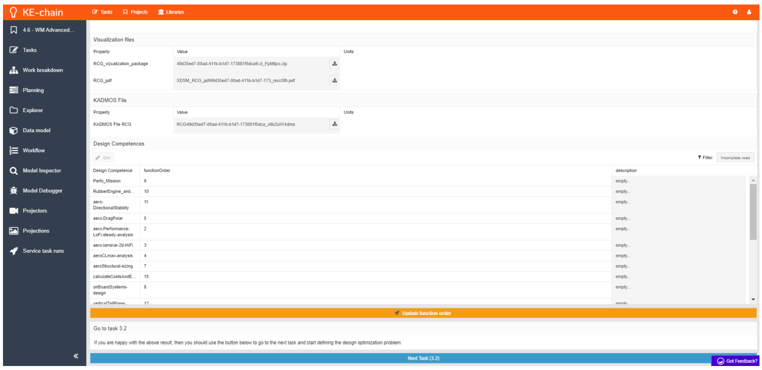



**Figure 2.40:** KE-chain framework step 3.1. RCG can be updated according to the function order indicated by the user.

Fig. 2.39). Once a CMDOWS file is imported, it is possible to update the order of each function or tool and update the RCG (see Fig. 2.40).

Moving to step 3.2, user can still *Manipulate design competences* deciding which ones need to be excluded or merged and which collision parameters must be removed (see Fig. 2.41). Applying these manipulations to the created RCG the result is the FPG as depicted in Fig. 2.42. Furthermore, FPG and KADMOS file can be also downloaded on a personal machine.

The next step concerns the roles of the parameters involved in the analysis and optimization task. In particular, in step 3.3 the user has to *Assign parameter roles* indicating which ones are the **design variables**, specifying their design space, setting one or more **objective functions** and **constraints** and finally choose which parameters will be monitored during the process and so will be considered as **state variables**. In Fig. 2.43 an example is shown. At the end of this step it is possible to click on *Enrich FPG* button to update the latest

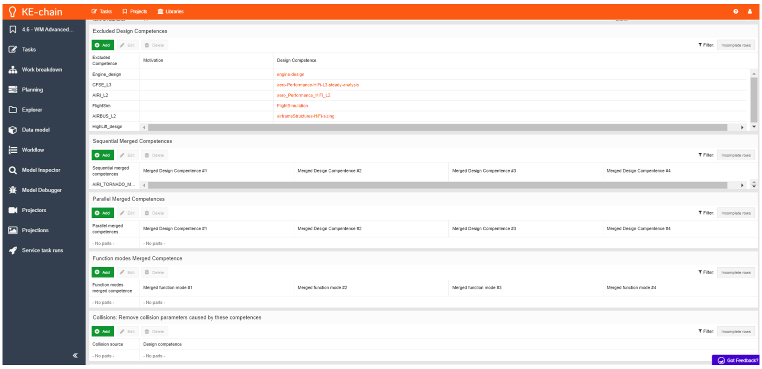



Figure 2.41: KE-chain framework step 3.2. Design competences can be manipulated, and collision parameters can be removed.

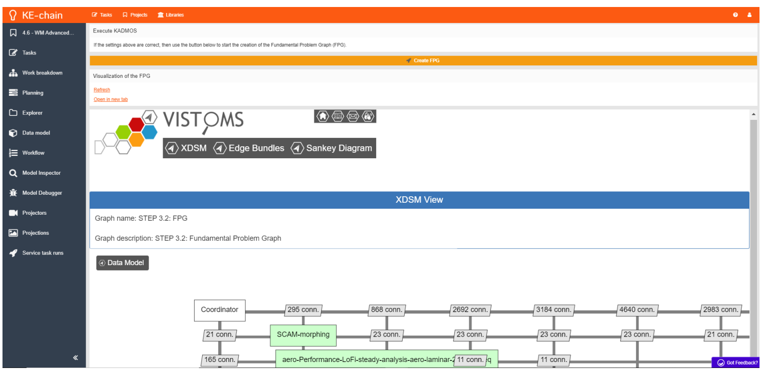



Figure 2.42: KE-chain framework step 3.2. RCG changes to FPG after design competences manipulation.

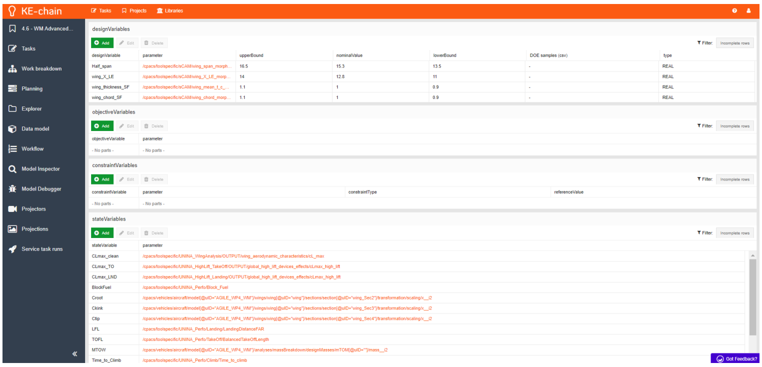



**Figure 2.43:** KE-chain framework step 3.3. Parameter roles assignment. User can set design variables, objective functions, constraints and state variables.

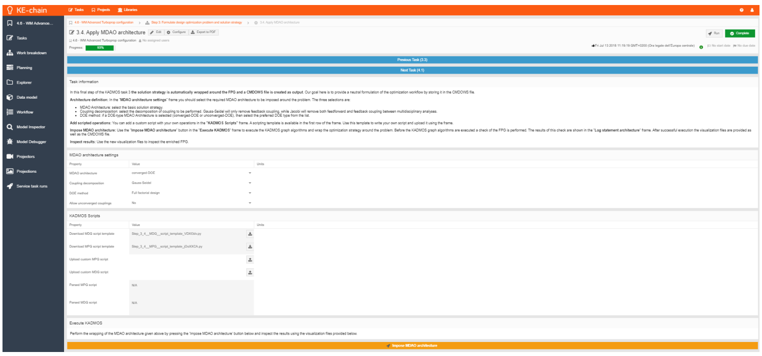



**Figure 2.44:** KE-chain framework step 3.4. MDAO architecture application.

FPG with the parameter roles assignment and inspect it thanks to VISTOMS.

Step 3.4 is one of the main steps. At that time the user must *Apply MDAO architecture* making a choice between simple converged or un-converged DOE or MDA, MDF (Multidisciplinary Feasible) and IDF (Individual Discipline Feasible) as illustrated in Fig. 2.44. Moreover, it is possible to indicate a *Full factorial*, *Latin Hypercube*, *Monte Carlo* or *Custom table* as DOE design approach and to choose between different tools and data coupling decomposition methodologies.

By clicking on the orange button *Impose MDAO architecture* the FPG will be updated and, through VISTOMS package, the user can visualize and inspect the workflow in XDSM, Sankey Diagram or Edge Bundle format and download it as PDF on his own machine. This is the last action in step 3. The next step deals with transforming the conceptual MDAO toolchain into an executable workflow.

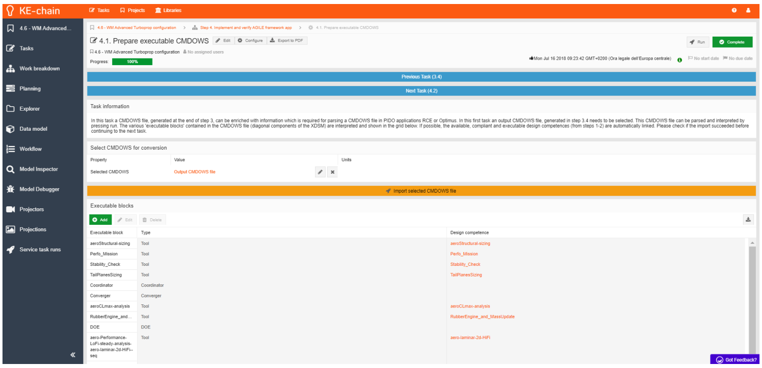



**Figure 2.45:** KE-chain framework step 4.1. CMDOWS file import linking executable blocks with design competences.

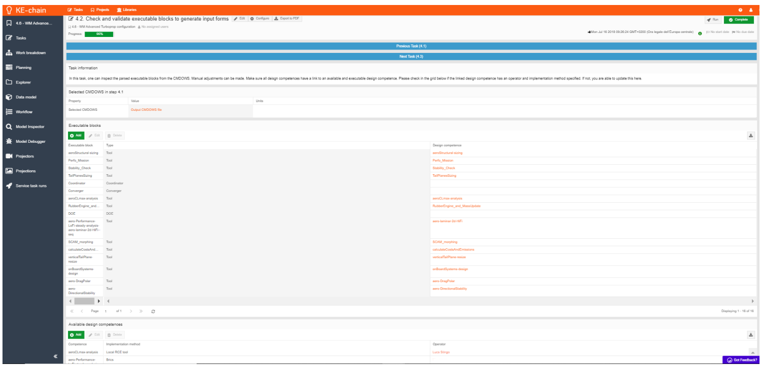


**Figure 2.46:** KE-chain framework step 4.2. Check and validate executable blocks to generate input forms.

To do that, the latest CMDOWS file needs to be imported and the tools, from now executable blocks, will be automatically linked to the design competence indicated in step 1 (see Fig. 2.45). If something goes wrong, the user can also manually link executable blocks with design competences.

In step 4.2 the user should to carefully *Check and validate executable blocks to generate input forms* not only ensuring the right links between blocks and discipline, but also imposing if those blocks need to be run on a local machine or in a remote way thanks to Brics software as can be noticed in the bottom of Fig. 2.46.

The step 4.3 is the last one before obtaining an executable CMDOWS file to import in RCE. Within this step the user has to *Add execution details* of the specific job to launch such as the *jobName*, a *notificationMessage*, *urlsite* and *folder* in which results must be stored and so on. An example of these actions is illustrated in Fig. 2.47. By clicking on the orange button *Generate*

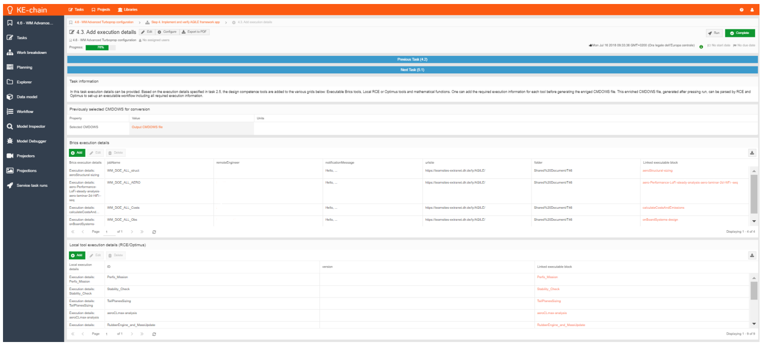


**Figure 2.47:** KE-chain framework step 4.3. Add execution details.

*the executable CMDOWS based on the input settings provided above* CMDOWS is created, stored in KE-chain and available to be downloaded, imported in a PIDO environment, RCE for instance, and ready to be executed to perform the MDAO task.

## Synthesis of the chapter

- AGILE instruments and technologies have been carefully described.
- **CPACS** standard file has been chosen as the unique way of communication between tools and teams of expert. It must contain both aircraft detailed description and tools results, largely reducing cross-connections among partners as shown in detail by the candidate.
- A user-friendly software named **RCE** has been provided to implement any kind of MDO problem formulation made of several tools. The author has deeply described the procedure to create and/or import a tool to allow a new reader to be able to be an RCE basic user.
- An overview of a generic structure of tool way of working developed by the author has been provided. It has been underlined that **CPACS** is the starting and arrival point. From it all data are collected by a **python** algorithm, developed by the author, through **TIXI** and **TiGL** libraries. Input are intepreted by a **JAVA** core and, thanks to python commands, output data are stored in the **CPACS** file again.
- An exhaustive description of the software developed during the project to support the AGILE MDO approach has been presented by the author. Key technologies are:
  - **KADMOS** $\rightarrow$ is capable to enable the inspection and debugging of a MDO problem allowing to reduce the set-up time required to compose a complex MDAO model and to ease the reconfiguration of the MDO startegies
  - **CMDOWS** $\rightarrow$ supports teams in the setup of MDO system, allows the creation of versatile MDO architectures and enables the translation from conceptual MDO problem to an executable one.
  - **VISTOMS** $\rightarrow$ makes possible the user-friendly visualization and inspection of MDO formulation through different view techniques.
  - **Brics** $\rightarrow$ guarantees IT security for each partner involved in a MDO task and enables the remote collaboration among specialists.
  - **KE-chain** $\rightarrow$ it contains all the described technologies. KE-chain is a online framework which allows the implementation of all the steps of the aircraft design activity from requirements to MDO workflow execution.

# Chapter 3

# Design Campaigns in the AGILE project

## Introduction

AGILE structure consisted of three sequential phases labeled as *Design Campaigns (DC)*. The structure was planned to be characterized by an increasing level of complexity in terms of aircraft type configurations and applied MDO techniques. Indeed, the first DC dealt with the definition of a set of TLAR to design and analyse the reference aircraft by means of available MDO methodologies in the consortium. In the DC-1 the author was involved as a specialist to develop several tools related to aerodynamic, ground performance and aircraft directional stability estimation. Then, the second DC was focused on the development of new and improved MDO techniques, of which one related to wing optimization developed by the candidate, to apply, in the third DC, to several disruptive aircraft configurations. In Sec. 3.1, Sec. 3.2 and Sec. 3.3 the DC-1, DC-2 and DC-3 will be described respectively.

## 3.1 Design Campaign 1

The first design campaign (DC-1) was labelled "Initialization". In this phase, the consortium provides, through each team of discipline specialists, already available analysis codes, interfaces, framework and optimization capabilities in order to formulate a reference distributed MDO system and workflow according to today's best-practice methods.

The definition of the DC-1 reference design system includes:

1. The formulation and deployment of a representative SOTA **"reference MDO architecture"**
2. The definition of a conventional **"reference aircraft"** as use case

The reference aircraft serves as Use Case for the reference AGILE Design System formulated and deployed during the DC-1. One of the objectives of the AGILE reference system is to develop the capability to produce a design solution as well as an optimum solution for conventional aircraft configuration based on a given a set of requirements. This latter was defined in WP2 as a set of TLAR chosen to be representative of state-of-the-art aircraft as designed today ("tube and wings" configuration), with applied technologies suitable to be adopted by aircraft with entry into service expected in 2020. TLARs for this Use Case were provided by Bombardier to the Consortium as listed in Table 3.1.

| Conventional Large Regional Jet Reference Aircraft (EIS: 2020) | | |
|---|---|---|
| | ***Metric*** | ***Imperial*** |
| **Range** | 3500 km | 1890 nm |
| **Design payload** | 9180 kg | 20220 lbs |
| **Max. Payload** | 11500 kg | 25330 lbs |
| **PAX** | 90 pax @ 102 kg | 90 pax @ 225 lbs |
| **MLW (% MTOW)** | 90% | |
| **Long Range Cruise Mach (LRC)** | 0.78 | |
| **Initial Cruise Altitude (ICA)** | 11000 m | 36000 ft |
| **Maximum operating altitude** | 12500 m | 41000 ft |
| **Residual climb rate** | 91 m/min | 300 ft/min |
| **TOFL (ISA, SL, MTOW)** | 1500 m | 4921 ft |
| **Vref (ISA, SL, MLW)** | <130 kts | |
| **Max. operation speed** ($V_{mo}/M_{mo}$) | 330 KCAS/0.82 | |
| **Dive Mach number** ($M_d$) | 0.89 | |
| **Fuselage diameter** | 3 m | 118 in |
| **Fuselage length** | 34 m | 111.5 ft |
| **Service life** | 80000 cycles | |
| **Fuel reserves** | 5% | 100 nm |
| **A/C configuration** | Low-wing, wing-mounted engines | |
| **Engine** | Provided (e.g.: PW1700G) | |
| **Design Objective** | Minimize COC (alternatively, min. MTOW) | |

**Table 3.1:** AGILE Reference Aircraft. DC-1 TLARs.

According to the DC-1 TLARs, a conceptual, but not optimized, design solution was provided by Bombardier as well. Main aircraft design parameters (e.g. planform area) and design choices used for the synthesis (e.g. airfoils type) are listed in Table 3.2 , and a 3D view of the corresponding geometry is given in Fig. 3.1.

The provided synthesized aircraft is not intended to be the "reference aircraft" itself, but it is indicative of a conceptual synthesis solution satisfying the given requirements, and provides notional data to verify the design solution produced by the conceptual tools available in the AGILE Consortium. The reference aircraft is then produced by the Consortium by employing the design tools available in the Design Campaign 1. Thereafter, the synthesized reference

| Typical geometry parameters, not optimized | |
|---|---|
| $S_w$ $(m^2)$ | **75** |
| $\Lambda_{c/4}$ **(deg)** | **25** |
| $\lambda_w$ | **0.25** |
| $AR$ | **9.5** |
| **Wing airfoils** | **DLR-F6 configuration** |
| **Winglet airfoils** | **Same as wing tip airfoil** |
| **Horizontal Tail & Vertical Tail airfoils** | **NACA 0012** |
| **Engine class (exemplary, not a fixed model)** | **e.g. PW1700G** |

**Table 3.2:** AGILE initial configuration. Main design parameters.

**Figure 3.1:** 3 views of DC-1 aircraft.

aircraft is evaluated by Bombardier and the Industrial partners. It is important to highlight that the scope of the DC-1 is not only to "produce" a reference aircraft, but mainly to be capable of managing several disciplines and tools aiming at assembling a well-workable toolchain. This latter is the reason why accomplishing the phases within DC-1 took more than one year following this roadmap:

1. Reference aircraft TLARs specification provided by aircraft manufacturer
2. Initialization L0. A synthesis solution for the reference aircraft is provided by using the available Consortium OAD tools → output was provided as an aircraft CPACS model
3. Feedback related to L0 solution for reference aircraft from aircraft manufacturers → refinement TLAR and synthesis update
4. Consolidation of CPACS L0 reference aircraft
5. Tools testing on DC-1 aircraft according to the reference Design System formulation and feedback on additional requirements and design options/assumptions

6. Assembling and running the distributed multi-level synthesis process (AGILE Design System reference)

The time spent was particularly addressed to collect all the tools already available, to make the tools compliant with CPACS file schema and to formulate a Design System taking into account for data exchanging and disciplines cross-connections.

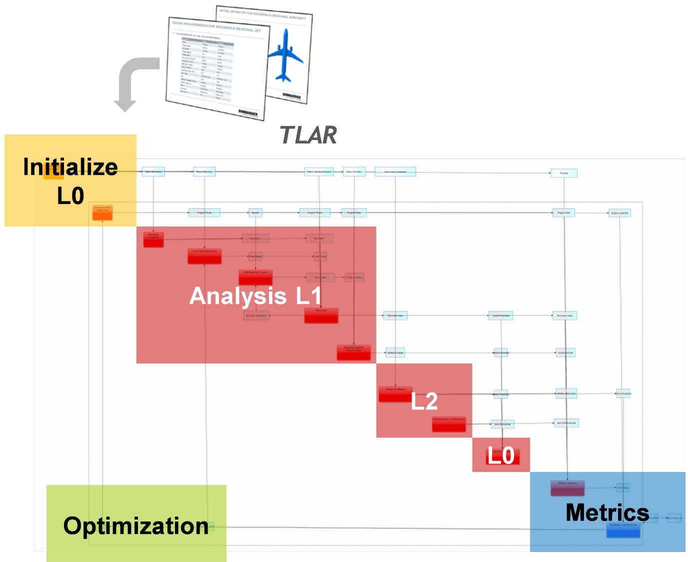



**Figure 3.2:** Reference Design System formulation.

The abovementioned reference Design System formulation is illustrated in Fig. 3.2. It contained several tools with different level of fidelity such as:

- L0 → empirical based design tools, as typical in SOTA conceptual aircraft design methods
- L1 → physics-based analysis tools, based on a simplified representation of the physics phenomena, suitable for the design of a conventional aircraft
- L2 → tools based on a complex and more realistic representation of the physics phenomena. These tools were included to be used as a final check after L1 tools analyses

The first step in assembling the reference aircraft was a Consortium investigation of the provided TLARs in order to provide a L0 synthesis design. This activity, named **Design Challenge – L0**, aimed at:

- quickly investigate the design space of the reference aircraft

- provide a reference DC-1 CPACS configuration to tune available analysis capabilities, also the L1/L2 tools, to be used during the DC-1
- setting up the "initialization process" in terms of details and assumptions underlying the L0 synthesis process through a set of iterations between the provided tools
- identifying early inconsistencies between the levels
- providing a feedback related to the design process and the optimization formulation

The time period addressed to achieve the above-mentioned target of **Design Challenge – L0** was divided in two phases:

1. A **synchronization phase** of the L0 capabilities, in which L0 tools providers discussed the design assumptions and the design methodologies already available in multiple iterations. The generated solutions were iteratively updated based on a weekly remote meeting and a consistent set of additional assumptions were defined for the reference aircraft. This phase was concluded in a period of nine months after a review of the solutions from Aircraft Manufacturer Bombardier and Leonardo company
2. **A consolidation phase**, in which assumptions were frozen and a L0 synthesis baseline was selected and refined by using the different L0 tools available. The resulting design was provided as a CPACS file and distributed to the Consortium.

During the synchronization phase different overall aircraft design capabilities and tools were used to generate the conceptual solution, such as:

- VAMPzero, DLR [49]
- Initiator, TU Delft [50]
- ASTRID, POLITO [51]
- ADAS, UniNa [35]

All the design capabilities employed rely on conceptual design methodologies and were already made available by the Partners. The CPACS compatibility for the design tools was developed and extended during the first six months of AGILE and was presented at the Half Year 1 Progress Meeting. These modules are capable to provide the full conceptual aircraft synthesis according to a set of the TLARs. However, at the first iteration, each of the partners was free to choose the required assumptions not explicitly specified in the requirements list. The solutions were compared and discussed, and additional common assumptions were considered for the next iterations.

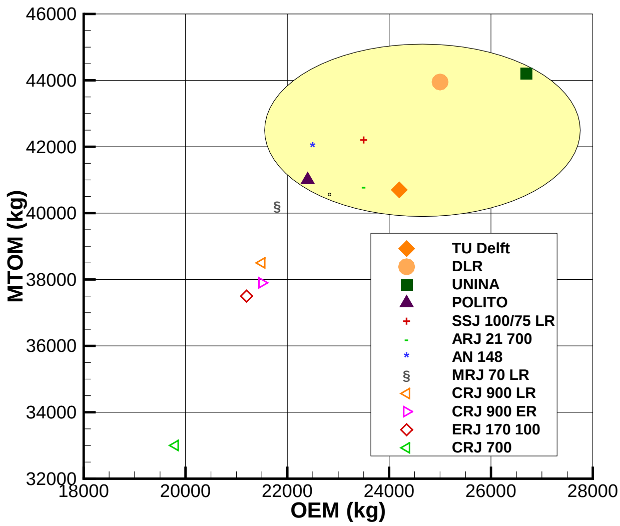



**Figure 3.3:** Design Challenge L0 solutions – existing aircraft

The design solutions have been reviewed at Month 09 and compared to existing aircraft with similar transportation missions (public available data), as shown in Fig. 3.3.

A comparison of the main synthesis results is presented in Table 3.3, and in Fig. 3.4. It can be noticed that all the synthesis tools produce comparable aircraft configurations. Most of the differences concern the estimation of the masses of the aircraft components (such as fuselage, systems, etc.) which affect the rest of the aircraft estimation components (e.g. wings masses). Another observation concerns the resulting wing areas (83-89 m2), which are higher than similar aircraft ( 70 m2). This effect was further analysed and was blamed on challenging take-off requirements and on being conservative as regards values assumed for the high-lift performance.

At the end of the first set of iterations, the provided TLARs were refined and new assumptions were introduced:

- LFL $\rightarrow$ 1400 m, challenging and main driver requirements for the lifting surfaces sizing
- $C_{L\_max}$ @ take-off $\rightarrow$ 2.2
- $C_{L\_max}$ @ landing $\rightarrow$ 3.0

Additionally, a reference engine performance deck was provided by CIAM. The granularity of information utilized by each of the conceptual tools is differ-

| Parameter | TU Delft | UniNa | POLITO | DLR |
|---|---|---|---|---|
| **Span (m)** | 28.4 | 29.11 | 28 | 29 |
| **Area ($m^2$)** | 84.35 | 89.17 | 83 | 88.9 |
| $\Lambda_{c/4}$ **(deg)** | 26.2 | 25 | 25 | 25 |
| **Æ** | 9.6 | **Fixed** 9.5 | **Fixed** 9.5 | **Fixed** 9.5 |
| $L_F$ **(m)** | **Fixed** 34 | **Fixed** 34 | **Fixed** 34 | **Fixed** 34 |
| $d_F$ **(m)** | **Fixed** 3.0 | **Fixed** 3.0 | **Fixed** 3.0 | **Fixed** 3.0 |
| **MTOM (kg)** | 40540 | 44183 | 41070 | 43861.5 |
| **OEM (kg)** | 24220 | 26721 | 22415 | 24974 |
| **MLM (kg)** | 38040 | 39764 | 37490 | 37951 |
| **Block Fuel (kg)** | 7480 | 8381 | 7160 | 7387 |
| **Wing mass (% OEM)** | 14% | 17% | 16% | 21% |
| **Wing loading (kg/$m^2$)** | 479.08 | 498.97 | 490 | 493 |
| **TOFL (m)** | **Fixed** 1500 | **Fixed** 1500 | **Fixed** 1500 | **Fixed** 1500 |
| **SFC cruise (-)** | **Fixed** 0.577 | **Fixed** 0.577 | **Fixed** 0.577 | **Fixed** 0.577 |
| **BPR (-)** | **Fixed** 6 | **Fixed** 9 | **Fixed** 9 | **Fixed** 9 |
| $C_{L\_max}$ **@ take-off (-)** | **Fixed** 2.1 | **Fixed** 2.1 | **Fixed** 2.1 | **Fixed** 2.1 |
| $C_{L\_max}$ **@ landing (-)** | **Fixed** 2.8 | **Fixed** 2.8 | **Fixed** 2.8 | **Fixed** 2.8 |
| **L/D @ cruise (-)** | 16.8 | 16.2 | - | 18.1 |

**Table 3.3:** Synthesis solutions comparison and assumptions.

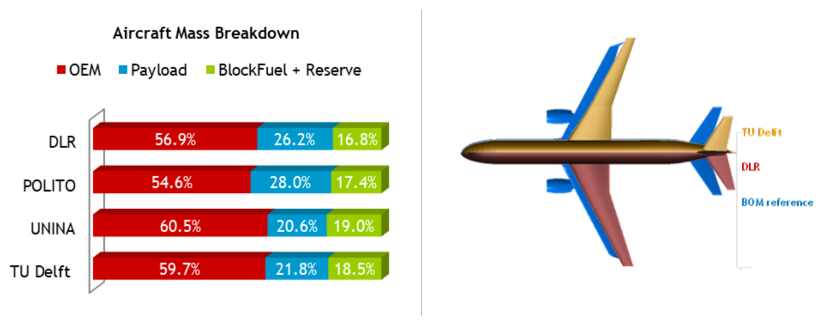



**Figure 3.4:** Design Challenge L0 – synchronization phase results.

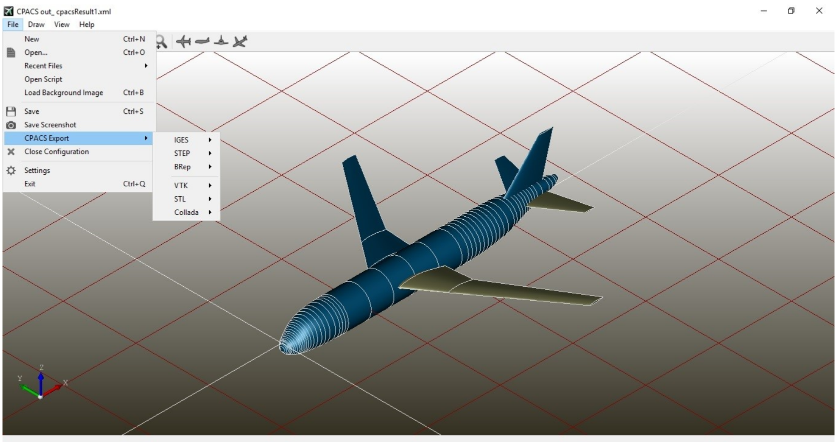



**Figure 3.5:** TiGL Viewer representation of the DC-1 reference aircraft.

ent, however, at the conceptual stage most of the information extracted from the performance deck were related to the values of specific fuel consumption for the main segments of the mission profile (cruise, climb, descent).

The main outcome of this design challenge is a consolidated conceptual synthesis solution of the reference aircraft wrapped into a CPACS file and represented thanks to TiGL Viewer software, as shown in Fig. 3.5.

Finally, the design space was further explored by performing additional synthesis studies, including the following variations:

- Assumed engine performance (SFC, BPR)
- Assumed high lift performance and high lift systems design
- 4 vs 5 abreast cabin configuration
- Traditional vs. more electric systems architecture
- Design of experiment on aircraft wing based on given TLARs

The author deeply contributed to the Design Campaign 1 by performing synthesis studies by using the internal Overall Aircraft Design software named ADAS and by providing different competences wrapped into peculiar tools. The Overall Aircraft Design phase was carried out manually by each involved partner, since at that stage was really difficult to orchestrate the whole system made of several disciplines and part of the time was spent in organizing a feasible MDA architecture. In particular, at least four months were spent by the author to implement and to improve a conceptual layout of the entire workflow by focusing on partners' data cross-connections and implementing that information manually.

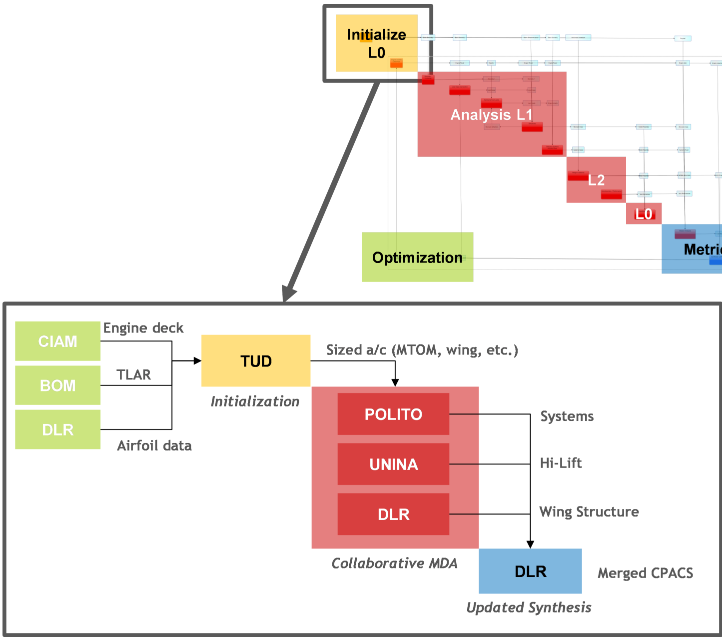



**Figure 3.6:** Consolidated L0 distributed synthesis. First AGILE distributed design system mock-up

Making use of the consolidated assumptions, the four individual conceptual tools were combined in a single L0 distributed synthesis, as shown in Fig. 3.6. The process can also be considered as a first 'mock-up' of the AGILE distributed design system.

Once a conceptual architecture was fixed, each partner started to share their own competences by implementing several algorithms and wrapping them into RCE tools. Implementation of these latter was really time-consuming and required a really large effort by the candidate because of the amount of disciplines, and so partners, and design variables involved in the design task.

The procedure to implement and wrap an algorithm into a tool can be divided in two sessions:

- **Information collection**
- **CPACS compliancy**

In the **Information collection** session, all information about input and output required and produced by the tools coming from other partners were carefully collected to create a set of tools able to of being cross-connected within a workflow and able to make use of results coming from each other. Each partner of the consortium was invited in compiling a certain file reporting all the useful information about its own tool describing in detail what its specific tool needed to work correctly. In this way a so-called conceptual *Tools catalogue* was created. A remarkable amount of time was spent by the candidate to manage all the information in order to create his own tools and assemble a conceptual layout of the initial MDAO workflow.

The second session, **CPACS compliancy**, was the hardest part in terms of time and efforts made by the author. Firstof all, for each of the tools created, the Python algorithm was implemented by coding a pre-processing section to extract from the specific CPACS file all the geometrical, performance and weights data of the aircraft useful to perform the expected calculations.

In particular, as regards the extraction and manipulation of the quantities directly available in the CPACS file it was possible to gather them through a large use of TIXI libraries, while about the geometrical aspect both for lifting surfaces and for fuselage components evaluations were made by means of TiGL capabilities and libraries.

TiGL and TIXI routines enabled the author to detect how many wings or fuselages the airplane contained, and for each one to evaluate all the useful characteristics. In particular, careful programming steps were required to extract the coordinates of the points in the Reference system describing a certain geometry, starting from the local reference system coordinates ([eta, xsi] for the wing and [eta, zeta] for the fuselage), which were fundamental to calculate the main geometrical features required to allow the developed tools work correctly.

Going deeply in details, in a CPACS file, a wing consists of a set of segments, each having a trapezoidal planform, needed to model a nearly arbitrary wing geometry. In order to define the shape of each section, a set of airfoils, usually described as a list of points in relative coordinates for unit chord length, is also required. All definitions of the wing are located in the <wings> node inside the aircraft model specification. Just like an airfoil, each aircraft component, such as wing, fuselage, horizontal and vertical tailplanes and so on, has a unique identifier (the uID) to make it always referable avoiding any misunderstanding possibilities.

As reported in Table 3.4, several TiGL and TIXI features were used by the author for every tools developed over the project in order to evaluate fundamental quantities to perfrom some preliminary design analysis. The list of the routines used by the author has been reported following the Python script line by line. For the fuselage components, considering also that nacelles can be considered as tapered bodies, libraries similar to those listed in Table 3.4 were used. Since there is no TIXI or TiGL library which can evaluate the fuselage maximum diameter, a further effort was necessary to collect all the points of the fuselage sections and compare each section height and width to calculate the maximum diameter. To provide a clearer explaination about the way which CPACS file describes the aircraft components by the location of their points, in Fig. 3.7a simple schema is shown.

In general, several burdensome procedures were applied by the author to calculate all the data not directly available in the CPACS file such as quarter chord wing sweep angle, wind dihedral angle, wing section maximum thickness,

| Routine | Description |
|---|---|
| Tigl.open | It opens an XML file |
| Tixi.getNamedChildrenCount | It detects how many tags are placed under a main branch. |
| Tixi.getTextElement | It extracts a text element from a specific CPACS (XML) path. |
| Tixi.getTextAttribute | It extracts an attribute element from a specific CPACS (XML) path. For example aircraft or component uID. |
| Tigl.wingGetUID(i) | It gets the *i-th* lifting surfaces uID. |
| Tigl.wingGetSegmentUID(i) | It gets the *i-th* component uID. |
| Tigl.wingGetIndex(wingUid[i]) | It gets the lifting surfaces indices starting from their uIDs. |
| Tigl.wingGetSegmentCount(wIndex[i]) | It counts the segments of each lifting surfaces starting from their indices. |
| Tigl.wingGetSurfaceArea(wIndex[i]) | It evaluates the upper and lower area of each lifting surfaces starting from their indices. |
| Tigl.wingGetSymmetry(wIndex[i]) | It detects the symmetry plane each lifting surfaces starting from their indices. |
| Tigl.wingGetReferenceArea(wIndex[i],wSymmetry[i]) | It evaluates the planform area of each lifting surfaces starting from their indices and plane of simmetry. |
| Tigl.wingGetSectionCount(wIndex[i]) | It counts the sections of each lifting surfaces starting from their indices. |
| Tigl.wingGetSpan(wingUid[i]) | It evaluate the span of each lifting surfaces starting from their indices. |
| Tigl.wingGetMAC(wingUid[i]) | It evaluate the mean aerodynamic chord value of each lifting surfaces starting from their indices. |
| Tigl.wingGetChordPoint(wIndex[i], j+1, eta[h], xsi[k]) | It calculates the point required by the user starting from the indices of the component and its segment, eta and xsi local coordinates that can assuming values between [0,1]. |
| Tigl.wingGetUpperPoint(wIndex[i], wSegIndex[i][j], 0, 0) | It compute the coordinates of specific points positioned on the upper side of the wing starting from wing and segment indices and eta and xsi local coordinates that can assuming values between [0,1]. |
| Tigl.wingGetLowerPoint(wIndex[i], wSegIndex[i][j], 0, 0) | It compute the coordinates of specific points positioned on the lower side of the wing starting from wing and segment indices and eta and xsi local coordinates that can assuming values between [0,1]. |
| Tixi.getIntegerElement | It extracts an integer element from a specific CPACS (XML) path. |
| Tixi.getDoubleElement | It extracts a double element from a specific CPACS (XML) path. |
| Tixi.getVectorSize | It detect the size of a vector stored in the cpacs file |
| Tixi.getFloatVector | It gets vactor elements starting from the computed size |
| Tixi.updateDoubleElement | It updates (replaces) a double element pointing to specific CPACS (XML) path. |
| Tixi.updateIntegerElement | It updates (replaces) an integer element pointing to specific CPACS (XML) path. |
| Tixi.updateTextElement | It updates (replaces) a text element pointing to specific CPACS (XML) path. |
| Tixi.updateFloatVector | It updates (replaces) a whole vector pointing to specific CPACS (XML) path. |
| Tixi.save | It saves the specific document. |
| Tixi.close | It closes the specific document. |

**Table 3.4:** TIXI and TiGL routines used by the author to develop all the AGILE tools coming from UniNa partner.

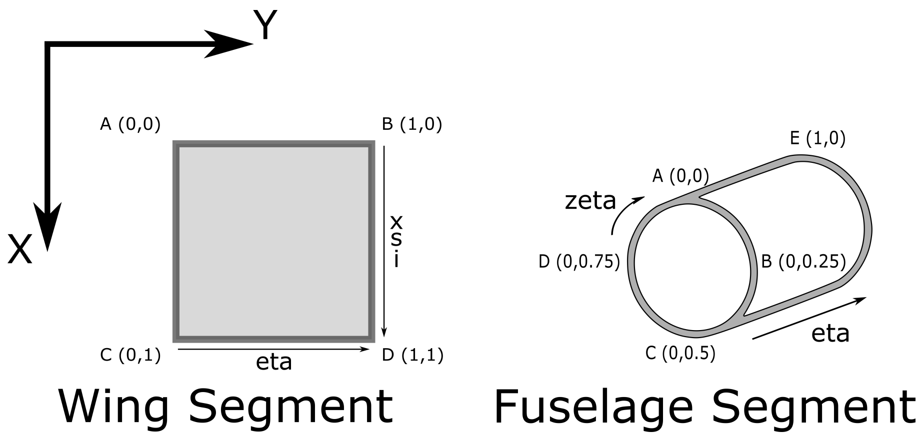



**Figure 3.7:** Wing and fuselage segment points definition in a CPACS file. This concept is the starting point for every calculation performed by the author through his own tools.

fuselage upsweep and windshield angle and so on.

Moreover, the best practice envisages that each CPACS tool needs to be equipped with a customized XML file named *Toolspecific*. This latter contains additional information such as aircraft and main components uIDs, airfoils names and input that can be received by other tools or manually inserted by the user (the author in this case). In that way, all the algorithms implemented for a certain discipline can be maintained unchanged even if the CPACS input file is referred to a different aircraft. The only file that needs to be modified is the *Toolspecific* one in which new uIDs, names and data must be varied. This way of working is one of the advantages provided by the usage of a common language schema such the CPACS and it allows to speed up the reconfiguration of a pre-assembled workflow saving a remarkable amount of time.

A simple flow of the procedure is depicted in Fig. 3.8. When the process starts the CPACS file is automatically merged with the toolspecific file generating a new CPACS file containing an extra tag named <toolspecific> in which all the additional information are stored. Then, the implemented python algorithm through TIXI and TiGL libraries is able to interpret the CPACS file and extract all the useful data that must be provided to the specific tool. When the calculations are finished, the python script generate a new CPACS tag, under the <toolspecific> one, named <partner_output>, where all the results are collected. As a final step, a new inspectable CPACS file is containing all the results came from all the partners involved in the task is created.

The tools developed by the author, as part of the DAF research group at UniNa, related to aircraft aerodynamic, design and ground performance estimation, were fully embedded into RCE framework and are capable to deal with a generic CPACS aircraft model through an automated process. During the first

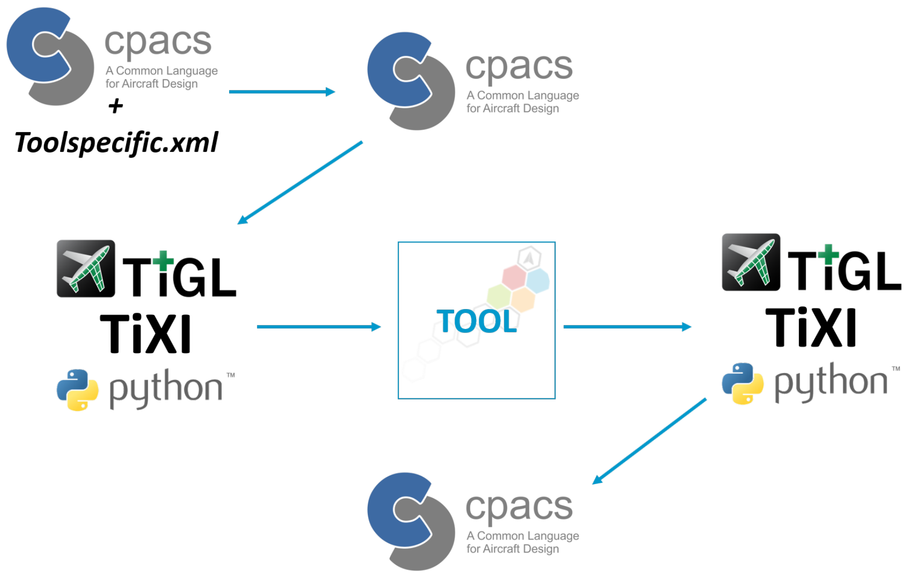



**Figure 3.8:** General way of working of a tool developed by the author.

Design Campaign, nine tools were developed, as shown in Fig. 3.9, and tested on the reference aircraft to evaluate its specific characteristics. These tools can be briefly described as follows:

- Wing Analysis

  It evaluates the wing lift curve of a lifting surface and the cl distribution along semi-span using the Nasa-Blackwell method [55]

- High-Lift

  It computes the aircraft aerodynamic coefficients with high lift devices (flaps and slats) deployed [56], starting from wing aerodynamic data in clean configuration calculated by means of Wing Analysis tool. The difference between take-off and landing condition lies on high lift devices rotation angle and chord extension.

- Take-Off Performances

  It is a simulation-based tool designed with the aim of evaluating the take-off distances and speeds of a generic aircraft in both All Operative Engine and One Engine Inoperative conditions by integrating the equations of motion that describe the aircraft state along all the maneuver [58]

- VeDSC (Vertical tail Design Stability and Control)

  It performs the calculation of vertical tail directional stability contribution and evaluates the interference factors among the main aircraft components [41] [52]

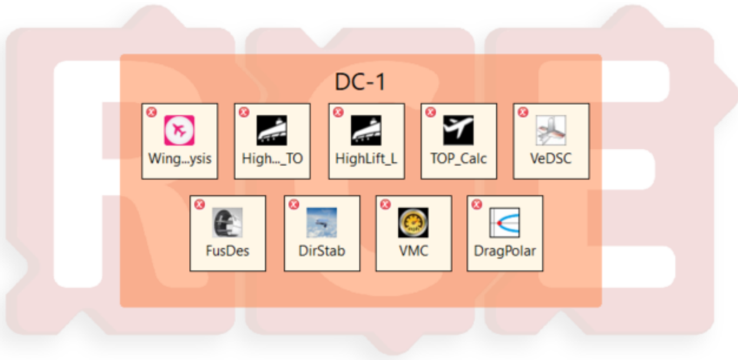



**Figure 3.9:** Tools developed by the author, as part of DAF ResearchGroup at UniNa, fully embedded into RCE environment.

- FusDes (Fuselage Design)

  It performs the calculation of fuselage directional stability contribution and evaluates the moment coefficients and geometry shape factors [42]

- Directional Stability

  It is a VeDSC and FusDes merging, in addition to these ones it performs the calculation of wing directional stability contribution and the directional stability of the whole aircraft configuration ($C_{N_\beta}$)

- VMC

  It computes the minimum control speed in case of inoperative engine(s) starting from engine and vertical tail characteristics. Moreover, it is capable of sizing the vertical tail surface corresponding to the VMC airspeed, to the VMC increased of 13% with respect to the stall speed in take-off condition, and to the VMC airspeed increased of 13% with respect to the stall speed in take-off condition specified by FAA documentation [57]

- Drag Polar

  It computes the aircraft drag polar according to semi-empirical approaches [53]. It is also possible to perform several analyses at different Mach and Reynolds number, angle of attack and yaw and altitude. Foe each analysis the tool provides zero lift drag values of the aircraft and its component in clean, landing and take-off condition, lift-drag curve taking also into account for induced drag contribution provided in the CPACS file by another partner.

Once the partners' modules were ready, in terms of CPACS compatibility and RCE integration, the next step was the Design Challenge L1 which was addressed to provide the AGILE reference system. It was planned to:

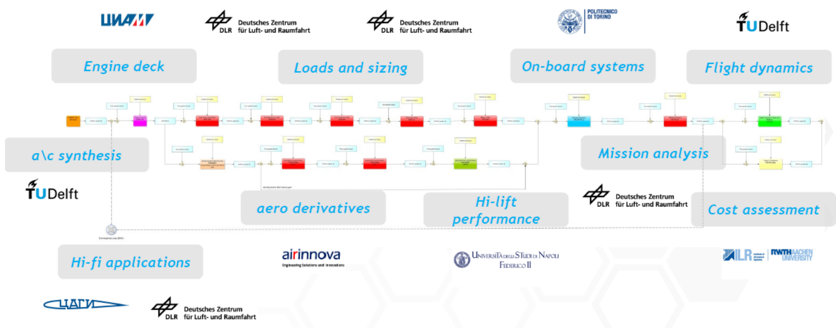



**Figure 3.10:** Conceptual structure of MDAO workflow.

- provide a full convergent MDAO process for the reference aircraft, coherent with the design capabilities available at this stage
- assemble the DC-1 distributed workflow including the partners' tools, which are CPACS compatible
- include design of experiments and a first optimization

The MDAO process can be represented by a "simulation chains" where several specialists' tools are involved. Each block is a design module or a group of them provided by the partners to the consortium and made accessible via remote service where UniNa, represented by the author, was the aerodynamic, ground performance and aircraft directional stability **specialist** and the DLR was the **Integrator**. In Fig. 3.10 a conceptual structure of the MDAO chain is shown, meanwhile in Fig. 3.11 the same MDAO integrated in RCE environment as an executable workflow is presented.

Furthermore, the deployed "workflow of workflows" has been provided as "service of services" and coupled to a surrogate based optimization strategy, named SEGOMOE, developed by ONERA [59]. This approach was retained for the SOTA distributed MDO system as it combines the advantage of MDF formulation (no modification of the MDA process, consistency of the design at each iteration of the optimization [60]) and of the use of surrogate models, enabling to reduce the number of calls to the MDA. The optimization problem can be defined as described in Eq. 3.1:

$$\begin{cases} min & \text{DOC} \\ w.r.t & \text{7 wing shape variables} \\ s.t & 2.2 - C_{Lmax} < 0 \end{cases} \tag{3.1}$$

The design variables, the constraints and the output (objectives) data considered to perform the MDA, DoE and MDO are listed in Table 3.5, Table 3.6 and Table 3.7.

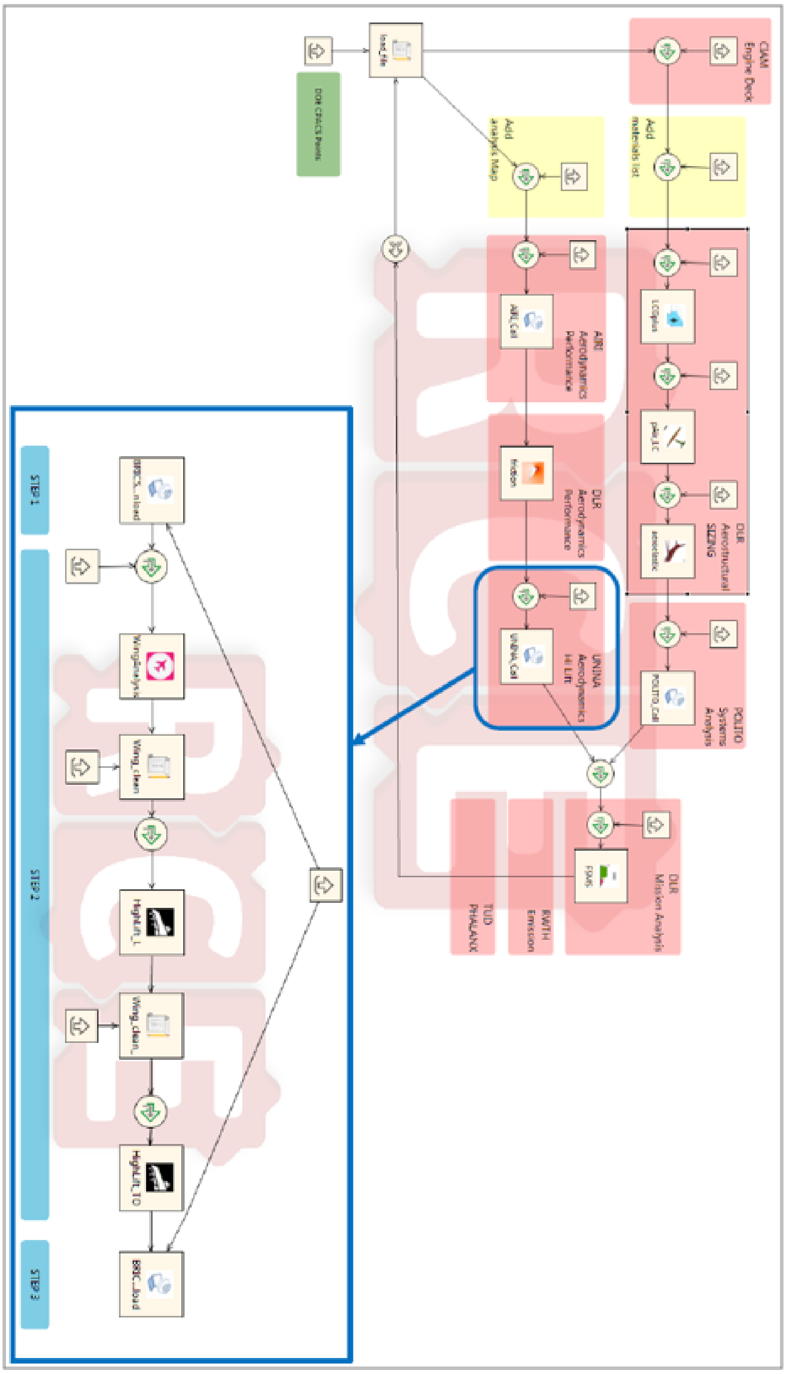



**Figure 3.11:** MDAO chain: the tools developed by the author in the first design campaign were provided to the main workflow managed by the Integrator through a remote service.

| Parameter | Wing area ($m^2$) | $\Lambda_{LE}(°)$ | Æ | $(t/c)_{root}$ | $(t/c)_{kink}$ | $(t/c)_{tip}$ | $Twist@tip(°)$ |
|---|---|---|---|---|---|---|---|
| Max | 95 | 34 | 9 | 15 | 12 | 11 | 1 |
| Min | 75 | 30 | 10.5 | 13 | 10 | 9 | -5 |

**Table 3.5:** Design space of the involved variables.

| State variables | | | | | | | |
|---|---|---|---|---|---|---|---|
| **chordKink** | **chordRoot** | **chordTip** | **kink_y** | **mac_Wing** | **span_Wing** | **taperRatioInboard** | **taperRatioOutboard** |

**Table 3.6:** Set of variables to be monitored over the process, also labelled as *State variables.*

At the end of the first Design Campaign was possible to run the whole MDA and DoE chain as a collaborative workflow, taking advantage of the tool(s) of each partner through the interaction between the coordinator and specialists which provided their own tools and competences as a remote service.

As the target workflow is characterized by both a high degree of discipline interdependency and many design variables one of the most straightforward solutions is the use of surrogate models. A surrogate model (SM) is an analytical formulation that replaces a complex model, or even a design analysis workflow, by means of data fitting. As a result, a surrogate model requires only little computation time, which is particularly useful for capturing complex analysis methods and applying them multiple times as part of a global optimization process. In the MDA workflow of DC-1, more than 2000 connections were identified between design competences; to reduce the complexity of the problem while keeping as much as possible its similarity with respect to the aircraft design process, several modifications were made and four clusters were built using a selection of design competences:

**Aerodynamic Cluster** This cluster gathers a morphing tool (that enables the modification of the full wing geometry from a set of design parameters) and aerodynamic performance computations including low-speed configurations. It takes as input the wing design parameters and provides lookup tables for aerodynamic coefficients, related to the specified wing design.

**On-Board Systems Cluster** This cluster aims at providing the On-Board systems performance in terms of weights and power, using the wing design parameters and other inputs such as the Fuel Weight and operational weights such as MTOM (Maximum Take-Off Mass).

**Structural sizing and Weight Cluster** This cluster provides the weight breakdown of the entire aircraft using as inputs the wing design parameters, the fuel weight and the systems weight. It also contains the load and structural sizing competence that sizes the wing structure and computes its weight.

**Mission performance Cluster** This cluster contains the Mission performance tool and uses as inputs the wing design parameters, the operational weights

| Output data | | | Constrains | | | | |
|---|---|---|---|---|---|---|---|
| $C_{LmaxTO}$ | $C_{LmaxL}$ | **DOC** | $S_{VMC}$ | **fuelMass** | **OEM** | **wingMass** | **systemsMass** |

**Table 3.7:** Output data and constraints.

and the Aerodynamic look up tables to run the full mission and provides the fuel weight.

The design competence clusters were then implemented as collaborative service-oriented workflows, and executed within a MDA and then a DOE studies in order to generate the databases for the clusters' surrogate models.

Starting from this achievement, the consortium made further efforts both in improving the reference MDAO architecture and in performing dedicated analyses and optimization tasks with regard to specific aircraft component such as the wing, the nacelle, the rudder and so on. These activities aimed at providing to the consortium and the scientific community new and high fidelity methodologies and results useful in aircraft design field. DC-2, described in Sec. 3.2, dealt with these tasks.

## 3.2 Design Campaign 2

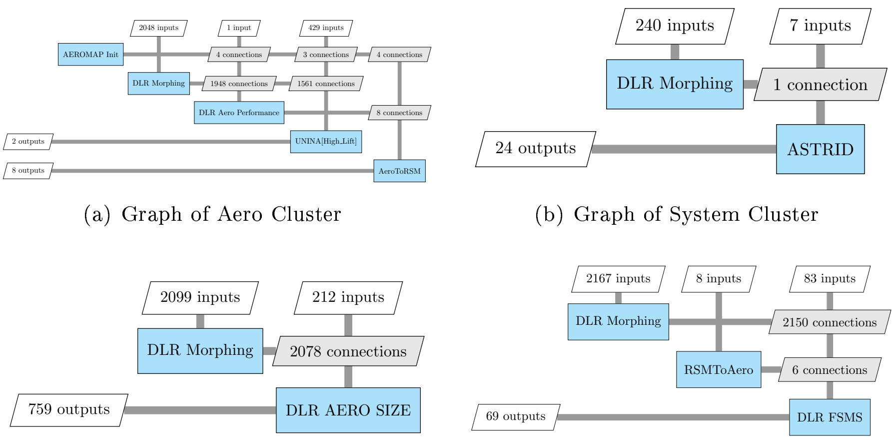
The second design campaign was labelled "MDO Test bench". The DC-2 activities were based on the outcome of the DC-1 work and were implemented during the second phase of the project. In this phase different optimization techniques are investigated in a structured way making use of the reference MDO framework and reference aircraft configuration, assembled in **Initialization** step. The developed best practice methods of all tasks are combined resulting in enhanced reference MDO systems. The number of use cases was expanded to five parallel ones and for each use case, a novel MDO strategy (addressing a specific collaborative scenario) was investigated and assessed for the resolution of the design of the reference aircraft. Depending on the use case, classical MDO formulations or more adapted ones were used. The five use cases were:

1. Use case focused on the improvement of MDO strategies with the development and integration of new design competences in terms of optimization algorithms and surrogates modelling. This use case and the results are presented in [23, 61]
2. The implementation of Uncertainty Quantification (UQ) methods and robust based design optimization in complex, variable fidelity optimization was the objective of the second use case [23]
3. The development of a mixed-fidelity MDO strategy was tackled with the integration of high-fidelity design competences and its combination with OAD level. The process is presented in [62]
4. A multi-scale application is described in [63] aiming at investigating the improvement of involving an aircraft component supplier (aircraft rudder) in the overall aircraft optimization process while keeping its specific framework
5. A large-scale system-of-systems application was studied, coupling Aircraft - Engine - On-board systems (OBS) - Emissions in a distributed framework approach with the involvement of disciplinary services from different partners [64].

This work will only focus on the enhanced methodologies developed by the candidate as part of the DAF research group (UniNa) to contribute to AGILE project over the DC-2. In particular, this section will only deal with the improvement of MDO strategies and with the development and integration of new design competences in terms of optimization algorithms and tools made available to all the consortium as a remote service provided through a Brics call.

Design and optimization studies can often involve multiple and conflicting objectives, a multi-objective optimization approach [65] should be employed allowing to consider many different parameters that could be a constraint or an objective function for a specific investigation. The design methodologies that allow to perform optimization studies during the aircraft preliminary design phase have been already implemented in a software package [66] and can be also built up by means of surrogate models [42, 52, 67].

These latter were obtained at the end of DC-1 by implementing different *Service Workflows* related to different disciplines. As shown in Fig. 3.12, each cluster is a workflow that can be run as a remote service through a Brics call. The XDSM graphs in Fig. 3.12 were then translated into executable blocks which contain a certain response surface, an example is depicted in Fig. 3.13, that can be queried by means of a suited interpolation technique. For example, the executable version of the Aerodynamic Cluster was used to evaluate the zero-lift drag coefficient and the maximum achievable lift coefficient, and the Structural sizing and Weight Cluster was employed to calculate the wing weight. The created clusters and their interpolation allowed to perform the optimization of the reference aircraft defined during DC-1 moving to a new optimized aircraft configuration achieved in the DC-2. The whole aircraft optimization took also advantage of new and advanced optimization techniques developed by different partner in the consortium.




(a) Graph of Aero Cluster

(b) Graph of System Cluster

(c) Graph of Weight Cluster

(d) Graph of Mission Cluster

**Figure 3.12:** Graphs of the 4 assembled clusters.

In this context, the methodological enhancement proposed and developed by the author couples the Nash game theory (N) to a typical genetic evolutionary algorithm (GA) [61], reducing the number of the needed analyses [68] and the computational time, allowing a more realistic association among variables and objective functions [69]. The regional turboprop aircraft wing optimization problem is here approached by means of game theory solutions, in particular, the Nash equilibrium solution concept, for which no player has anything to

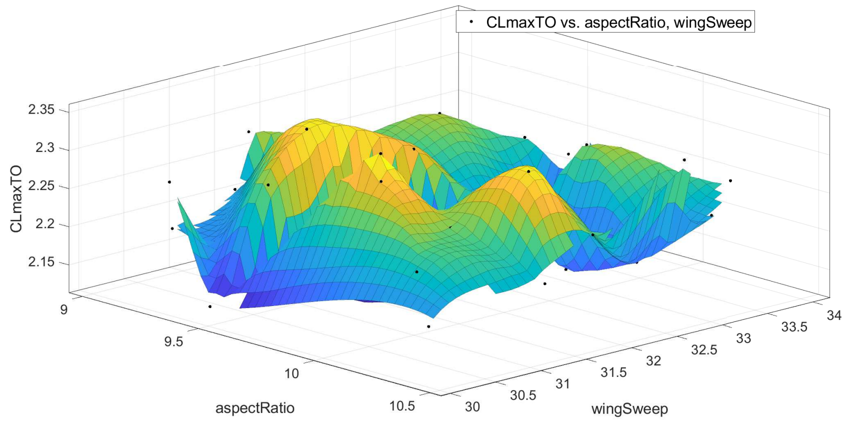



**Figure 3.13:** Response surface example from the Aerodynamic Cluster. $CL_{maxTO}$ variation with respect to AR and $\Lambda_{LE}$.

gain by unilaterally changing his strategy [70]. Reducing the general multi-player formulation to a two-player situation, the mathematical expression for the Nash equilibrium problem N is:

$$\begin{cases} \text{find } (\overline{x}_1, \overline{x}_2) \in X_1 \times X_2 \text{ such that} \\ f_1\left(\overline{x}_1, \overline{x}_2\right) = \min_{x_1 \in X_1} f_1\left(x_1, \overline{x}_2\right), \\ f_2\left(\overline{x}_1, \overline{x}_2\right) = \min_{x_2 \in X_2} f_2\left(\overline{x}_1, x_2\right) \end{cases} \tag{3.2}$$

where $(x_1, x_2) \in X_1 \times X_2$ are the players' variables or strategies, defined in their own strategy domains $X_1$,$X_2$, while $f_1, f_2$ are the players' objective functions. In the specific case object of this research the players' variables are in themselves a variables' set, as $x_1 = [\xi_1, \ldots, \xi_n]$, $x_2 = [\eta_1, \ldots, \eta_m]$ of dimension *n, m* depending on the variables partition introduced by the optimization problem decomposition, the latter being case specific. The genetic algorithm (GA) is an adaptive heuristic search method based on the principles of genetics and natural selection. Its name sets the roots in the analogy with living organisms in nature, being a GA capable of driving the evolution of a population (in conjunction with game theory, of players) under specified selection rules aiming to maximize their fitness w.r.t. the environment (i.e. an objective function under operating conditions and constraints). A GA structure could be regarded as a composition of the following pieces: i) a finite set of n-dimensional array, i.e. the population or players, usually encoded as a string of bits named genotype; ii) an adaptive function, called fitness, that estimates the goodness of the solution, indicating the individuals to let reproduce; iii) semi random genetic operators such as selection, crossover and mutation that operate on individuals, changing their associated fitness. The constraints are implemented by means of penalty functions, decreasing the individuals' fitness. The solution quality, enhanced by a large population, is also the bane of a GA in simple problems [71], leading in general to higher computational time. However, its wide usage is justified by several advantages, among which:

- The use of continuous or discrete variables.
- The trend of the objective function and its derivatives can be unknown.
- It deals with problems with many variables.
- It offers an intrinsic parallelization of the algorithm.
- It delivers satisfactory results in problems with extremely complex object functions hypersurfaces (i.e. with many local minima).
- It performs properly with numerically and/or experimentally generated data.

These features favor the GA in cases where the traditional optimization approaches fail. The algorithm for a two player Nash equilibrium game [72, 73] is here described for simplicity. Let *U, V* be players' strategy sets (both are metric spaces). Let $f_1, f_2$ be two real valued functions defined on $U \times V$ representing the players' objective functions. The used algorithm is based on the Nash adjustment process [74], where players take turns setting their outputs, and the chosen output of each player (in *U*) is his best response to the output previously chosen by his opponents (in *V*). The converged steady state value of this process is a Nash equilibrium of the game. Let $s = u, v$ be the pair representing the potential solution for a 2 person Nash problem. Then $u$ denotes the subset of variables handled by the player 1, belonging to *U*, and optimized under the objective function $f_1$. Similarly, $v$ indicates the subset of variables handled by the player 2, belonging to *V*, and optimized under a different objective function $f_2$. As stated in the Nash equilibrium definition [75], the player 1 optimizes pair $s$ with respect to $f_1$ by modifying $u$ while $v$ is fixed by the player 2; symmetrically, the player 2 optimizes pair $s$ w.r.t. the $f_2$ by modifying $v$, while $u$ is fixed by the player 1. This procedure can be implemented numerically considering $u^{k-1}$ and $v^{k-1}$ be the best values found by players 1 and 2, respectively, at step (or generation) $k-1$ . At next step, $k$, the player 1 optimizes $u^k$ using $v^{k-1}$ to evaluate the pair $\mathbf{s} = u^k, v^{k-1}$. At the same time, the player 2 optimizes $v^k$ using $u^{k-1}$ to evaluate the pair $\mathbf{s} = u^{k-1}, v^k$. The algorithm is structured in several phases, see also Fig. 3.14:

1. Generation of two different random populations, one for each player, at the first step. Player 1's optimization task is performed by acting on the first population and *vice versa.*

2. The sorting of the individuals among their respective population, is based on the evaluation of a fitness function typical of GAs. The results of the matches between each individual of population 1 with all individuals of population 2 (scoring 1 or -1, respectively, for a win or lost, and 0 for a draw) are stored, see Eq. (3.3).

$$\begin{cases} \text{if } f_1\left(u_i^k, v^{k-1}\right) > f_1\left(u^{k-1}, v_i^k\right), \text{fitness}_1 = 1 \\ \text{if } f_1\left(u_i^k, v^{k-1}\right) < f_1\left(u^{k-1}, v_i^k\right), \text{fitness}_1 = -1 \\ \text{if } f_1\left(u_i^k, v^{k-1}\right) = f_1\left(u^{k-1}, v_i^k\right), \text{fitness}_1 = 0 \end{cases} \tag{3.3}$$

A similar procedure is need for the player 2, as expressed in Eq. (3.4).

$$\begin{cases} \text{if } f_2\left(u_i^k, v^{k-1}\right) > f_2\left(u^{k-1}, v_i^k\right), \text{fitness}_2 = 1 \\ \text{if } f_2\left(u_i^k, v^{k-1}\right) < f_2\left(u^{k-1}, v_i^k\right), \text{fitness}_2 = -1 \\ \text{if } f_2\left(u_i^k, v^{k-1}\right) = f_2\left(u^{k-1}, v_i^k\right), \text{fitness}_2 = 0 \end{cases} \tag{3.4}$$

The individuals having an equal fitness value are sorted by $f_1$ for player 1 and on $f_2$ for player 2.

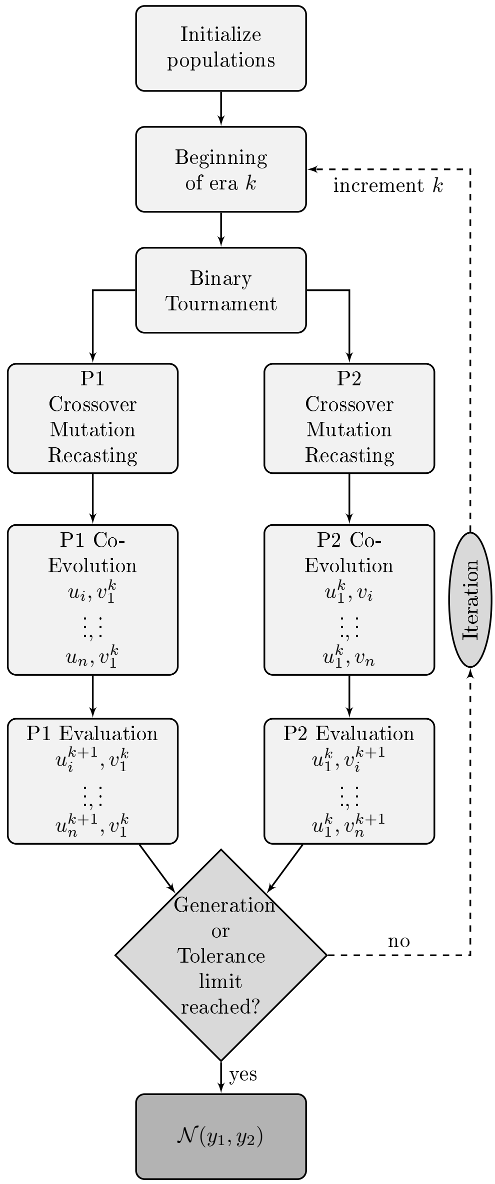
3. A mating pool for parent individuals is established, and crossover and mutation operations are performed on each player population. This new, evolved, population is sorted again, as described in phase 2.

4. At the end of the $k - th$ step, the player 1 deliver his best value, $u^k$ , to player 2 who will use it at step $k + 1$ to assign a unique value for the first part of his pair, i.e. the one depending on player 1, while the second part is that derived from crossover and mutation operations. Conversely, player 2 delivers his best value, $v^k$, to player 1 who will use it at step $k+1$, assigning a unique value for the second part of the pair, i.e. the one depending on player 2.

5. A Nash equilibrium is found when a maximum number of steps is reached, by repeating the phases 2-4. This algorithmic structure is similar to some of those used in literatures [76], with a major emphasis on fitness function consistency [69, 77].

In the game theory approach, a multi-objective problem is considered as a game with $n$ players, each one characterized by a pay-off. Each player wants to maximize his profit and will try to find an optimal game strategy. If each player has selected a strategy and no player can benefit from changing strategies while the other players keep theirs unchanged, then the current set of strategy choices and the corresponding payoff functions constitute a Nash equilibrium; other feasible possibilities are either to merge the advantages of Nash game and Genetic Algorithms (Nash-GAs) strategy or to use evolutionary optimization algorithms. The idea to apply NGA equilibrium solutions to the aircraft design and optimization leads to the chance to avoid a more arbitrary and less physically based variables association among the different objective functions, using, instead, a

**Figure 3.14:** In this figure the Nash Genetic Algorithm structure is shown. The sequence is composed by five steps, from the generation of random populations, one for each player, to achieving the Nash equilibrium, through the evaluation of a fitness function based on GAs approach and mutation operations among the individuals of each population. The detailed description is presented in the text.



more engineering reliable variables assignment based on well-known parameter association [61, 69]. In the NGA optimization approach, variables "cards" can be assigned to "players" (objective functions) in a unique case, assigning in a static manner these variables, or in multiples combinations, choosing cases to be optimized (until to the maximum number of possible combinations). The abovementioned approach has been applied, during the DC-2, to the common test case, a regional jet swept wing (see Fig. 3.25), in order to benchmark it against the mono-objective approach relying on a composite objective function, and also to a regional turboprop straight tapered wing (see Fig. 3.15). The use of the NGA optimization process was applied to the wing design considering both aerodynamic (including low-speed performance) and structural objectives in a single step. Starting from the turboprop use case, the game theory approach was used to perform and compare two different studies.

### 3.2.1 Two players wing optimization applied to turboprop aircraft configuration

The first study concerns the implementation and application, carried out by the author, of a two players NGA (see Eq. 3.5). The objective functions or pay-offs are: the $C_{Dw}$, computed with simple equivalent flat plate method and parabolic drag approximation, and the $W_w$, according to the methodology proposed by Raymer [5]. Five design variables have been used in this application, as shown in Eq. (4), including the wing aspect ratio AR, the mean wing thickness t/c, the swept area $S_w$, the leading edge swept back angle $\Lambda_{LE}$, and the taper ratio $\lambda$. These variables are assigned among the two players in all possible combinations, leading to *30* different games.

$$\begin{gathered}\Gamma = \langle players : 2;\ AR : \{11.45 - 13.26\},\ \tfrac{t}{c} : \{0.14 - 0.18\}, \\ S_w : \{55.27 - 70.1\},\ \Lambda_{LE} : \{0 - 3\},\ \lambda : \{0.45 - 0.64\}; \\ C_{Dw}, W_w\rangle\end{gathered} \tag{3.5}$$

In order to consider the effect on the overall aircraft weight the following considerations have been done: I) the aircraft weight is calculated as shown in Eq. (3.6), summing up the operative empty weight $W_{OE}$, the payload $W_{Payload}$ and the fuel $W_{Fuel}$; II) the wing weight, which affects the overall aircraft weight, is evaluated thanks to Eq.(3.9), where $W_{dg}$ and $N_Z$ represent the design gross weight and the ultimate load factor respectively; III) the operating empty weight $W_{OE}$ and the fuel weight $W_{Fuel}$ are calculated according to equations (3.7) and (3.8), respectively, where $W_{OE\_ref}$ and $W_{wing\_initial}$ are the initial reference weights; IV) based on the aircraft cruise lift coefficient $C_L$, fixed during the optimization process, the aircraft drag coefficient is assumed to be equal at the value obtained through Eq. (3.10) where the $AR$ and the Oswald factor $e$ vary for each wing. Equations (3.11) and (3.12) represent, respectively, the objective functions: $F_{obj\_1}$ considers the Prandtl-Glauert compressibility correction

| | b (m) | $C_{root}$ (m) | $\Lambda_{LE}$ (deg) | $\lambda$ | t/c | $S_w$ ($m^2$) | MTOW (kg) |
|---|---|---|---|---|---|---|---|
| **Reference Wing** | 27 | 2.57 | 2.80 | 0.62 | 0.173 | 61 | 22215.1 |
| **$C_{Dw}$ - Wing Weight @** $C_L = 0.50$ | 0.0209 - 1048 (kg) | | | | | | |
| | $W_{OE}$(kg) | | $W_{wing}$(kg) | | $W_{Fuel}$(kg) | | $W_{Payload}$(kg) |
| **Mass Breakdown** | 11917 | | 1048 | | 3098.1 | | 7200 |

**Table 3.8:** Turboprop reference wing characteristics.

($M_{corr}$); $F_{obj\_2}$ is the non-dimensional weight objective function.

$$W_{AC} = W_{payload} + W_{OE} + W_{Fuel}, \tag{3.6}$$

$$W_{OE} = W_{OE\_ref} + F_{obj\_2} \cdot W_{wing\_initial}, \tag{3.7}$$

$$W_{Fuel} = 0.54 \cdot \frac{S_w^2}{b} \cdot \left(\frac{t}{c}\right) \cdot \frac{\left(\left(1 + \lambda \cdot \sqrt{\frac{5}{6} \cdot \left(\frac{t}{c}\right)} + \lambda^2 \cdot \left(\frac{t}{c}\right)\right)\right)}{(1 + \lambda^2)} \cdot \rho_{fuel}, \tag{3.8}$$

$$W_{wing} = 0.0051 \cdot (W_{dg} \cdot N_z)^{0.557} \cdot S_w^{0.649} \cdot AR^{0.5} + \left(\frac{t}{c}\right)_{root}^{-0.4} \cdot (1 + \lambda)^{0.1} \cdot \cos^{-1}(\Lambda_{LE}), \tag{3.9}$$

$$C_{Dw} = C_{D0w} + \frac{C_L^2}{\pi ARe}, \tag{3.10}$$

$$F_{obj\_1} = C_{D0w} + \frac{C_L^2}{\pi ARe} \cdot M_{corr}, \tag{3.11}$$

$$F_{obj\_2} = \frac{W_w}{W_{wing\_initial}}, \tag{3.12}$$

Formally, the game is stated as shown in Eq. (3.5). The 2 players could play with the 5 cards (AR, t/c, $S_w$, $\Lambda_{LE}$, $\lambda$) alternatively assigned to the both players, and player 1 wants to optimize the wing drag coefficient (Eq. 3.11) and player 2 the wing weight (see Eq. 3.12).

Comparison of the NGA optimization results and the reference wing value is presented in Table 3.8 and shown in Fig. 3.16, where three solution points are marked with different shapes and colors.

In Fig. 3.16, the orange square and the red triangle represent two different Nash equilibrium points: the first one (game 3) represents the best point for the player 1 which minimizes wing drag coefficient with an increment in the wing weight. Vice versa the second point (game 23) minimizes the wing weight

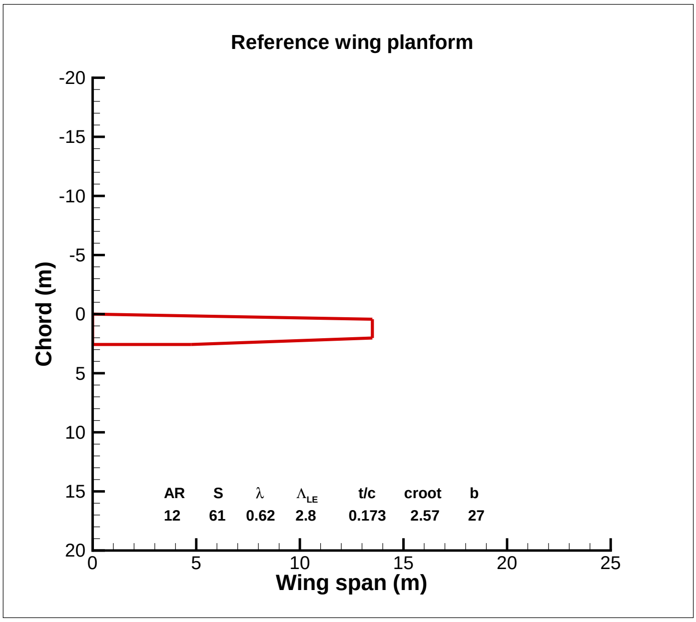



**Figure 3.15:** Turboprop reference wing.

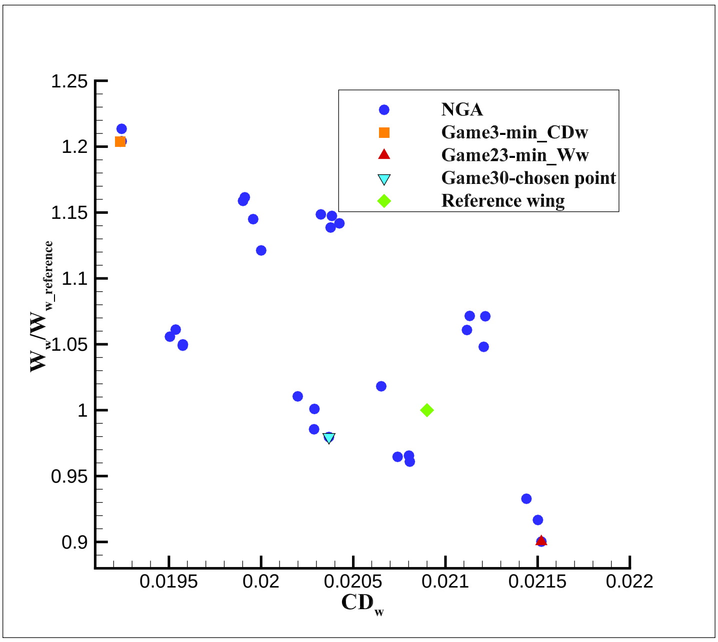



**Figure 3.16:** NGA optimization results applied to turboprop use case.

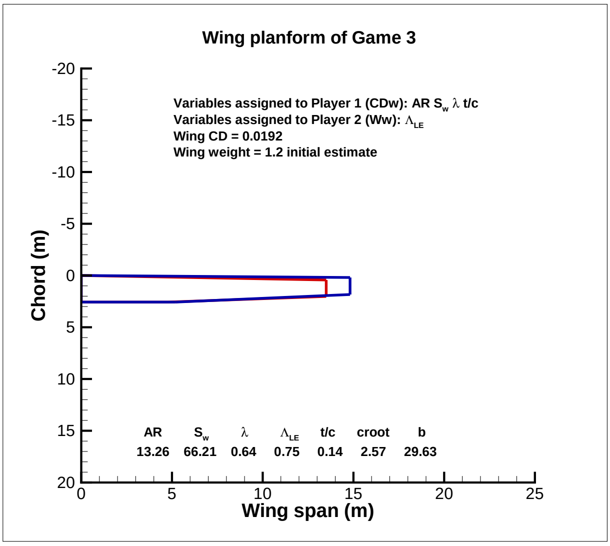



**Figure 3.17:** In this figure a comparison between the wing planform of Game 3 (blue line) and reference wing (red line) is shown. The wing configuration of Game 3 is characterized by a lower wing drag coefficient value and a higher wing weight value w.r.t. the reference wing. That is due to a higher aspect ratio value and a lower thickness ratio value than the reference wing respectively.

with an increment in the drag coefficient. The light blue triangle (game 30) represents one of the configurations chosen characterized by drag coefficient and wing weight values lower than the reference one. Table 3.9 summarizes the results of the NGA optimization, while in Fig. 3.17, Fig. 3.18 and Fig. Fig. 3.19 the three best wings planform compared to the reference planform (in red) are shown. The wing of the game 3 is characterized by a higher AR with respect to the wing reference value, which leads to a lower wing drag coefficient, while a lower value of the mean wing thickness percentage leads to a higher wing weight with respect to the reference one.

The opposite reasons lead to the results obtained for the game 23. The wing of the game 30 has characteristics similar to the wing in game 23 but

| | AR | $\Lambda_{LE}$ (deg) | b (m) | $\lambda$ | t/c | $S_w$ ($m^2$) | MTOW (kg) |
|---|---|---|---|---|---|---|---|
| **game 3** | 13.26 | 0.75 | 29.63 | 0.64 | 0.14 | 66.21 | 21934.25 |
| **$C_{Dw}$ - Wing Weight** | 0.0192 - 1257.6 (kg) | | | | | | |
| **game 23** | 11.45 | 3 | 25.18 | 0.64 | 0.18 | 55.40 | 21867.97 |
| **$C_{Dw}$ - Wing Weight** | 0.0215 - 943.2 (kg) | | | | | | |
| **game 30** | 11.45 | 0.46 | 25.29 | 0.45 | 0.14 | 55.85 | 21465.11 |
| **$C_{Dw}$ - Wing Weight** | 0.0204 - 1027 (kg) | | | | | | |
| **Reference Wing** | 11 .95 | 2.80 | 27 | 0.62 | 0.173 | 61 | 22215.1 |
| **$C_{Dw}$ - Wing Weight** | 0.0209 - 1048 (kg) | | | | | | |
| Mass Breakdown | $W_{OE}$(kg) | | $W_{wing}$(kg) | | $W_{Fuel}$(kg) | | $W_{Payload}$(kg) |
| **game 3** | 12126.60 | | 1257.60 | | 2607.65 | | 7200 |
| **game 23** | 11812.20 | | 943.20 | | 2855.77 | | 7200 |
| **game 30** | 11896 | | 1027 | | 2369.11 | | 7200 |
| **Reference Wing** | 11917 | | 1048 | | 3098.1 | | 7200 |

**Table 3.9:** Results of NGA application to the turboprop wing

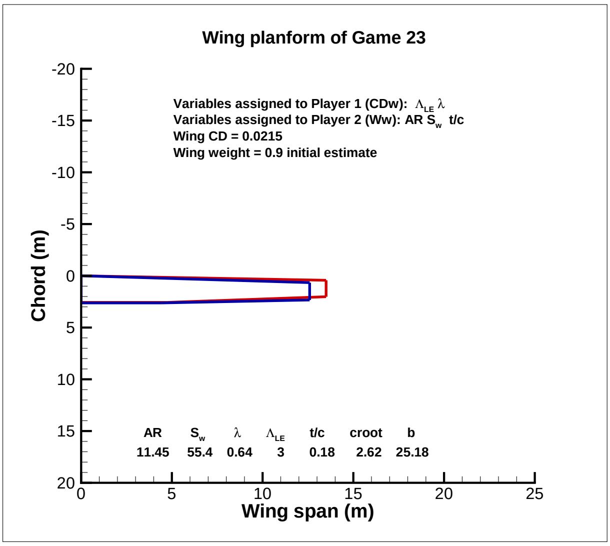



**Figure 3.18:** In this figure a comparison between the wing planform of Game 23 (blue line) and reference wing (red line) is shown. The wing configuration of Game 23 is characterized by a higher wing drag coefficient value and a lower wing weight value w.r.t. the reference wing. That is due to a lower aspect ratio value and a higher thickness ratio value than the reference wing respectively.

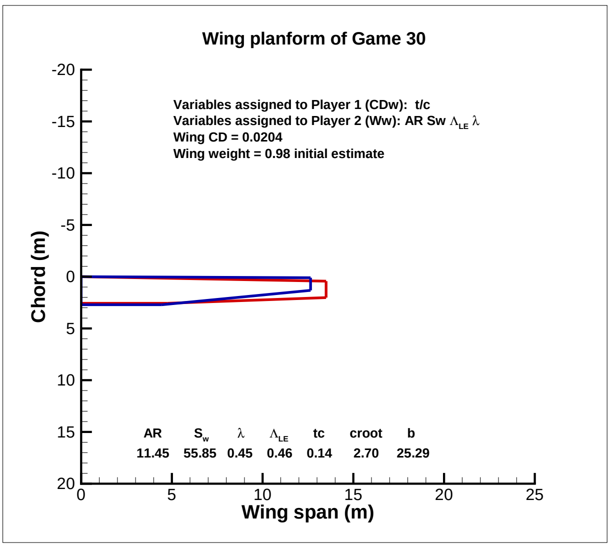



**Figure 3.19:** In this figure a comparison between the wing planform of Game 30 (blue line) and reference wing (red line) is shown. The wing configuration of Game 30 is characterized by a lower wing drag coefficient value and a higher wing weight value w.r.t. the reference wing. That is due to a combination of multiple factors such as a lower value of the taper ratio, the sweep leading edge angle.

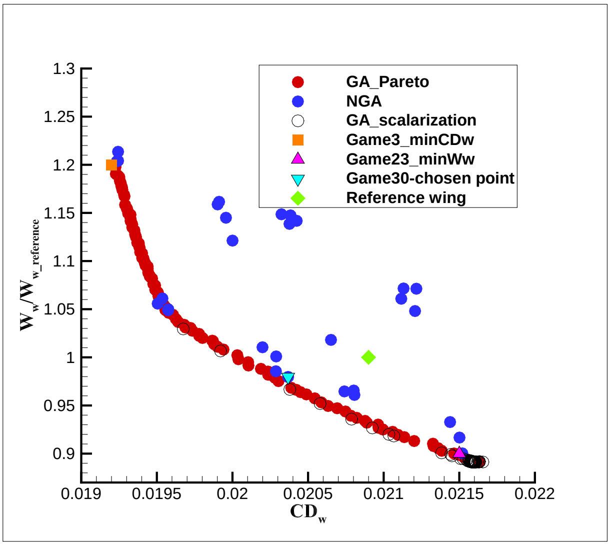



**Figure 3.20:** The results comparison among the three optimization approaches is presented. The NGA results (blue filled circle) are characterized by a good spread in the feasible zone of the Pareto front (red filled circle), while the GA scalarization points (black empty circle) are only located in the lower area of the feasible zone.

lower values of the mean thickness percentage and of the taper ratio lead to an optimized wing with lower $W_w$ and $C_{Dw}$. The detailed results are reported in Table 3.9. The developed NGA optimization methodology was also compared to GA scalarization and a multi-objective GA (Pareto front) algorithms provided by already existing Matlab libraries [78, 79]. In the scalarization optimization, GA algorithm has been chosen and the objective function was defined as an average weighted function, as shown in Eq. 3.13.

$$F_{obj} = F_{obj\_1} \cdot k_w + F_{obj\_2} \cdot k_{CDw} \cdot s_{CDw} \tag{3.13}$$

where:

- $k_w$, is the weight which represents the importance of the wing weight in the optimization process.
- $k_{CD}$, is the weight which represents the importance of the wing drag coefficient in the optimization process.
- $s_{CDw}$, is the scale factor useful to keep the same order of magnitude between the objective functions.

The range of values for $k_w$ and $k_{CD}$ is [0,1], each one used for weighting the corresponding objective functions. In Figure 6 the comparison between the results of the three approaches is shown, remarking a good agreement. The NGA results are characterized by a good spread in the feasible zone of the Pareto front (convexity of the Pareto front), while the GA scalarization points are only located in the lower area of the feasible zone.

### 3.2.2 Three players wing optimization applied to turboprop aircraft configuration

The second study was accomplished making a step forward in the implementation of optimization strategy increasing task complexity. Starting from the two players' optimization, a third player ($C_{Lmax_w}$) was added to also consider the aircraft performance in terms of maximum wing lift coefficient. For this reason, the set of equations was enriched by adding another equation (see Eq. 3.14) and, consequently, the NGA game was modified according to Eq.(3.15).

The design variables are assigned among the players in all possible combinations, leading to $n$ games among players. In this specific case, considering *5* design variables and *3* players, there are six way of assigning these variables:

- $[3, 1, 1]$, that means 3 parameters to Player 1, 1 to Player 2, 1 to Player 3
- $[1, 1, 3]$, that means 1 parameter to Player 1, 1 to Player 2, 3 to Player 3
- $[1, 3, 1]$, that means 1 parameter to Player 1, 3 to Player 2, 1 to Player 3
- $[2, 2, 1]$, that means 2 parameters to Player 1, 2 to Player 2, 1 to Player 3
- $[1, 2, 2]$, that means 1 parameter to Player 1, 2 to Player 2, 2 to Player 3
- $[2, 1, 2]$, that means 2 parameters to Player 1, 1 to Player 2, 2 to Player 3

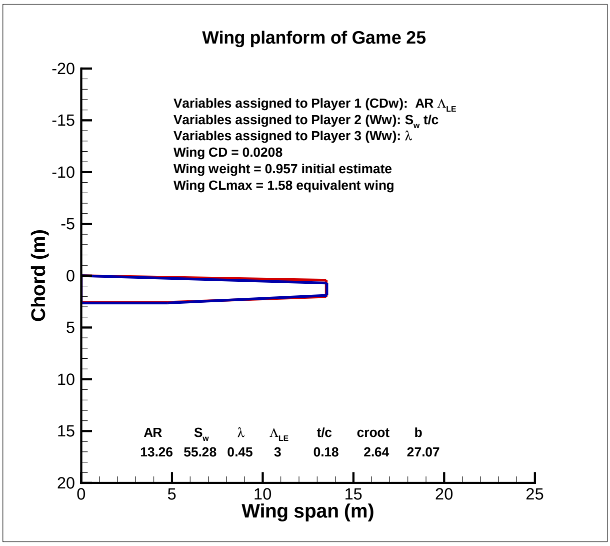
Each of these assignments leads to *10* possible combinations, and so *60* games in total.

$$F_{obj\_3} = C_{Lmax_w} \tag{3.14}$$

$$\begin{aligned}\Gamma = \langle players: 3;\ AR: \{11.45 - 13.26\},\ \tfrac{t}{c}: \{0.14 - 0.18\},\\ S_w: \{55.27 - 70.1\},\ \Lambda_{LE}: \{0 - 3\},\ \lambda: \{0.45 - 0.64\};\\ C_{Dw}, W_w, C_{Lmax_w}\rangle\end{aligned} \tag{3.15}$$

It must be noticed that the maximum wing lift coefficient, calculated using the Nasa-Blackwell method [55], is referred to the equivalent wing. For the three players' optimization, the algorithm scans all the *60* possible solutions, selecting only those for which the values of the objective functions are simultaneously better than the reference's wing weight and drag coefficient and greater than maximum lift coefficient. This solution is shown in Fig. 3.21. The solution proposed is characterized by three players' values improved with respect to the wing reference ones. In particular, the wing drag coefficient is reduced of about 1 drag count and the wing weight of about 4%, while the maximum lift coefficient is increased of about 0.07. In Table 3.10 the solution proposed has been compared to the reference wing. The application proposed was also compared to the Pareto front and the Genetic Algorithm modifying Eq. 3.13 in Eq. 3.16.

**Figure 3.21:** Comparison between the optimum wing solution of Game 25 (blue line), and the reference wing solution could be achieved assigning the leading-edge sweep angle and wing aspect ratio cards to the drag player, thickness ratio and wing area cards to the weight player, and the wing taper ratio card to the maximum wing lift coefficient player.




| | AR | $\Lambda_{LE}$ (deg) | b (m) | $\lambda$ | t/c | $S_w$ ($m^2$) | MTOW (kg) |
|---|---|---|---|---|---|---|---|
| **Reference wing** | 12 | 2.80 | 27 | 0.62 | 0.173 | 61 | 22215 |
| **$C_{Dw}-W_w-C_{Lmax}$** | 0.0209 - 1048 (kg) - 1.516 | | | | | | |
| **game 25** | 13.26 | 3 | 27.07 | 0.45 | 0.18 | 55.28 | 21924 |
| **$C_{Dw}-W_w-C_{Lmax}$** | 0.0208 - 1003 (kg) - 1.580 | | | | | | |
| **Mass Breakdown** | $W_{OE}$(kg) | | $W_{wing}$(kg) | | $W_{Fuel}$(kg) | | $W_{Payload}$(kg) |
| **Reference wing** | 11917 | | 1048 | | 3098 | | 7200 |
| **game 25** | 11872 | | 1003 | | 2853 | | 7200 |

**Table 3.10:** Comparison between the reference wing and the best solution of NGA application with 3 players.

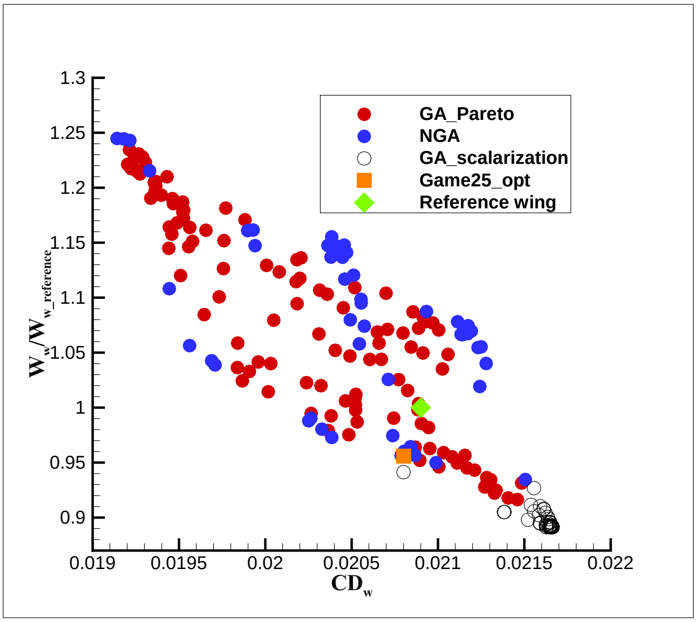



**Figure 3.22:** Results comparison ($W_w$-$C_{Dw}$) among the three optimization approaches.

$$F_{obj} = F_{obj\_1} \cdot k_w + F_{obj\_2} \cdot k_{CDw} \cdot s_{CDw} - F_{obj\_3} \cdot k_{CL} \tag{3.16}$$

In Eq. 3.16, the weight $k_{CL}$ represents the importance of the wing maximum lift coefficient in the optimization process. The comparison between all the results performed by the NGA and those calculated by GA and Pareto front algorithm is shown in Fig. 3.22, Fig. 3.23 and Fig. 3.24. Since there are three objective functions which vary simultaneously the final results should be represented on a 3-axis graph but, to show the comparisons as well as possible, three cutting planes are presented in the figures abovementioned, focusing the attention on two pay-off functions at once. The NGA better solution (the orange square in the three figures, referred to the wing planforms shown in Fig. (3.21)) always lies on the Pareto front, leading to comparable results among different approaches.

### 3.2.3 Three players wing optimization applied to transport jet aircraft configuration

The second application of the optimization task performed by the candidate was referred to a jet swept wing 3 players optimization using the NGA algorithm and then comparing the results obtained to a classical Pareto front and single objective scalarization (GA). The TLAR of the aircraft are given in Table 3.11,

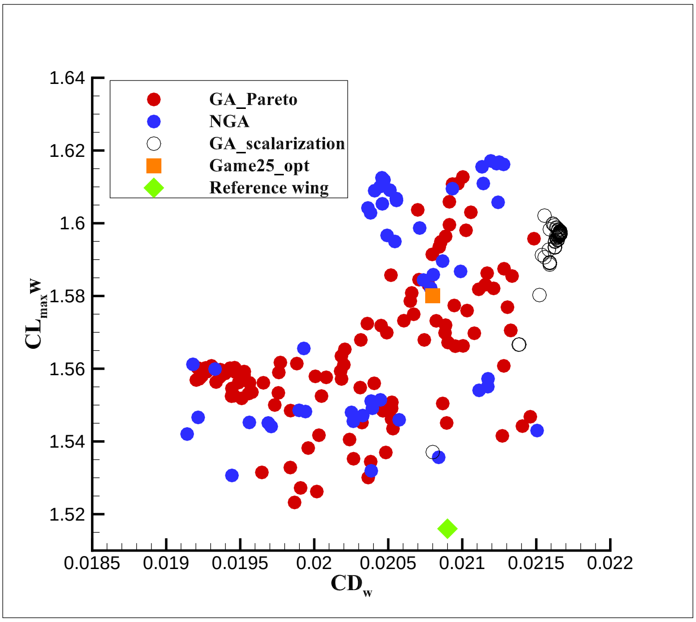



**Figure 3.23:** Results comparison ($C_{Lmax}$-$C_{Dw}$) among the three optimization approaches.

and the resulting wing design from DC-1 is characterized by a reference area equal to 82.7 square meters and a sweep angle at the quarter of the chord equal to 30 degrees. The DC-1 main wing parameters are summarized in Table 3.12 and the wing is shown in Fig. 3.25.

| Description | Value |
|---|---|
| Range | 3600 km |
| Cruise Mach number | 0.78 |
| Initial Climb altitude | 11 000 m |
| Number of passengers | 90 pax |
| Take-off field length | 1500 m |
| Approach speed | 130 kts |
| A/C configuration | Low-wing, wing-mounted engines |

**Table 3.11:** Top level aircraft requirements.

A multi-objective optimization was performed involving five design variables: the taper ratio ($\lambda$), the maximum mean thickness percentage (t/c), the aspect ratio ($AR$), the leading-edge sweep angle ($\Lambda_{LE}$) and the wing area ($S_w$). The three objective functions (players) are the wing drag coefficient (computed according the Aerodynamic Cluster), the wing weight (computer according to the

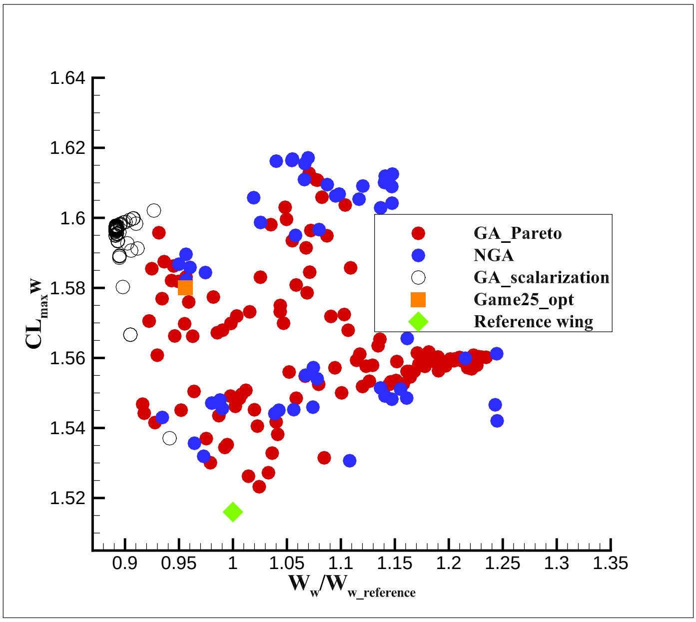



**Figure 3.24:** Results comparison (($C_{Lmax}$-$W_w$) among the three optimization approaches.

| | b(m) | $C_{root}$(m) | $\Lambda_{LE}$(deg) | Taper Ratio | t/c | $S_w(m^2)$ |
|---|---|---|---|---|---|---|
| **Reference Wing** | 28.01 | 6.39 | 30 | 0.164 | 0.13 | 82.7 |
| **$C_{Dw}$ - Wing Weight @ $C_L = 0.49$** | 0.0254 - 4887kg | | | | | |

**Table 3.12:** Reference wing characteristics.

Structural sizing and Weight Cluster) and the wing maximum lift coefficient in clean configuration (computed again according to the Aerodynamic Cluster). During each loop the five design variables and the resulting objective functions change. This application is also described by Eq. 3.17 where the first number represents the number of players involved, inside the curly brackets the upper and lower values of the five cards of the game (AR, t/c, $S_w(m^2)$, $\Lambda_{LE}(deg)$, $\lambda$ respectively) and finally the specific players (objectives). The 3 players could play with these 5 cards, with player 1 optimizing the wing drag coefficient, player 2 optimizing the wing weight and player 3 optimizing the maximum lift coefficient in clean configuration.

$$\Gamma = \langle 3; \{9 - 10.5\}, \{0.125 - 0.138\}, \{75 - 95\}, \{30 - 34\}, \{0.12 - 0.17\}; C_{Dw}, W_w, C_{Lmax_w} \rangle \tag{3.17}$$

Figure 3.26 shows the best solution in terms of wing planform compared

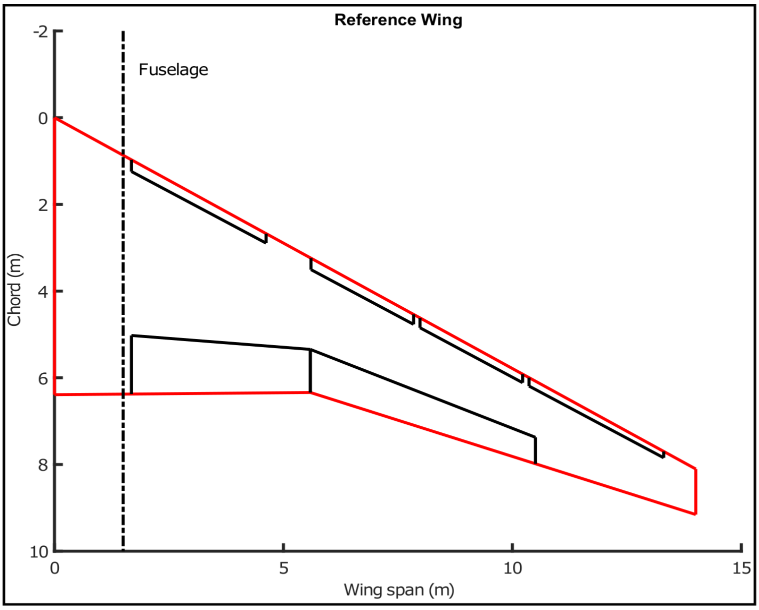



**Figure 3.25:** Reference wing planform.

| | AR | $\Lambda_{LE}$(deg) | b(m) | $\lambda$ | t/c | $S_w(m^2)$ |
|---|---|---|---|---|---|---|
| **Reference Wing** | 9.43 | 30 | 28.01 | 0.164 | 0.13 | 82.7 |
| **$C_{Dw}$ - Wing Weight - $C_{Lmax_w}$** | 0.0254 - 4887 (kg) - 1.39 | | | | | |
| **Game 35 (NGA)** | 10.5 | 33.69 | 28.05 | 0.17 | 0.138 | 75.25 |
| **$C_{Dw}$ - Wing Weight - $C_{Lmax_w}$** | 0.0240 - 4851 (kg) - 1.53 | | | | | |
| **GA best solution** | 10.5 | 33.69 | 28.1 | 0.17 | 0.138 | 75 |
| **$C_{Dw}$ - Wing Weight - $C_{Lmax_w}$** | 0.0240 - 4838 (kg) - 1.53 | | | | | |

**Table 3.13:** Comparison between AGILE DC-1 wing and the best solution of NGA and GA applications with 3 players.

with the reference wing planform (red line).

Table 3.13 provides the comparison between the reference wing planform, the best solution chosen at the end of the NGA optimization and the best solution obtained through GA scalarization. The latter solution was obtained by associating AR and $\Lambda_{LE}$ to $F_{obj\_1}$, $S_w$ and (t/c) to $F_{obj\_2}$, $\lambda$ to $F_{obj\_3}$. As can be seen, the optimum solution simultaneously improves the drag coefficient (reduction of about 14 drag counts), the wing weight (reduction of about 40 kg) and increases the maximum achievable lift coefficient (increase with 0.12). Although the best solution obtained using the GA scalarization approach is similar to the one obtained using NGA, the solution obtained using the scalarization approach is largely dependent on the values of the $k$ weigths and does not take into account the association among variables and objective functions.

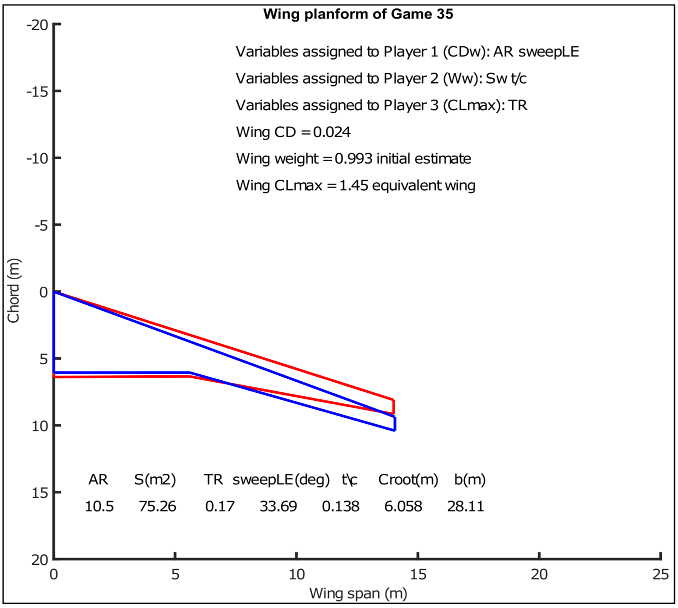



**Figure 3.26:** Wing planform (Game 35) three players' optimization for reference wing (blue) and optimized wing (red).

A comparison of all the NGA points (60 games) with a typical Pareto frontier and scalarization optimization approach is shown in Fig. 3.28 and 3.29 highlighting that the NGA points are characterized by a better spread compared to the GA scalarization points which are only located in a specific portion of the feasible area bounded by the Pareto front. It is useful to underline that Fig. 3.28 and 3.29 only show a cutting plane of the multi-objective optimization among the three players/variables involved.

This application showed that the Nash game theory coupled with a typical genetic evolutionary algorithm (NGA) is a viable optimization strategy because it firstly permits a more realistic association among variables and objective functions and secondly it reduces the computational time. Moreover, the reduced distance between NGA solution points and the Pareto front demonstrates the reasonableness and the feasibility of the results obtained.

At the end of the described application it is necessary to make a consideration. The three players optimization involved the maximum achievable wing lift coefficient in clean condition as one of the objective variables. As stated in Eq. 3.18, the $C_{Lmaxw}$ depends on maximum lift force ($L_{max}$), dynamic pressure ($q$) and wing area ($S_w$). In particular, $C_{Lmaxw}$ maximization can be obtained by decreasing the $S_w$ (see. Eq. 3.18) which can lead to a decrement of the $L_{max}$ (see Eq. 3.19). So, a stricter definition of the objective function should be provided by considering the factor ($C_{Lmax_w} \cdot S_w$) as a third player of the

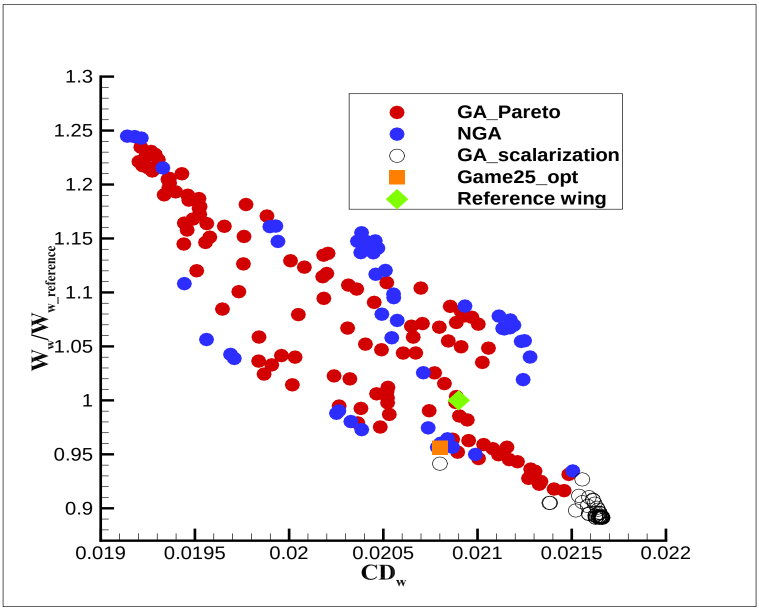



**Figure 3.27:** Results comparison among the three optimization approaches (data referred to the equivalent wing). $\mathbf{C_{Dw}}$ vs $\mathbf{W_w/W_{w\_ref}}$.

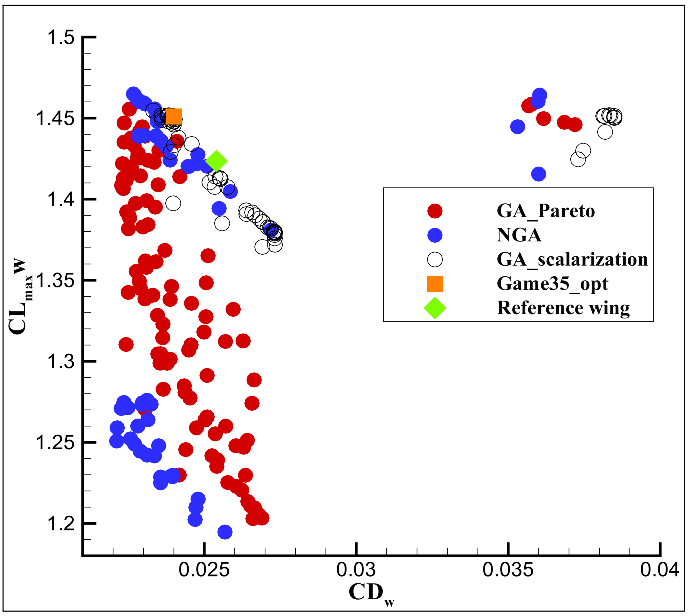



**Figure 3.28:** Results comparison among the three optimization approaches (data referred to the equivalent wing). $\mathbf{C_{Lmax_w}}$ vs $\mathbf{C_{Dw}}$.

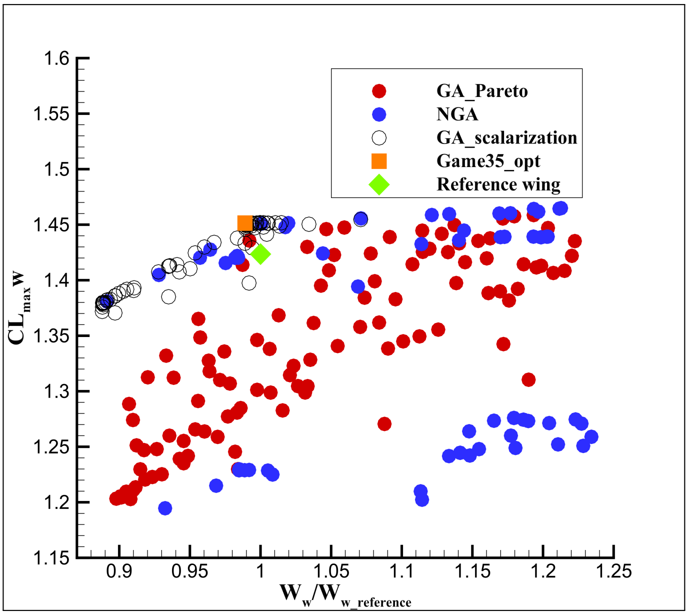



**Figure 3.29:** Results comparison among the three optimization approaches (data referred to the equivalent wing). $\mathbf{C_{Lmax_w}}$ vs $\mathbf{W_w/W_{w\_ref}}$

optimization task. Future works and applications will count for this aspect.

$$C_{Lmax_w} = \left(\frac{L_{max}}{q \cdot S_w}\right) \tag{3.18}$$

$$L_{max} = C_{Lmax_w} \cdot q \cdot S_w \tag{3.19}$$

The goal of this application is to show that the Nash game theory coupled with typical genetic evolutionary algorithm, NGA, is a viable approach to use in the optimization field in order to: firstly, allow a more realistic association among variables and objective functions; secondly reduce the computational time. Moreover, the reduced distance between NGA solution points and the Pareto front attests the reasonableness and the feasibility of the results obtained. Finally, a verification of the computational time between the Pareto front, a single game of the NGA, and the GA scalarization approach has been performed on a laptop equipped with a single CPU (2.0 GHz). The elapsed time for a single NGA solution point for the 2 players application is equal to 5.14 seconds, for a single scalarization GA solution point is 5.91 seconds, and for the Pareto front is 7.57 seconds. The larger the number of variables or objective functions, the larger the computational time that is saved. In this case correctly design based assignment of the NGA variables to the players leads to a reduction higher than 30% in terms of computational time. As future outlook is foreseen


**Figure 3.30:** New methodology developed by the author and integrated in RCE environment.

the introduction of a higher fidelity models, like those of computational fluid dynamics (either panel based or grid resolved) and structural mechanics, to predict with a larger degree of accuracy the figures listen in Eq. 3.7 -3.12, and surrogate-based optimization strategy, to reduce evaluation function time. Moreover, the simplified models behind Eq. 3.7 -3.12 could still be applied, and extended by means of the inclusion of term-specific uncertainty factors affecting each of the figures building up the objective functions used in this optimization application.

The methodology proposed was not just a single application performed by the author at UniNa, but was also fully integrated in RCE, as shown in Fig. 3.30, and made available to other partners through a remote service request. To make it possible, a XML input file was created to collect user's initial data and a python script was implemented to read the input file, run the NGA methodology in batch mode making a sub-call to Matlab script and collect the output of the optimization task in a new CPACS file under the `<toolspecific>` tag as indicated by AGILE best practice.

This way of working allowed other specialists to perform a wing optimization related to their own tasks in a fully automated way just by making a Brics call to NGA, GA or Pareto tools available in a main workflow assembled by the author.

## 3.3 Design Campaign 3

The third design campaign was named: "Novel Configurations". In this phase, the combined best practice methods were applied to novel aircraft configurations in order to investigate how the developed optimization techniques can face different physical problems also related to non-conventional aircraft configurations. In particular, the proposed aircraft design architectures are the following:

a) Strut-Braced Wing (SBW)

b) Box-Wing (BW)

c) Blended Wing Body (BWB) with/without BLI technology

d) Unmanned Aircraft Vehicle (UAV)

e) Wing Mounted engine advanced turboprop (WM)

f) Rear Mounted engine advanced turboprop (RM)

This phase was mainly focused on delivering extended knowledge on optimization techniques applied to large MDO frameworks in which disruptive aircraft configurations were involved. All the achievements in terms of new MDAO approaches, enhanced MDO procedures and results of have been published in the AGILE *Open MDO Test Suite*.

In Chapter 4, a detailed description of the third design campaign and of the work package related to it will be provided.

## Synthesis of the chapter

- AGILE Design Campaign have been carefully described
- AGILE Design Campaign 1 have been focused on the formulation of a reference aircraft obtained by means of overall aircraft design tools. This activity has deeply involved the author in achieving Design Challenge L0 solutions.
- A first AGILE distributed design mock-up has been implemented in collaboration with other partners.
- **Information collection** and **CPACS compliancy** sessions have required a large effort for the author in order to start the development of several tools related to aerodynamic, ground performance and aircraft directional stability estimation.
- In DC-1, nine tools have been developed and tested by the candidate on the reference aircraft configuration. Tools have been also shared with other partners and a MDA workflow have been implemented in RCE
- Results obtained by performing a DOE, based on MDA workflow, have been used to build up aerodynamic, on-board systems, structural, weight and mission performance cluster to speed up the optimization process.
- In DC-2 several new and/or improved optimization methodologies have been developed.
- A new wing optimization technique (NGA) coupling the Nash game theory with the genetic algorithm have been developed and tested by the author on different wing architecture and considering multiple objective functions. The methodology has also been fully integrated in RCE and have been made available to other partners.
- Achievement and tools coming from DC-1 and DC-2 have been applied to several disruptive aircraft configuration in the DC-3.

# Chapter 4

# AGILE WP4: Turboprop aircraft optimization

## 4.1 Introduction

The third design campaign aimed to show the capability of the technologies developed in the previous design campaigns, by applying them to six different aircraft architectures described in Sec. 3.3. In particular, this chapter is addressed to present and describe the task T4.6 of AGILE WP4 [80], where the candidate, as part of UniNa, was involved in analyzing and optimizing two different innovative turboprop aircraft configuration: i) a wing mounted engine (WM) configuration and ii) a rear mounted engine (RM) configuration. In the DC-3 UniNa, in the person of the author and its supervisor, covered both the Integrator and the Spcialist role. LEONARDO company as Architect of task T4.6 released the TLAR and the main objectives of the design and optimization activities. UniNa as Integrator of T4.6 and LEONARDO selected, from the *Tools catalogue*, all the disciplines (tools) available and needed to accomplish the prescribed goals of the T4.6.

The entire process, from the TLARs until the whole MDAO, was implemented in KE-Chain through which was possible to build up the workflow step by step. In the first steps the Architect and the Integrator inserted the TLAR and information about all the involved participants and uploaded, on the framework, the common aircraft definition file and tools chosen from the catalogue. The heterogeneous team of experts was composed by a group of Specialists in different disciplines which took care of adding input and output files related to the tools they provided and the Integrator was in charge of assembling the entire workflow starting from an MDA to DOE until the MDO carefully choosing the design variables, the constraints and the objective variables or functions. In order to make the remote, collaborative and distributed design possible, NLR was appointed as Collaborative Engineer and responsible in providing BRICS software.

Starting from the TLARs summarized in Table 4.1, LEONARDO, as task's Architect, also required to comply several aerodynamic requirements listed in. Table 4.2 and in Table 4.3.

| | *Metric* | *Imperial* |
|---|---|---|
| **Design Range** | 2222.4 km | ≥1200 nm |
| **Design payload** | 9540 kg | 21032 lbs |
| **Max. Payload** | 11590 kg | 25552 lbs |
| **PAX** | 90 pax @ 106 kg | 90 pax @ 233.7 lbs |
| **MLW (% MTOW)** | 97% | |
| **Cruise Mach** | 0.56 @ 7620 m | 0.56 @ 25000 ft |
| **Maximum Operating Altitude** | 7620 m | 25000 ft |
| **Climb time** | 13 min | |
| **TOFL (ISA, SL, MTOW)** | ≤1500 m | ≤4920 ft |
| **Landing distance** | ≤1500 m | ≤4920 ft |
| **Max. operation speed** ($V_{mo}/M_{mo}$) | 270 KCAS/0.60 | |
| **Dive Mach number** ($M_d$) | 0.64 | |
| **Fuselage diameter** | 3.53 m - 5 abreast | 139.17 in - 5 abreast |
| **Service life** | ≥110000 cycles | |
| **Fuel reserves** | 5% of Block Fuel - 100 nm Alternate | |
| **Holding** | 30 min @ 457 m | 30 min @ 1500 ft |
| **A/C configuration** | High-wing (wing-mounted engines), Low wing (rear mounted engines) | |
| **nEngine** | 2 - TurboProp | |
| **Design Objective** | Minimum D.O.C. | |
| **External Noise** | CHAP14 - 15 epndb | |

**Table 4.1:** Regional turboprop aircraft configurations TLARs provided by Leonardo company.

| WM Reference Aircraft | | |
|---|---|---|
| Condition | $\mathbf{C_L}$ | **Efficiency** |
| **Climb** | 0.75 | 16 |
| **Cruise** | 0.51 | 16.5 |
| **OEI** | 1.30 | 14.5 |
| | | $\mathbf{C_{Lmax}}$ |
| **TakeOff & Approach** | | 2.3 |
| **Clean** | | 1.7 |
| **Landing** | | 3.0 |

**Table 4.2:** Aerodynamic TLARs for WM configuration.

In order to provide to the consortium, and in particular to the Specialists and Integrators involved in T4.6, the preliminary aircraft reference configurations, LEONARDO took also care the preliminary aircraft sizing in terms of mass breakdown and main aircraft geometrical characteristics as summarized in Table 4.4 and in Table 4.5.

The author was initially in charge of collecting all the information provided about the turboprop configurations, using them to create two new different

| RM Reference Aircraft | | |
|---|---|---|
| Condition | $\mathbf{C_L}$ | **Efficiency** |
| **Climb** | 0.8 | 16.5 |
| **Cruise** | 0.55 | 17.5 |
| **OEI** | 1.16 | 14 |
| | | $\mathbf{C_{Lmax}}$ |
| **TakeOff & Approach** | | 2.3 |
| **Clean** | | 1.7 |
| **Landing** | | 3.0 |

**Table 4.3:** Aerodynamic TLARs for RM configuration.

| WM and RM Reference Aircraft | | |
|---|---|---|
| Wing | **WM** | **RM** |
| **AspectRatio** | 12 | |
| **Area** | 78 $m^2$ | |
| **Span** | 30.6 m | |
| **Root chord** | 3.33 m | |
| **Kink chord** | 3.22 m | |
| **Tip chord** | 1.16 m | |
| **mac** | 2.74 m | |
| Fuselage | | |
| **Overall length** | 3.33 m | |
| **Diameter** | 3.33 m | |
| Vertical Tail | | |
| **Area** | 18 $m^2$ | 23.53 $m^2$ |
| **Span** | 5.38 m | 5.27 m |
| Horizontal Tail | | |
| **Area** | 14.72 $m^2$ | 32.8 $m^2$ |
| **Span** | 8.18 m | 9.06 m |

**Table 4.4:** Aircraft main geometrical characteristics of WM and RM configurations.

CPACS files, one for each turboprop architecture, and importing them in TiGL Viewer allowing the generation of the reference aircraft geometries as shown in Fig. 4.1 and in Fig. 4.2.

Once the CPACS files were ready to be used and the data from the partners were provided to the Integrator, all the steps described in 2.4 from Fig. 2.31 to Fig. 2.35 were accomplished in KE-chain in order to assemble a complex and complete MDAO toolchain both for WM and RM architecture. In the next section a clear description of the phases concerning the setting up of the real

| Weights | WM | RM |
|---|---|---|
| **MTOW** | 35380 kg | 37500 kg |
| **MZFW** | 33020 kg | 35060 kg |
| **MOEW** | 21430 kg | 23470 kg |
| $\mathbf{M_{payload}}$ | 11590 kg | 11590 kg |
| **MLW** | 34319 kg | 36375 kg |
| $\mathbf{M_{F}}$* | 2360 kg | 2440 kg |
| *Related to design payload condition and based on $SFC = 0.36lb/hph$ | | |

**Table 4.5:** Preliminary mass breakdown for WM and RM configurations provided by Leonardo company.

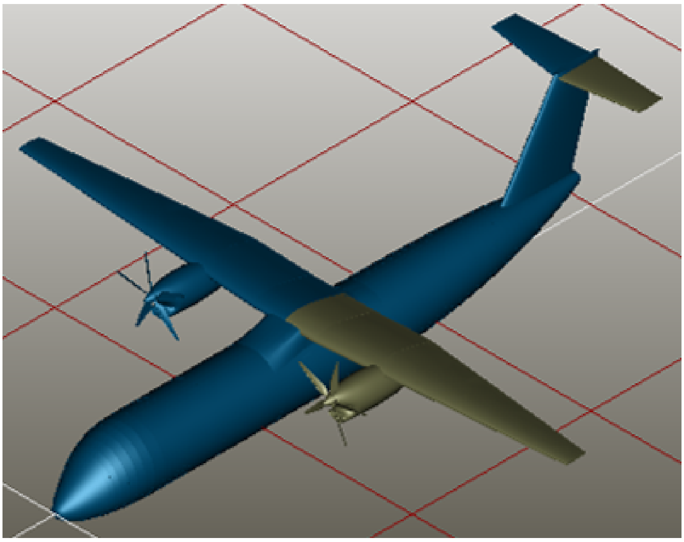


**Figure 4.1:** WM configuration in TiGL Viewer.

workflow made of tools available in the consortium is provided.

## 4.2 Setup of the MDAO architecture

The first step concerned the set-up of MDAO workflows, one for each configuration, which was composed by several and specific partners that can contribute to improve the accuracy of the analyses related to a specific discipline thanks to their expertise. At this stage, the Architect (LEONARDO) and the Integrator (the candidate as part of UniNa) identified, among the partners, tools and specialists to be involved in building-up a suited complete workflow. Each partner, as a specialist in one or more disciplines, provided his own competence(s) (low/medium level of fidelity L0-L1) by sharing, through Brics, his

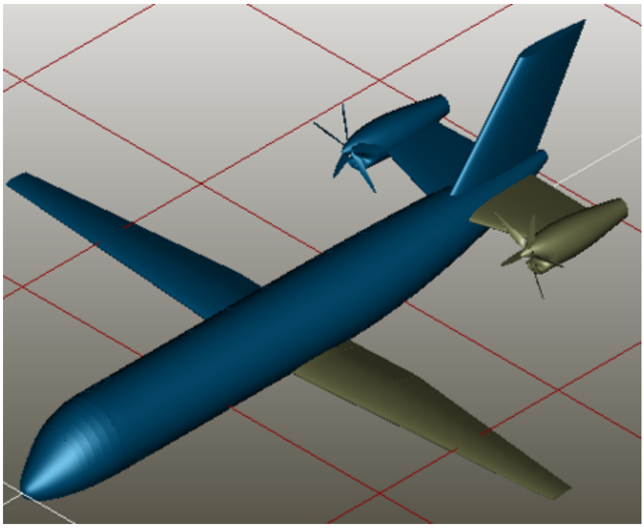


**Figure 4.2:** RM configuration in TiGL Viewer.

own computer codes and methods regarding fields such as aerodynamics, aircraft weights estimation, mission analysis, costs and so on. Each one of these CPACS-compliant competences were stored in different tools, black boxes [48], which can be run in a remote manner and then results obtained were collected to update the reference CPACS file automatically and used by other partners as input for their tools. The MDAO set up following this approach is enough flexible to analyze an arbitrary number of CPACS files (aircraft configurations) allowing the data exchanging between partners through the CPACS schema. The workflow integration was automatically generated using KE-Chain (step 2.2), KADMOS and VISTOMS packages by importing the partners' design competences one by one creating a tool repository useful to generate the corresponding RCG inspectable by means of XDSM, Sankey Diagram or Edge Bundle technology. After KE-chain step 2.2, a real MDAO toolchain needs to be created. The author, as Integrator, took care of it carefully planning how the toolchain should have worked in terms of input and output data exchanging order, tools in the loop addressed to reach a certain convergence, design variables, constraints and objective functions to monitor.

The conceptual MDAO workflow, depicted in Fig. 4.3, can be divided in several steps described as follows [39]:

1. Aircraft initialization: the starting point is the initialized aircraft, as described in Sec. III. The CPACS baseline file describes the initial condition of the aircraft (black box in Fig. 3).
2. Engine deck provider: the baseline engine deck is provided and correctly integrated into the baseline file (see red box in Fig. 3), according to the

engine top level requirements.

3. Aerodynamic competences branches: the overall aerodynamic database is performed according to different levels of fidelity. The results are updated into the CPACS file of the aircraft and passed to the next competences (see red dashed box in Fig. 3).

4. Aerostructural sizing, weight competence: here the aircraft structural sizing is performed according the certification load cases. The aircraft empty weight is updated, and results passed to the next tools. Different levels of fidelity are provided as shown in the dashed blue box.

5. On-board-system design: here OBS are designed and systems masses updated. Results are passed to the next tools and to Engine Deck box to account for power-off-takes.

6. Performance and mission analysis: overall aircraft performance are computed; mission profile is simulated and block fuel is evaluated.

7. Mass update and rubber engine tools: aircraft mass breakdown is updated according steps (4-5-6); engine deck is scaled according to aircraft thrust to MTOW ratio.

8. Repeat steps 4 to 7 until MTOW value has reached the convergence.

9. Stability & Control calculation.

10. Costs & Emission Calculation.

The workflow represented in Fig. 4.3 was then transformed in a real one in KE-chain which taking into account for all provided tools and cross connections between them. Applying a specific MDA architecture (i.e. *Converged MDA*) to the RCG, it was possible to generate the FPG as shown in Fig. 4.4

In order to provide a clearer overview of the workflow assembled, it is useful to give some information about the tools shared by the partners.

A) **Engine competence**

The engine design was accomplished by CIAM partners, based on Architect specifications. Moreover, a "rubber" engine tool was developed and provided by UniNa, in the person of the candidate and his supervisor, to re-size the engine deck during convergence loop according to aircraft weights variations. Indications about the scaling factors was provided by CIAM engine designer. The adopted assumption was that the fuel flow and the installed thrust can be linearly scaled with respect to MTOW, while geometrical dimensions and engine dry mass vary with the square root of the scaling factor. Making these assumptions would lead to an engine performance estimation inaccuracy lower than 1.5%.

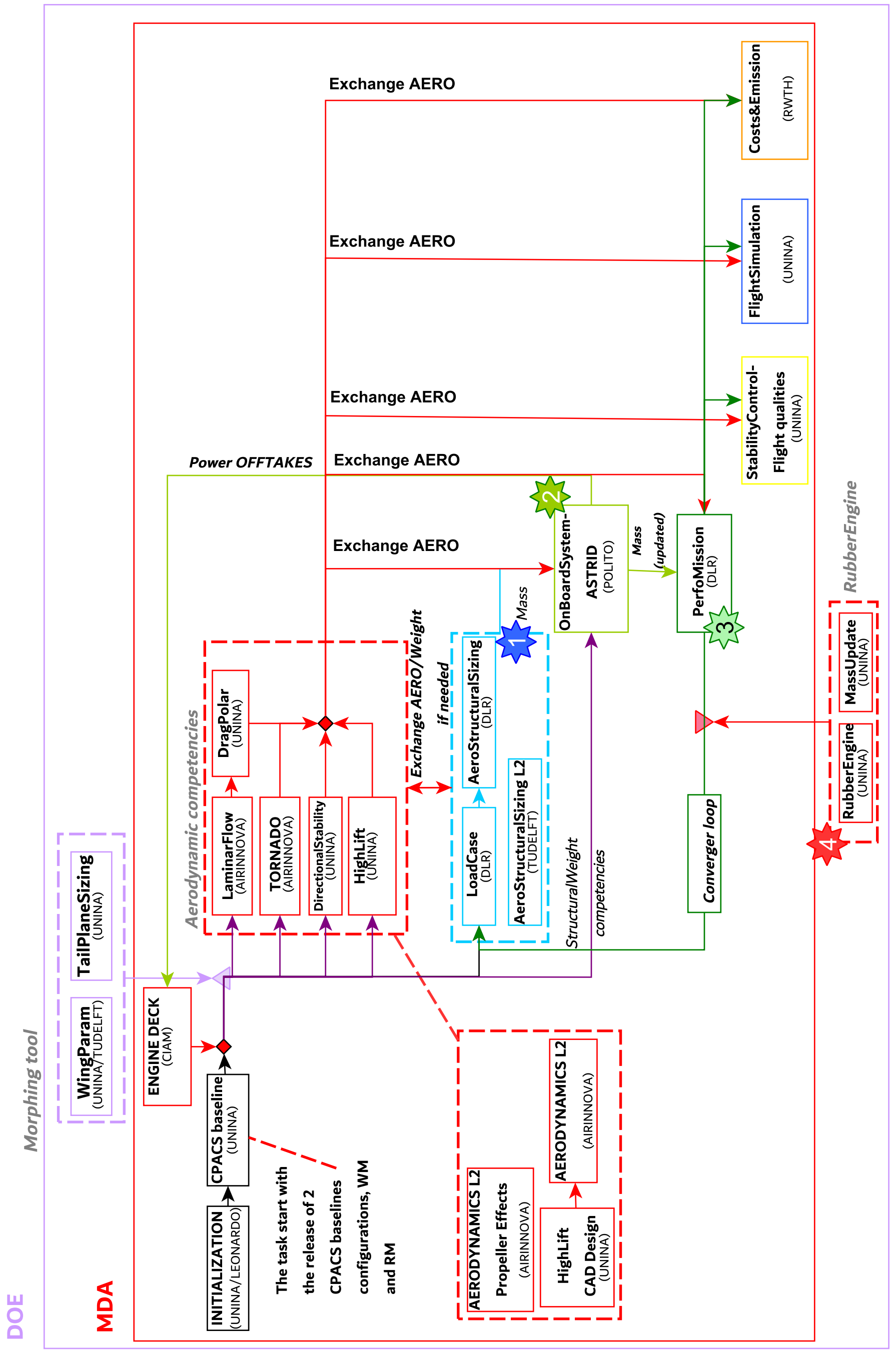



Figure 4.3: MDAO workflow sketch.

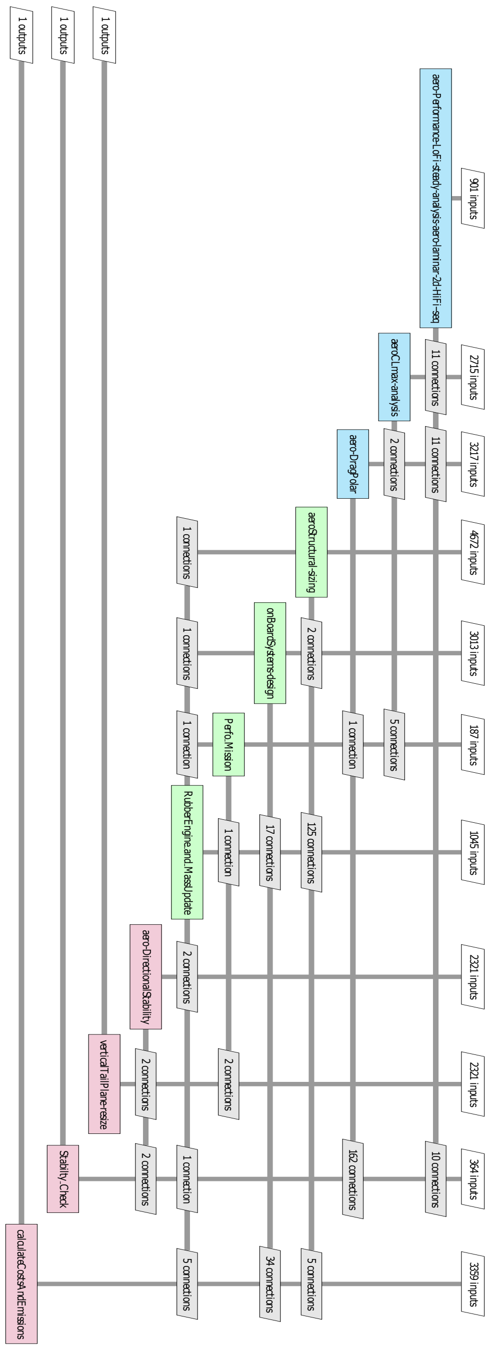



**Figure 4.4:** FPG of the MDA workflow in Ke-chain environment. XDSM visualization through VISTOMS package.

B) **Aerodynamic competence**

AIRINNOVA provided a multi-fidelity and multi-method combined aerodynamic analysis for the both WM and RM configurations. The tools were integrated into the MDAO process to provide i) low-fidelity aerodynamic analyses by using TORNADO [81], ii) laminar flow airfoil analyses making use of MSES [82], iii) high-fidelity aerodynamic analyses by means of SU2 software [83]. The author developed and provided several tools that covered different area of study such as i) low speed aerodynamic tools which evaluate wing $C_{Lmax}$ in clean, take-off and landing condition; drag contribution due to flaps and slats deflection in take-off and landing condition; pitching moment contribution due to flaps to the aircraft longitudinal stability. These tools were already applied to a turboprop aircraft optimization process [61, 84] and to the development of an improved high lift prediction method [85]; ii) complete aircraft drag polar estimation taking into account the friction drag contribution not estimated by low-fidelity tools provided by AIRINNOVA.

C) **Structure competence**

The aircraft structure design and analysis competence was also divided into low-fidelity and high-fidelity analyses, performed by DLR and TuDelft partners respectively. The low-fidelity aerostructural remote service was composed by several physics based disciplinary analysis and design modules suitable for the preliminary development of the airframe structures [86], such as i) loads cases generation module, ii) in-house developed aerodynamics solver for loads analysis, iii) in-house aeroelastic FEM modeler and FEA solver [87], iv) estimation module of the secondary airframe masses. All these tools are fully automated and CPACS compliant. The main output includes the mass breakdown of the structural components, as well as detailed aerostructural sizing results. High-fidelity service was related to the possibility of considering wing composite materials.

D) **On board systems competence**

The on-board systems design and analysis was performed by POLITO, based on LEONARDO specifications and engine characteristics, by means of ASTRID tool [51, 88]. It uses semi-empirical and physic-based models, some of them are dedicated to main equipment design such as actuators for flight control and landing gear movement, avionics, fuel pumps, electric generators and converters, hydraulic pumps and others OBS main components and the other models are more focused on the whole system design. For this specific application an All Electric Aircraft (AEA) architecture for OBS was selected, as prescribed by LEONARDO. The output of ASTRID tool concerns the main OBS masses and power off-takes.

E) **Performance competence**

Aircraft ground [58] and mission performance were performed by the candidate. The module evaluates the aircraft mission performance by means of empirical methodologies suited for each specific mission phase. The tools receives in input aerodynamic, engine, weight and geometrical data through the CPACS data exchange format and gives in output results such as the block fuel mass for given mission and reserve segments, the required runway length for take-off and landing and a scaling factor for engine sizing using a "rubber engine" principle, trajectory information.

F) **Stability & Control competence**

The candidate also gave its contribution to the stability and control competence, by evaluating the vertical tail contribution to the aircraft directional stability [41] and the fuselage contribution to the complete aircraft longitudinal stability [42]. Furthermore, by creating an additional python tool, also the vertical tail is re-sized taking into account both for the minimum control speed constraint and the yawing moment calculation due to engine thrust in one engine inoperative condition during the take-off phase. Finally, making use of the updated weight and balance breakdown values, a specific tool was used to evaluate stability and control derivatives, neutral point and static margin.

G) **Costs & Emissions competence**

RWTH oversaw the aircraft costs and emissions. Within T4.6, the main focus was set on the production and operational phase. With regard to the costs, RWTH Aachen's module comprises both non-recurring (e.g. development, testing and test facilities) and recurring (indirect and direct) costs for an aircraft's life cycle using low-fidelity methods [89]. Concerning emissions evaluation, the tool proposed by RWTH Aachen calculates aircraft life cycle emissions starting from the development phase to production, operations and finally end-of-life. The implemented methodologies use both semi-empirical as well as physics-based calculations to account for the different emissions throughout the life cycle. Within the module not only the amount of emissions, but also their climate impact considering different climate metrics such as Average Temperature Response (ATR) and Absolute Global Warming Potential (AGWP) are calculated using a climate model [90].

H) **Morphing tool competence**

The morphing tool competence has been provided by TUDelft University through a tool named sCAM. It is able to receive as input data some wing geometrical variables such as span, sweep angle at leading edge, taper ratio, dihedral angle, root chord, location and to create a new CPACS baseline file automatically. In this way it is possible to set up a DOE toolchain in which the sCAM tool is the one that provides the new CPACS input file, based on the value of the specified design variables, to all the tools involved in the workflow. Moreover, the developed tail planes sizing tool is used together the sCAM tool modified and customized by the candidate to modify horizontal and vertical tails according to wing parameters variations.

I) **CAD competence**

At the end of the entire workflow the creation of an aircraft CAD model is necessary to carry out high-fidelity aerodynamic analyses by means of CFD software. To face this necessity UniNa provided an automated CAD maker tool capable of collecting all aircraft geometrical data from the CPACS file and create an accurate CAD model ready to be used in a CFD software. Moreover, the tool is also capable to modify the created model by adding flaps and slats.

J) **Optimizer**

The optimization tools used at the end of the workflow was the MOEA Framework, which is in-house developed at UniNa within the JPAD software library [58, 66]. The MOEA Framework is a free and open source Java library and several algorithms are also provided out-of-the-box, including genetic algorithms, particle swarm and so on. In this specific application the $\epsilon$-NSGAII and OMOPSO are used [92, 93]. The optimization was based on a multi-fidelity response surface, obtained through the above-mentioned DOE.

In the DC-3, among the abovementioned competences, other six tools were developed, tested and provided by the author concerning the aircraft mission and performance calculation, tailplanes sizing, main wing parameters morphing, aircraft stability behaviour check, CAD maker (supervision activity of a master thesis work) and rubber engine and aircraft mass update. These tools, combined to those developed during the DC-1, as shown in Fig. 4.5, were totally used to assemble the entire MDAO workflow as depicted in Fig. 4.7.

The workflow in Fig. 4.4 allow to perform a single design analysis, while a *Converged DOE* strategy allow to perform a certain number of design analyses corresponding to the numbers of variables of the DOE design space.

Referring to Fig. 4.6 which shows the FPG of the DOE architecture, and considering that the engine deck was already stored in the original CPACS file, the yellow boxes cover the conceptual workflow steps from 1) to 3) since the first tool is the morphing tool which is used to vary specific design parameters and the others pre-coupling tools are used to compute aerodynamic calculations; the green boxes cover steps from 4) to 8) in the convergence loop aiming at provide a converged MTOW value; the red boxes cover steps from 9) to 10) using the convergence loop output data as input variables and evaluating both the objective functions and the variables that the integrator wants to check at the end of the entire workflow. The FPG in Fig. 4.6 was then converted in a real workflow by downloading the CMDOWS file from KE-chain and importing it

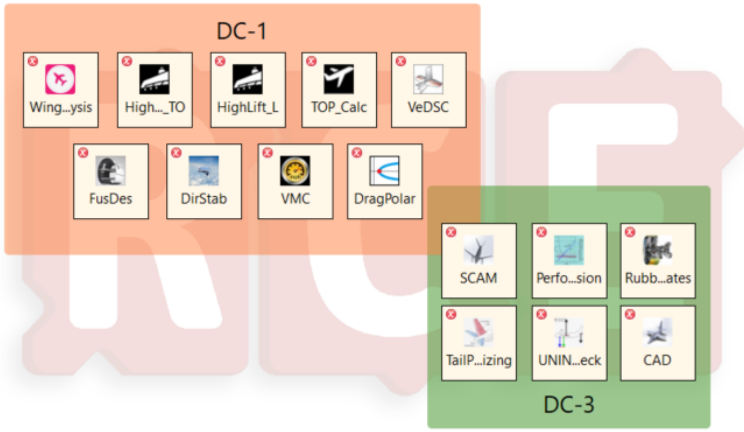



**Figure 4.5:** DC-1 and DC-3 tools developed by the author and fully embedded into RCE environment to make them available to the consortium through a remote service request.

in RCE. It is possible to notice the complete similarity between the conceptual and real workflow illustrated in Fig. 4.7, where the *"black boxes"* labelled by the red small AGILE logo represent the remote services provided by the partners through Brics technology.

The DOE campaign was performed by running the MDA chain $n$ times changing 4 different wing parameters. In this way was possible to create several surrogate models concerning disciplines such as aerodynamics, structures, on-board systems, mission performance and costs.

The last step concerned the assembling and running of the MDO chain taking advantage of the surrogate models generated during the DOE phase. The essential contribution given by the surrogate models is that this *database* can be queried through a simple interpolation by applying to it the RSM [23,64] reducing the time of every optimization task and speeding up the entire design and optimization process. The accuracy of the Response Surfaces can be also improved by performing some high-fidelity calculation regarding certain aircraft configurations.

## 4.3 Applications to WM and RM use cases

The approach described in Sec. 4.2 led to obtain a set of optimized results characterized by a quietly high level of fidelity (from L0 to L3) in a time period quietly short considering that each partner, focused on his own discipline, needs to give just a single permission to provide its remote service even if multiple analyses must be performed and then results are collected automatically. The optimization task can be performed choosing among several optimization algo-

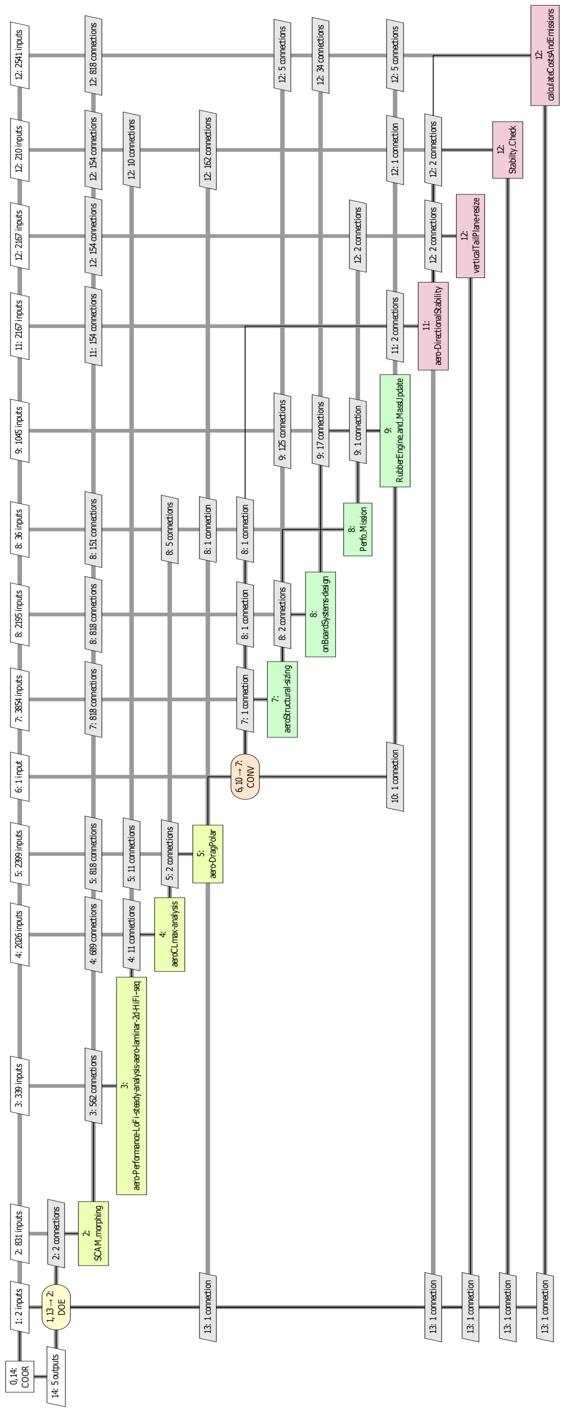



**Figure 4.6:** FPG of the MDA workflow in Ke-chain environment. XDSM visualization through VISTOMS package.

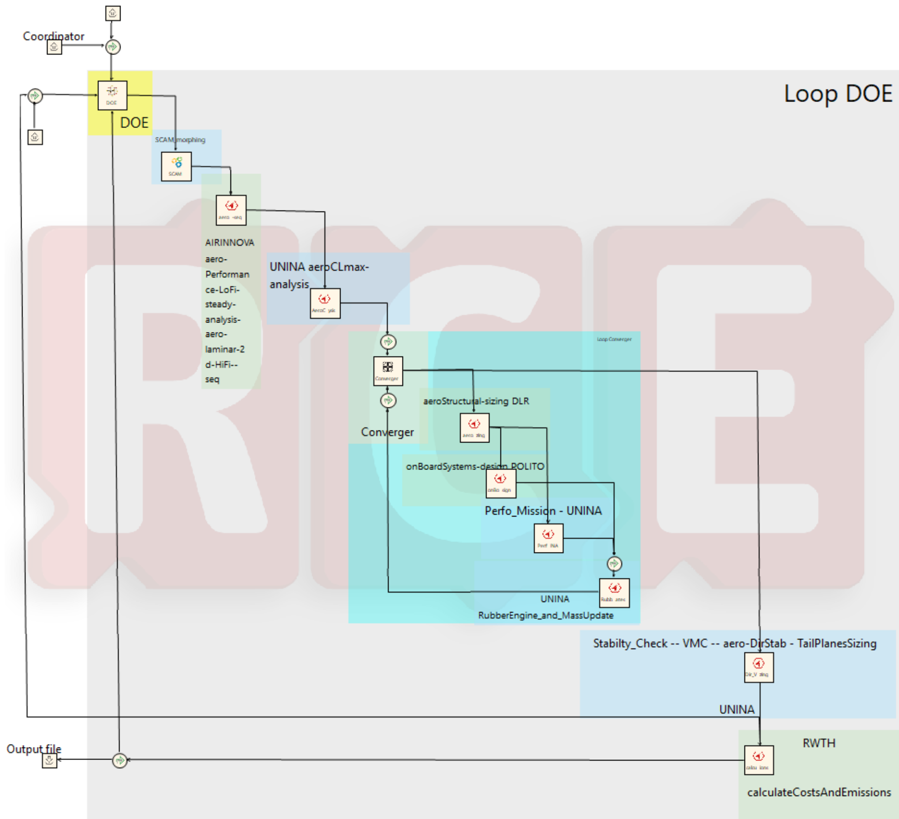



**Figure 4.7:** DOE workflow in RCE environment.

rithms and several workflow architectures. The author in collaboration with its supervisor took care of all the activities related to this application from the definition of the optimization problem to the exploration of the results.

The optimization problem was defined as summarized in Table 4.6. The objective function for both the configurations was the DOC [89]. Moreover, to also meet the CleanSky2 objectives, the Architect added a second objective function defined as the total GWP, defined accordingly Ruijgrok and Van Paassen [91]. A set of constraints was also fixed by the integrator considering TLARs listed in Table 4.1 and the aircraft static stability margin (SSM). The design variables for this application are the main wing planform parameters.

The WM configuration is characterized by high wing with under wing engine installation and T-tail architecture, while the RM has a low wing and rear engine installation on the horizontal tail tip. The wing of the RM configuration was back shifted along x axis with respect to WM configuration in order to face stability problems, due to a high value of the maximum rearward position of

| Objective functions | *min:* |
|---|---|
| | $f_1 = DOC$ |
| | $f_2 = GWP$ |
| **Constraints** | *w.r.t* |
| | SSM $\geq 0.05$ (5% mac) |
| | TOFL $\leq 1500$ m |
| | LFL $\leq 1500$ m |
| | time to climb $\leq 13$ min (from 1500 to 20000 ft) |
| **Design variables** | *by varying:* |
| | $X_{LE_w}$ |
| | $AR_w$ |
| | $\lambda_w$ |
| | $b_w$ |

**Table 4.6:** Optimization problem, variables and constraints.

the center of gravity. The low wing layout was used to avoid the interference between the wing wake and horizontal tail. In Fig. 4.9 and Fig. 4.8 comparisons between potato diagrams and the drag polar of the two configurations are shown.

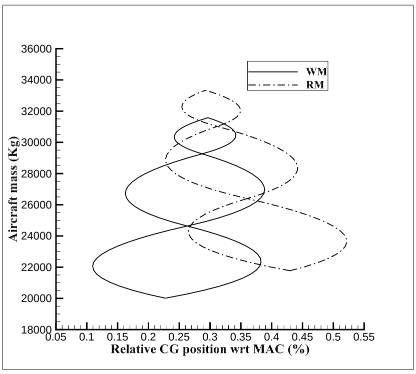



**Figure 4.8:** WM and RM baseline. Potato diagrams comparison.

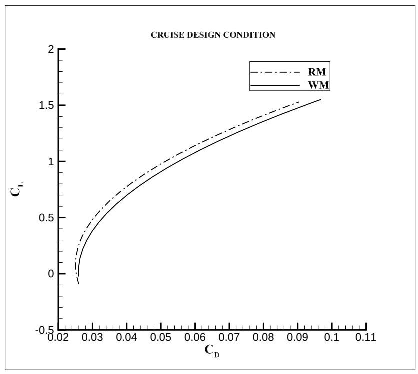



**Figure 4.9:** WM and RM baseline. Drag polar diagrams for trimmed configuration comparison.

Due to different engine installation, the CG shift of WM configuration is in the range 11% - 39%, while for RM is 22.8% - 59% with respect to the *mac*. The RM layout can lead to a very large CG excursion which can also affect aircraft performance. Moreover, a wide CG excursion could imply an oversized horizontal tail to trim the aircraft in the maximum rearward CG position condition resulting in a reduction of the maximum lift capabilities. Meanwhile, in the most forward CG position the longitudinal the SSM could achieve a too much high value penalising the aircraft longitudinal control. This latter will reduce the cruise efficiency affecting the fuel burned and aircraft DOC. One possible solution could be a reasonable reduction of the CG excursion. Although some disadvantages due to rear engine installation, this choice allows to reach several advantages such as cabin noise reduction, community noise reduction because of the absence of T-tail noise reflection, more efficient high-lift system with a possible increase in aircraft maximum lift capabilities due to laminar flow on the main wing reducing the total friction drag.

In cruise condition the friction drag for WM configuration is equal to 262 drag counts while for the RM is reduced by 20 drag counts thanks to the laminarity of the main wing. The SSM of the RM and WM configuration are equal to 2% and 5% of $mac$ respectively. Starting from these configurations, the optimization task was performed matching the constraints described in Table 4.6. Taking advantage of the competence described in Sec. 4.2, the optimization task was carried out by means of response surfaces created for aerodynamic, weight and structures, mission and costs and emissions clusters by using RSM technology, avoiding running the entire toolchain asking for each partner's tool in a remote manner. In particular, the response surfaces were created setting up a DOE using low-fidelity methods obtaining 280 points (aircraft configurations) and were then improved through medium high-fidelity calculations. This approach allows saving time and ensuring a good level of fidelity at the same time. In Fig. 4.10, Fig. 4.11, Fig. 4.12 DOE points obtained by using of low and medium/high fidelity tools are depicted showing the trend comparison with respect to several quantities, variables or objective functions, of interest. Here it is possible to notice that an aircraft chracterized by an increased value of the wing aspect ratio lead to a minimization of the block fuel. The reason lies in a reduction of the induced drag ,due to a high aspect ratio, which allows to save a certain amount of fuel burnt for the entire mission and so a reduced value of the block fuel. As regards the DOC, they are strongly linked to the block fuel value [89] and so are characterized by the same trend. However, the maximum take off weight value has a different behaviour if monitored with respect to aspect ratio variation, but since aspect ratio value is mainly affected by an increasing of the wing span, in this case, is easy to envisage a growth of the wing weight. Moreover, considering the same aspect ratio but a different the wing location along the x axis, the more the wing is close to aircraft rear the more is the area of the tailplanes due to a reduction of the arm between the tailplanes and the wing. This is the reason why in Fig. 4.12 for a constant value of the aspect ratio it is possible to notice a variation in terms of MTOW.

The optimization algorithms OMOPSO and $\epsilon$-NSGAII converged on similar results both in terms of design variables and objective functions reaching different geometrical solutions for WM and RM, as shown in Fig. 4.13 and Fig. 4.14 to minimize DOC and emissions, as listed in Table 4.7, Table 4.8, Table 4.9 and Table 4.10.

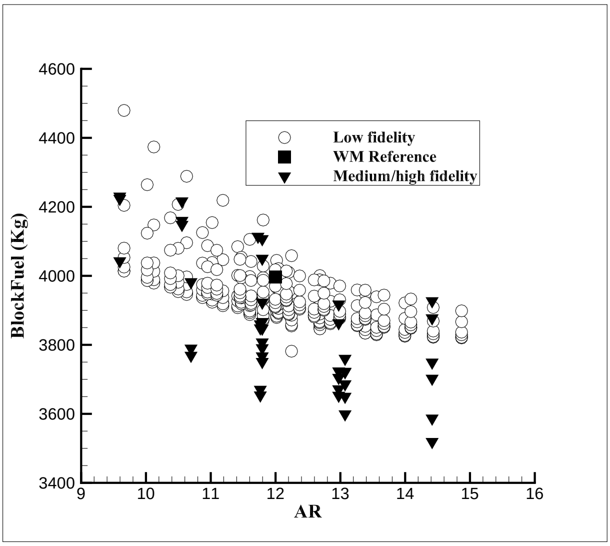



**Figure 4.10:** DOE comparison between low and medium/high fidelity tools. $AR_w$ vs block fuel trend.

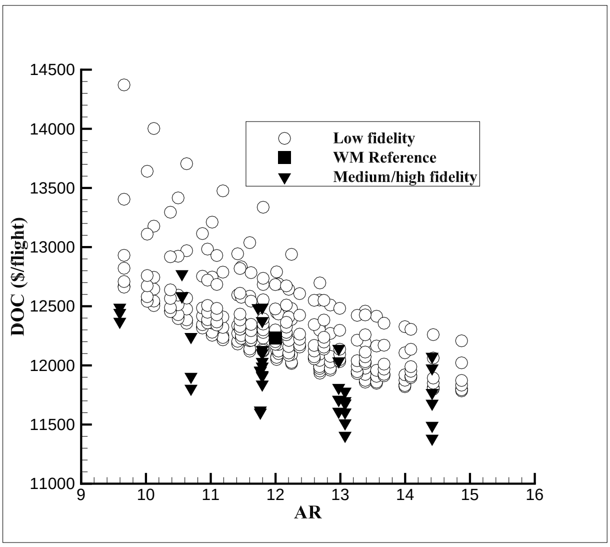



**Figure 4.11:** DOE comparison between low and medium/high fidelity tools. $AR_w$ vs DOC trend.

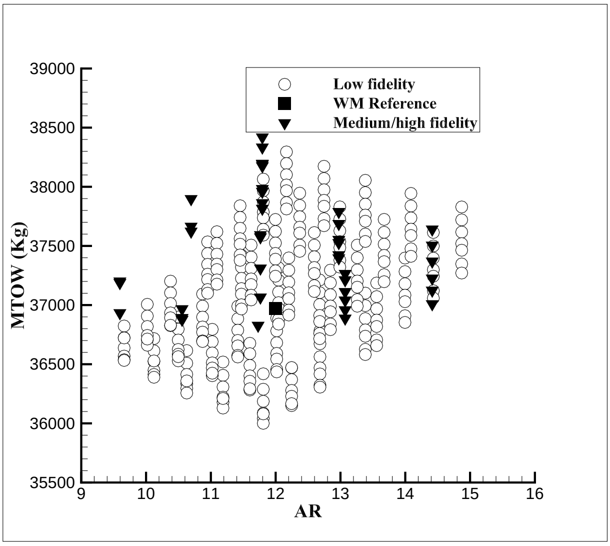



**Figure 4.12:** DOE comparison between low and medium/high fidelity tools. $AR_w$ vs MTOW trend.

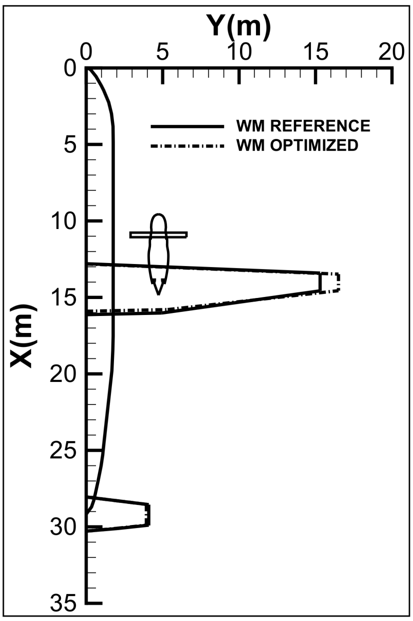



**Figure 4.13:** WM reference and optimized layout comparison.

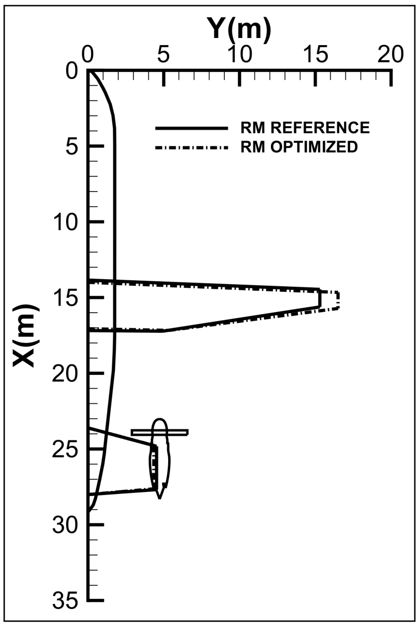



**Figure 4.14:** RM reference and optimized layout comparison.

| | WM Reference | WM Optimized |
|---|---|---|
| Wing | | |
| $\mathbf{AR_w}$ | 12 | 14.79 |
| $\mathbf{S_w}$ | 78 $m^2$ | 78.17 $m^2$ |
| $\mathbf{b_w}$ | 30.6 m | 34 m |
| $\mathbf{c_r}$ | 3.33 m | 3.01 m |
| $\mathbf{c_k}$ | 3.22 m | 2.91 m |
| $\mathbf{c_t}$ | 1.16 m | 1.05 m |
| **mac** | 2.74 m | 2.49 m |
| $\mathbf{X_{LE_w}}$ | 12.80 m | 12.84 m |
| Horizontal Tail | | |
| $\mathbf{S_h}$ | 14.72 $m^2$ | 13.39 $m^2$ |
| $\mathbf{b_h}$ | 8.18 m | 7.81 m |
| Masses | | |
| **Block Fuel** | 3920 kg | 3798 kg |
| **MTOW** | 35496 kg | 35851 kg |

**Table 4.7:** Optimized and reference WM configurations geometry comparison.

| | RM Reference | RM Optimized |
|---|---|---|
| Wing | | |
| $\mathbf{AR_w}$ | 12 | 14.87 |
| $\mathbf{S_w}$ | 78 $m^2$ | 77.74 $m^2$ |
| $\mathbf{b_w}$ | 30.6 m | 34 m |
| $\mathbf{c_r}$ | 3.33 m | 3.00 m |
| $\mathbf{c_k}$ | 3.22 m | 2.90 m |
| $\mathbf{c_t}$ | 1.16 m | 1.04 m |
| **mac** | 2.74 m | 2.47 m |
| $\mathbf{X_{LE_w}}$ | 13.80 m | 13.98 m |
| Horizontal Tail | | |
| $\mathbf{S_h}$ | 32.8 $m^2$ | 29.79 $m^2$ |
| $\mathbf{b_h}$ | 9.06 m | 8.63 m |
| Masses | | |
| **Block Fuel** | 3981 kg | 3915 kg |
| **MTOW** | 37317 kg | 37671 kg |

**Table 4.8:** Optimized and reference RM configurations geometry comparison.

Considering that, for this specific aircraft category, the possible number of flights per day could be equal to 6 and it could work 358 days per year, assuming 7 days for maintenance check A and B, it could be possible to save

| WM layout | | | |
|---|---|---|---|
| | **Baseline** | **Optimized** | **%** |
| **DOC ($/flight)** | 17205.8 | 16829 | 2.1 |
| **DOC (Mln$/year)** | 36.95 | 36.14 | |
| **GWP (kg/flight)** | 13191.6 | 12780.8 | 3.1 |
| **GWP (tons/year)** | 28335.5 | 27453.1 | |

**Table 4.9:** Objective functions comparison for WM configuration.

| RM layout | | | |
|---|---|---|---|
| | **Baseline** | **Optimized** | **%** |
| **DOC ($/flight)** | 16974.3 | 16767.9 | 0.6 |
| **DOC (Mln$/year)** | 36.46 | 36.02 | |
| **GWP (kg/flight)** | 13396.86 | 13311.95 | 1.2 |
| **GWP (tons/year)** | 28776.5 | 28594.1 | |

**Table 4.10:** Objective functions comparison for RM configuration.

more than 800 k$ per year for WM and more than 440k$ for RM in terms of DOC. Furthermore, it is possible to consider that the GWP reduction means a decrease of more than 850 tons for WM and more than 180 tons per year for RM in terms of emitted $CO_2$ mass.

## Synthesis of the chapter

- During the DC-3, AGILE Paradigm and AGILE technologies developed during the DC-1 and DC-2 have been applied by the candidate to two innovative aircraft configuration with turboprop propulsion system.
- The two turboprop configurations proposed by the task Architect were then manually formalized by the author in the CPACS schema in order to be usable by each partner of the consortium by means of their own CPACSized tools.
- In the third Design Campaign, the author developed six new tools dealing with aircraft mission and performance calculation, tailplanes sizing, main wing parameters morphing, aircraft stability behaviour check, CAD maker and rubber engine and aircraft mass update.
- A well-assessed combination of the tools developed by the author during the DC-1 and DC-3 and of those available in the *Tools Catalogue* coming from other partners allowed the author to implement, as a task Integrator, a concrete and reliable MDAO workflow for each turboprop configuration.
- The entire setup, from the TLAR to the executable workflow, of the MDAO processes, one for WM and one for RM configuration, was accomplished by using KE-chain platform taking advantage of all the technologies provided by the consortium.
- The execution of the single MDA, of the DOE and MDO task was conducted and accomplished by the author which was also deeply involved in the investigation of the results and the exploration of the design space.

# Chapter 5

# AGILE project: achievements, drawbacks and open challenges

## 5.1 Achievements and drawbacks

In the Multidisciplinary Design Analysis and Optimization scenario, AGILE project enters to propose a solution to the MDO community questions and requirements. What really affects the successful outcome of a MDO campaign is the time needed to go from the setup phase to the optimal solution and how the state of art techniques are employed [94].

As already mentioned in Sec. 2.4.1, three are the main phases that compose a Design and Optimization process as shown in Fig. 5.1.

The **setup phase** is the one in which many activities are included such as the organization of the requirements released for the specific design task, the definition of the MDO architecture, the formulation of the design task and the

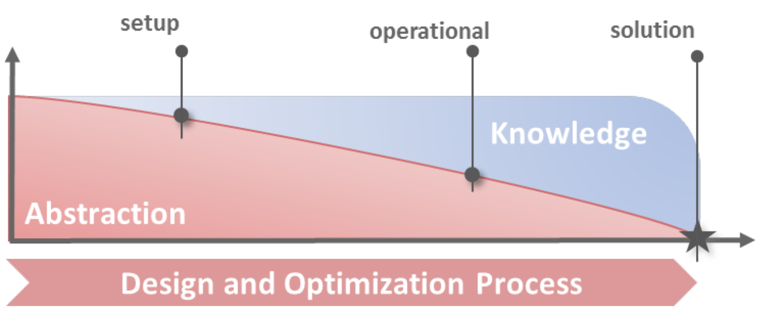



**Figure 5.1:** Design and optimization process phases.

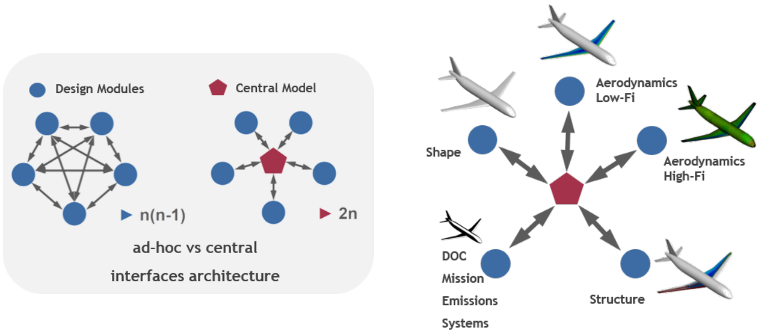



**Figure 5.2:** CPACS schema as central data model.

preparation of the framework and its infrastructure to allow data exchanging and communication among partners and organizations involved in the design and optimization activity.

The **operational** and the **solution phases** are mainly related to the design space exploration, dealing with the assessment of the results, proposing a re-configuration of the process if needed, providing extra analyses or new design or optimization techniques and selecting an optimal and robust solution.

As it can be noticed, the pre-processing activities in the **setup phase** can cover the 60% to 80% of the total process time [3] that might allow little margin for enhancing the knowledge concerning the development of new optimization techniques, the analysis of the achieved results and the improvement of the entire process. So, the way to carry on an MDAO process often results to be very complex, time-consuming and careless about setting aside time in powering up the knowledge about several optimization themes. This is one of the main issues to which the European Community has proposed a solution by funding the AGILE project.

AGILE faced the lack of agility in the setup and reconfiguration of collaborative MDO system by introducing several solution to reduce the time needed to perform an MDO process of a large-scale system also characterized by a complex cross-organizational nature in terms of data exchanging, partners coordination and MDO problem formulation.

A first crucial impact to the successful AGILE's outcomes has been given by fulfilling the necessity of **easing communication** between different disciplines through the implementation and release of CPACS technology as a central data model as illustrated in Fig. 5.2. Adopting this **standard format** and hierarchical schema to describe an aircraft and to store all the related information led to avoid spreading redundant information and to **speed up** and to make **robust** the communication among the partners. To improve the robustness of the process, a preliminary phase labeled "tools synchronization" needs to be completed, and then a certain competence or tool can be chosen by the *tools catalogue* and used if needed.

Moving to the next step, the collaborative MDO process where a fundamental milestone such as the problem formulation needs to be fixed, has been also enhanced by means of new technologies development and releasing of different PIDO. In particular, a CMDOWS schema has been developed to enable the creation of **versatile** and **re-configurable** MDO problem formulations. CMDOWS has been created to be machine-interpretable and human-readable allowing user inspections at any level, ii)neutral because it is not specific to any product or project guaranteeing great flexibility, iii)adaptable to any updates or enrichment, iv)suitable to minimize information redundancy, v)able to support all MDO strategies, vi)support tool heterogeneity from simple mathematical expressions to advanced analysis tool or surrogate model. Furthermore, it is ready to be converted to an executable workflow suited for working in RCE or OPTIMUS environments. RCE and OPTIMUS **drastically reduced** the integrator's efforts in deploying and managing the design and optimization (sub-) processes by enabling the **automated execution** of workflows composed by heterogeneous competences. Moreover, companies, organizations and institutions involved in the project which share their data and results need to be sure of the absence of any violation of their IT security. Brics ensured a **secure** data exchanging and also allowed a **remote** connection and execution between tools worldwide distributed.

Such a complex workflow composed by several disciplines managed by multiple partners and thousands of variables needs to be carefully monitored and inspected requiring a huge amount of time and human's effort. The complicated monitoring activity has been tackled by developing and releasing KADMOS technology which provides a fundamental aid to **identify** the dependencies among the design competencies and to **reconfigure** a MDO architecture in a largely reduced time period. While, **inspecting** and **debugging** actions has been considerably **accelerated** thanks to the development of the visualization package VISTOMS.

All the abovementioned technologies have been embraced within a single framework suited for implementing, formulating and solving a MDO problem. The platform developed is KE-chain which taking advantage of all the AGILE features has been able to **widely decrease** the time required to collect, connect and manage all the information coming from all the actors involved in the design task and to **largely simplify** the management of the requirements and the supervision of the entire MDO development process.

All the features presented result in the AGILE framework which has been able to provide support to every design team in each phase of the process by enabling the investigation of multiple design strategies and the development of time effective MDO methodologies [3, 95]. Furthermore, the AGILE framework offered the agility necessary to modify and/or reconfigure the system for multiple use cases from the conventional transport aircraft to the BWB configuration.

In order to give an estimation of the amount of time saved to go from the definition of the aircraft requirements to the optimal solution comparing the first Design Campaign 1 and the Design Campaign 3 of the project, the

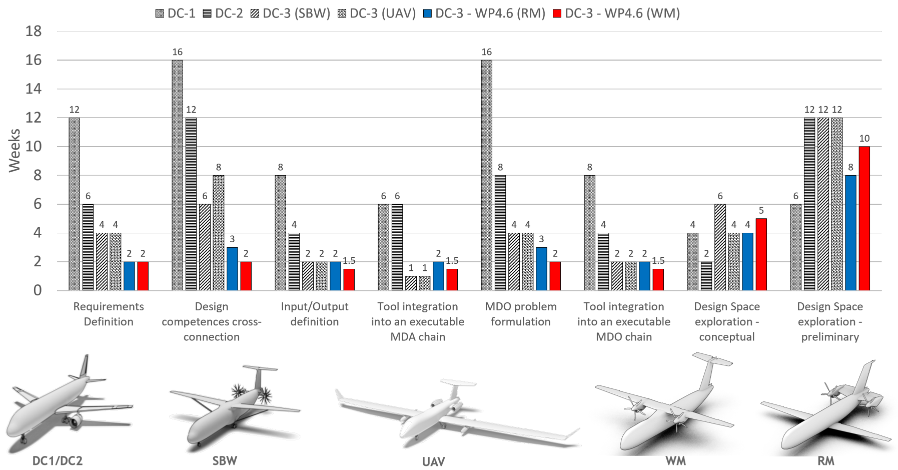



**Figure 5.3:** Comparison of the main activities of the AGILE project during the Design Campaigns.

time dimension metric accounting for time and efforts needed for each Design Campaign to setup, deploy and execute the AGILE MDO use cases [3] was monitored.

In Fig. 5.3 a detailed time history is proposed. In particular, the most time-consuming activities during DC-1 were the *Requirements Definition*, *Design competences cross-connection* and the *MDO problem formulation*. This latter due to the huge amount of data, information and variables to manage requiring a continuous inspection and re-configuration of the MDO strategy. This situation led to dedicate just few weeks to the exploration of the design space and results investigation. Then, in DC-2, taking advantage of the initial development of new technologies, the first phases of the process needed a shorter time and lower efforts. In the third Design Campaign, thanks to a robust release of KE-chain and of all the techniques in it contained, a considerably acceleration of the process from the *Requirements Definition* to the *Tool integration into an executable MDO chain* phase was obtained. This achievement and the lessons learned from the previous DC allowed to dedicate a significant amount of time to the *Design Space exploration* phase enabling the heterogeneous teams to investigate the results, enhance the methodologies, collect feedback from the specialists and converge to a better optimal solution for each use case of the DC-3. To highlight the results obtained, a focus on the third Design Campaign needs to made.Here the author was also involved as specialist and integrator (WP 4 - Task 6) in the multidisciplinary design analysis and optimization of two different use cases characterized by good level of commonality. It is important to underline that the time spent to carry out the Task 6 and the related activities were carefully planned to guarantee an equal distribution of effort both for

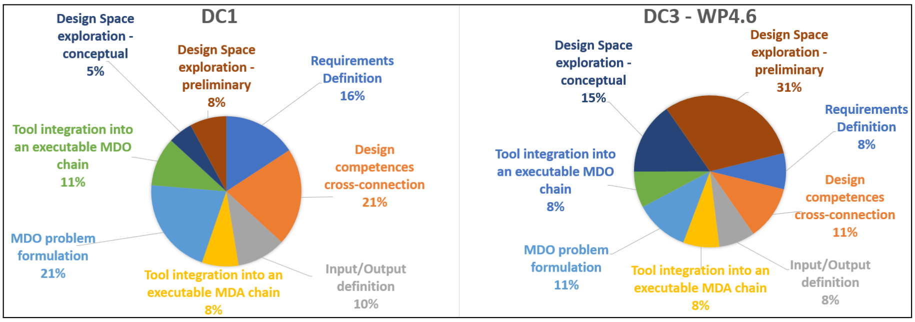



**Figure 5.4:** Comparison of the time spent for the main activities between DC-1 and DC-3 WP4.6 RM use case.

the WM and for the RM aircraft configuration. Obviously, common activities accomplished for the RM architecture were easily re-used for the WM use case and vice-versa. Indeed, as shown in Fig. 5.3, some of the initial activities of the WM use case were less time consuming, while more time was addressed to the *Design Space exploration* steps with respect to the setup preliminary phases.

An interesting comparison can be made between the DC-1 and DC-3 RM use case in terms of time spent for the same activities but using new and most effective technologies. As shown in Fig. 5.4, starting from the *Requirements Definition* activity, and moving in a clockwise direction, until the *Tool integration into an executable MDO chain* phase, the most of the steps are almost halved in time and efforts thanks to the application of the AGILE paradigm and features, while the time addressed to the analysis and investigation in the design exploration steps was tripled.

Such results and achievements need to be shared to all the MDO community in order to move a step forward in building up the next generation of MDO. With this aim, AGILE dissemination activities was really impressive. Every year of the project several paper were published by different journal and in conference proceedings, aviation events such as *ILA Air Show* held in Berlin or the *MDOpen Day* held in Hamburg, the ICAS Conference in 2018 where AGILE received the *"ICAS Award for Innovation in Aeronautics"*, were the occasion for the AGILE team to show the project challenges and achieved goals.

One of the most appreciated dissemination activity was the *AGILE Academy* initiative [96]. It was carried out aiming at injecting into the Academic institutions and educational environments the "AGILE Paradigm", providing tutoring activities and all the technologies developed over the project. The *AGILE Academy* consisted of two phases:

1. **AGILE Incubator**, where a common aircraft design task was assigned to a heterogeneous team of student worldwide distributed
2. **AGILE Challenge**, where a competition was proposed between 3 differ-

ent teams of students, also coming from different countries and organization not involved in AGILE, working on different design task(s)

Despite all the achievements obtained, EU community appreciation and remarkable results, such a complex project made of several disciplines, tens of people and really challenging tasks cannot be exempt from some drawbacks.

Concerning IT sector, several were the issues related to the neutral Teamserver used by the partners to download/upload data and results remotely. Unfortunately, Teamserver credential often failed leading to workflows execution crashes and slowing down technical and dissemination activities. Furthermore, KE-chain, that was crucial to reach the project goals, was affected by some issues which sometimes annoyed the users. For example, if an error occurred accomplishing a certain step, the ".log" file created by KE-chain did not provide any information on what was the origin of the error putting the user in trouble and making the external support request the only possible solution. In other situations, making a little change in a step meant to restart all the step process from the beginning. Trying to solve all IT and framework issues, to make the whole process works, led often to be more focused on conceptual activities rather than technical activities. Few time was spent for activities like the investigation related to the definition of peculiar criteria to evaluate the reliability of the MDO models, the integration of methodologies to support the decision-making aspect, the definition of models to count for verification and validation of the products in terms of maintenance and/or certification tasks and the analysis of requirements related to aircraft models manufacturing. In Sec. 5.2 possible solutions to the described drawbacks are presented.

## 5.2 Open challenges and future project

The drawbacks described in Sec. 5.1 have been considered as still open challenges waiting for a solution. Starting from the successful AGILE project and its outcomes, the DLR proposed to create another consortium composed by several AGILE partners but enriched by the presence of more aeronautical companies. The new assembled consortium, coordinated by the DLR, have been received funding by the EU Community to carry on a new project named *AGILE 4.0: Towards cyber-physical collaborative aircraft development.*

AGILE 4.0 is a 3 years EU project part of the H2020 program participated by 19 partners coming from research centers, academia and companies located in Europe, Russia and Canada. It will extend the AGILE's scope by introducing aspects such as maintenance, manufacturing and certification and providing an aircraft product optimization model which will cover the entire life cycle also improving the aeronautical supply chain. A guidance schema is proposed in Fig. 5.5.

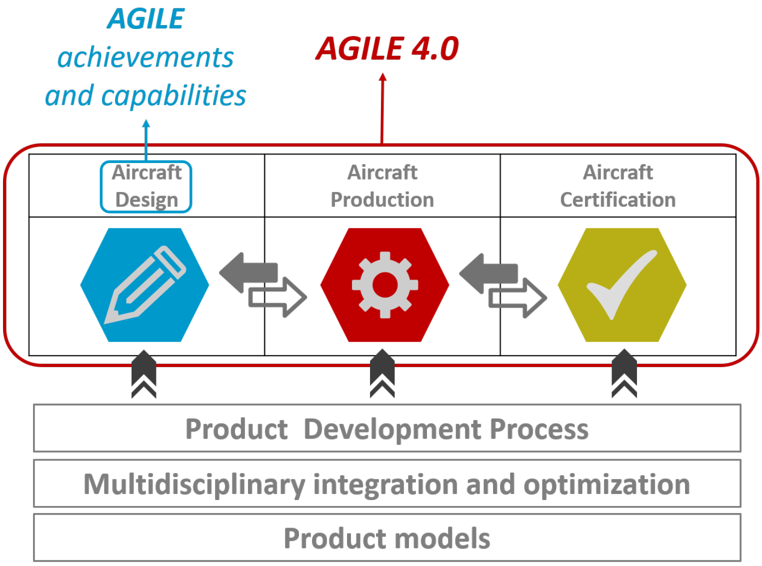



**Figure 5.5:** AGILE 4.0 project schema. Extension of AGILE capabilities and introduction of new technologies and aspects.

In particular, AGILE 4.0 aims to reach four main targets:

1. The development of innovative and constantly evolving aircraft products in a **time and cost efficient** manner by implementing a **new design and optimization paradigm** which will enable virtual products, testing and manufacturing

2. To **integrate computational design environment** and to allow the **collaboration between actors and the stakeholders in the supply chain** by developing a suited framework, methodologies and tools also related to support the decision-making aspect

3. The development of solutions such as **user-centric concurrent visualization techniques** and **data analysis and multi-objective optimization methodologies** to support the **trade-off** and the **decision-making activities**

4. To implement a digital collaborative design and optimization environment characterized by a very high flexibility aiming at zero time needed to reconfigure a pre-assembled MDO workflow

Matching the abovementioned targets should led to **reduce by 50%** the time spent in **iterations** between **design and manufacturing phase** and **by 30%** the overall process development **lead time** also thanks to the introduction of virtual integration of design, manufacturing and certification models.

Several elements will be developed and tested during the project, while some other are already under development such as the Knowledge Based Engineering (KBE) systems and Model Based Systems Engineering methodologies (MBSE). For example, a new and improved version of KE-chain is under development and it is already no more affected by the issues described in Sec. 5.1.

Reaching the AGILE 4.0 challenging objectives will mean feed the MDO community with conceptual and practical results both improving the state of the art approaches employed in the discipline of multidisciplinary design and optimization applied to large scale problems and giving a cost and time effective solutions suited for the aeronautical supply chain of the next future.

## Synthesis of the chapter

- A detailed description of the benefit encountered by the author in implementing a real MDAO framework and workflow thanks usage of AGILE technologies has been provided.
- Application of the AGILE paradigm, KE-chain, VISTOMS, KADMOS, CMDOWS, Brics and tools and methodologies developed by the author allowed a time reduction of about 60% comparing setup activities of DC-1 related to conventional aircraft configuration with respect to those performed during the DC-3 concerning disruptive and novel aircraft architectures.
- Lower efforts were required to the author in the setup phase, meanwhile the time addressed to results investigation and design space exploration was tripled.
- AGILE dissemination activities were really impressive spreading the new generation of multidisciplinary optimization to the entire MDO community by releasing an Open MDO test suite, results of the applications and the approach employed by the consortium. A note of credit also goes to the AGILE Academy initiative where the author was involved providing a remarkable contribution.
- Open challenges and future works have been presented by introducing the follow-on project "AGILE 4.0" and its main goals.

# Chapter 6

# Conclusions

The current research work presents the activities related to the AGILE project in which the author, as part of the DAF research group of the University of Naples Federico II, was involved. The main topics concerned the development of tools and methodologies for aircraft design and optimization and the handling of a complex MDO process in order to foster the creation of the next MDO generation through the implementation of the AGILE Paradigm thank to a tight collaboration with worldwide distributed partners.

The development of tools carried out by the author and other partners made possible the creation of a *Tools Catalogue* from which each partner had the possibility of choosing the missing discipline to perform his own task related to aircraft design and optimization. Usually, different organizations located in different countries are not too much in favour in sharing their knowledge or tools to collaborate to accomplish a common task. Instead, AGILE Paradigm and approach made it comfortable because the technologies developed during the project allow a remote connection among the partners ensuring a safe IT connection. In this way a collaborator can take advantage of a certain tool available in the catalogue just querying for it as a black box and running it on the machine of its owner.

The strong collaboration, efforts, new and available technologies have shown that it is possible to face and carry out the entire setup and run of a complex MDAO framework of a large-scale system like an aircraft involving 19 partners, tens of disciplines and thousands of variables in a time period of about 1 year.

In this way, a considerable amount of time can be spent for investigation activities to improve already existing methodologies or develop and test new ones. Indeed, the author had the possibility to develop a new optimization method which couples Nash game theory and Genetic Algorithm, tested on a wing optimization task, which allows a more realistic association among variables and objective functions reducing the computational time by the 13%-32% if compared to simple Genetic Algorithm and Pareto front.

A robust framework with a consolidated setup, like AGILE one, allows to rapidly reconfigure a custom MDAO workflow for novel and disruptive aircraft configuration starting from the initial setup workflow implemented for conventional aircraft architecture. Being each tool as a black box, it can be changed with another improved or reliable tool in every moment just making some minor updated to the link between the tools involved in a reconfiguration. This approach leads to avoid any waste of time allowing to quickly reach the solutions taking advantage of an already well-assessed setup phase.

This is the reason why the application of this Paradigm to innovative configuration give the possibility to address a remarkable amount of time in the exploration of the results and solutions achieved enabling the improvement of the methodologies applied. Indeed, in the phase of the project covered by the author as integrator and specialist to perform analysis and optimization tasks of innovative turboprop aircraft configurations half of time was spent in workflow reconfiguration activities and tools collection, while was triple the amount of time spent for design space exploration if compared to the first phase of the project.

The so-called AGILE Paradigm is not only an abstract formalization of a methodology, but an applicable framework that can be used by any other organizations for future research project that can follow the trail of a successful project awarded in 2018 in Belo Horizonte at the ICAS conference with the *ICAS Award for Innovation in Aeronautics*.

The main outcomes of AGILE can be summarized in:

1. The creation of the "AGILE novel aircraft configurations database". The database contains a huge amount of results and the digital models of the 6 innovative aircraft configuration analyzed in the DC-3 and can be used as a solid starting point for future research activities related to aircraft systems.

2. The development of the "AGILE Open MDO suite". The MDO suite will be made accessible to a consistent number of organizations to implement their own MDO processes of large-scale problems by taking advantage of AGILE technologies.

# Acknowledgments

The research presented in this work has been performed in the framework of the AGILE project (Aircraft 3rd Generation MDO for Innovative Collaboration of Heterogeneous Teams of Experts) and has received funding from the European Union Horizon 2020 Program (H2020-MG-2014-2015) under grant agreement $n^\circ$ 636202. The authors are grateful to the partners of the AGILE Consortium for their contribution and feedback.

# Bibliography


[1] Egbert Torenbeek *Synthesis Of Subsonic Airplane Design: An Introduction to the Preliminary Design of Subsonic General Aviation and Transport Aircraft, with Emphasis on Layout, Aerodynamic Design, Propulsion and Performance*, Springer, Delft; (1976)

[2] AGILE Project url: *http://www.agile-project.eu/*

[3] CIAMPA, P. D. and NAGEL, B., *AGILE the Next Generation of Collaborative MDO: Achievements and Open Challenges.* In MDO-07. Special session – The AGILE Project at AIAA AVIATION, 25-29 June 2018, Atlanta, Georgia, AIAA-2018-3249.

[4] CIAMPA, P. D. and NAGEL, B., *AGILE Paradigm: developing the next generation collaborative MDO.* In 18th AIAA\ISSMO Multidisciplinary Analysis and Optimization Conference, Denver, USA, June 2017.

[5] Daniel P. Raymer
*Aircraft Design: A Conceptual Approach*, AIAA education series, J.S. Przemieniecki, Series Editor-in-Chief, Air Force Institute of Technology, Wright-Patterson Air Force Base, Ohio, Published by American Institute of Aeronautics and Astronautics, Inc.

[6] John P. Fielding, College of Aeronautics, Cranfield University *Introduction to Aircraft Design*, Part of Cambridge Aerospace Series, Cranfield University, Cambridge University Press, May 2017, ISBN: 9781107680791

[7] Ilan Kroo, Steve Altus, Robert Braun, Peter Gage, and Ian Sobieski ,Aircraft Aerodynamics and Design Group, Department of Aeronautics and Astronautics ,Stanford University, Stanford, California
*Multidisciplinary Optimization Methods For Aircraft Preliminary Design*, 5th Symposium on Multidisciplinary Analysis and Optimization, 07 September 1994 - 09 September 1994, Panama City Beach,FL, U.S.A., https://doi.org/10.2514/6.1994-4325

[8] Website $http://c2.com/cgi/wiki?MonolithicDesign$

[9] Mark Klein, Peyman Faratin *Massachusetts Institute of Technology*
Hiroki Sayama, Yaneer Bar-Yam *New England Complex Systems Institute*
*The Dynamics of Collaborative Design: Insights From Complex Systems and Negotiation Research*, Acoustics, Speech, and Signal Processing Newsletter, IEEE 11(3):201-209, October 2002 DOI: 10.1177/106329303038029

[10] Erwin Moerland, Richard-Gregor Becker, Björn Nagel
*Collaborative understanding of disciplinary correlations using a low-fidelity physics-based aerospace toolkit*, September 2013, DOI: 10.1007/s13272-015-0153-4

[11] Natalia M. Alexandrov,M. Yousuff Hussaini *Multidisciplinary Design Optimization: State of the Art*, Philadelphia : Society for Industrial and Applied Mathematics, ©1997, Edited by Natalia M. Alexandrov and M. Y. Hussaini

[12] Lingaraj, H.,
*A Study on Genetic Algorithm and its Applications*, in International Journal of Computer Sciences and Engineering, Vol. 4., pp. 139-143, 2016.

[13] Eberhart, R. and Shi, Y.,
*Particle swarm optimization: Development, applications and resources*, in Proceedings of the IEEE Conference on Evolutionary Computation, ICEC. 1. 81 - 86 vol. 1. 10.1109/CEC.2001.934374.

[14] Nzanywayingoma, F. and Yang, Y.,
*Analysis of Particle Swarm Optimization and Genetic Algorithm based on Task Scheduling in Cloud Computing Environment*, in International Journal of Advanced Computer Science and Applications,Vol. 8, No. 1, 2017, DOI : 10.14569/IJACSA.2017.080104

[15] J.Sobieszczanski-Sobieski, NASA Langley Researh Center, Hampton, VA 23681-0001, USA
R.T. Haftka, University of Florida, Gainesville, FL 32611-6250, USA
*Multidisciplinary aerospace design optimization: survey of recent developments*, 34th Aerospace Sciences Meeting and Exhibit, January 15-18, 1995, Reno, Nevada

[16] Jayaram, S., Myklebust, A., Gelhausen
*ACSYNT - A standards-based system for parametric, computer aided conceptual design of aircraft* Conference paper, AIAA, Aerospace Design Conference, Irvine, CA, Feb. 3-6, 1992. 14 p., Document ID: 19920050721

[17] Martins J., R., R., A. and Lambe A., B.,
*Multidisciplinary Design Optimization: A Survey of Architectures*, AIAAJ, Vol. 51, Number 9, September 2013, https://doi.org/10.2514/1.J051895

[18] J.R.R.A. Martins, Gaetan K. W. Kenway and Timothy Brooks.
*Multidisciplinary Design Optimization of Aircraft Configurations-Part 2:*

*High-fidelity aerostructural optimization.* Lecture series,Von Karman Institute for Fluid Dynamics, Sint-Genesius-Rode, Belguim, May 2016.

[19] D. Pate, J. S. Gray, and B. German,
*A Graph Theoretic Approach to Problem Formulation for Multidisciplinary Design Analysis and Optimization*, Structural and Multidisciplinary Optimization, vol. 51, iss. 5, pp. 743-760, 2013.

[20] G. La Rocca and M. J. L. Van Tooren
*Knowledge-Based Engineering Techniques to Support Aircraft Design and Optimization* Journal of Aircraft, Vol. 46, No. 6, November-December 2009 https://doi.org/10.2514/1.39028

[21] A. Morris, P. Arendsen, G. LaRocca , M. Laban, R. Voss, H. Hönlinger
*MOB - A European Project on Multidisciplinary Design Optimisation* ICAS Conference paper, Yokohama,Japan, 29 August - 3 September 2004. https://doi.org/10.2514/6.2002-5444

[22] Görtz, S., Ilic, C., Abu-Zurayk, M., Liepelt, R., Jepsen, J., Führer, T., Becker, R., Scherer, J., Kier, T., Siggel, M.,
*Collaborative Multi-Level MDO Process Development and Application to Long-Range Transport Aircraft*, in ICAS conference proceedings, September 2016, Deajeon, South-Korea

[23] T. Lefebvre, N. Bartoli, S. Dubreuil, M. Panzeri, R. Lombardi, P. D. Vecchia, F. Nicolosi, P. D. Ciampa, K. Anisimov, A. Savelyev, *Methodological enhancements in MDO process investigated in the agile european project*, in: 18th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, Denver, USA, 2017, https://doi.org/10.2514/6.2017-4140

[24] Ciampa, P., D., Nagel, B.,
*AGILE Aircraft Development Process.* Paper 876, 6th CEAS Air & Space Conference Aerospace Europe, Bucharest, Romania, 2017

[25] Daniel Böhnke, Björn Nagel, Volker Gollnick ,Institute of Air Transportation Systems German Aerospace Center (DLR e.V.) Blohmstrasse 18, 21079 Hamburg, Germany
*An Approach to Multi-fidelity in Conceptual Aircraft Design in Distributed Design Environments* 2011 Aerospace Conference, Big Sky, MT, USA, DOI: 10.1109/AERO.2011.5747542

[26] Website: *https://cpacs.de* (Retrieved on January 8th 2020)

[27] Website: *https://github.com/DLR-SL/CPACS* (Retrieved on January 8th 2020)

[28] P. S. Prakasha, P. D. Ciampa, B. Nagel, L. Boggero and M. Fioriti,
*Collaborative Systems Driven Aircraft Configuration Design Optimization* 30th ICAS Conference, 25-30 Sep 2016, Daejeon, South Korea

[29] Website: *https://rcenvironment.de/pages/documentation/documentation.html* (Retrieved on July 10th 2019)

[30] Website: *https://www.noesissolutions.com/our-products/optimus* (Retrieved on January 8th 2020)

[31] Gray, J., S., Hwang, J., T., Martins, J., R., R., A., Moore, K., T., Naylor, B., A.,
*OpenMDAO: an open-source framework for multidisciplinary design, analysis, and optimization*, Structural and Multidisciplinary Optimization Journal, April 2019, Vol. 59, pp. 1075-1104, ISSN: 1615-1488, DOI: 10.1007/s00158-019-02211-z

[32] Website *http://tixi.sourceforge.net/Doc/index.html*

[33] Website *https://dlr-sc.github.io/tigl/doc/3.0.0-rc2/*

[34] Website *www.daf.unina.it*

[35] M. Zhang, A.W. Rizzi, F. Nicolosi, A. De Marco ,
*Collaborative aircraft design methodology using ADAS linked to CEASIOM* 32nd AIAA Applied Aerodynamics Conference, Session: Aerodynamic Analysis and Design Methodologies, Atlanta, Georgia, USA, 2014, DOI: 10.2514/6.2014-2012

[36] Ruocco M. Cusati V. Nicolosi F. Trifari, V. and A. De Marco,
*Multidisciplinary analysis and optimization java tool for aircraft design*, 31st congress of the International Council of the Aeronautical Sciences,2018, EID: 2-s2.0-85060465239.

[37] Trifari V.
*Development of Multi-Disciplinary Analysis and Optimization framework and applications for innovative efficient regional aircraft.* Phd, University of Naples Federico II, Department of Industrial Engineering, University of Naples Federico II, Naples, Italy, 2020.

[38] Ruocco M.
*High lift and stability issues for innovative transport aircraft configurations in aerodynamic design.* PhD. Department of Industrial Engineering, University of Naples Federico II, Naples, Italy: University of Naples Federico II, 2020.

[39] P. Della Vecchia, L. Stingo, F. Nicolosi, G. Cerino, P.D. Ciampa, P.S. Prakasha, M. Fioriti, M. Zhang, A. Mirzoyan, B. Aigner, D. Charbonnier,
*Advanced turboprop multidisciplinary design and optimization within AGILE project* 2018 Aviation Technology, Integration, and Operations Conference, AIAA 2018-3205,Session: Design Processes and Education, Atlanta, USA, https://doi.org/10.2514/6.2018-3205

[40] L. Stingo, G. Nardone,
*Development of an Innovative Collaborative Framework for Aircraft Design* 6th SCAD & READ Workshop, Warsaw, Poland, 2016.

[41] D.Ciliberti, P.Della Vecchia, F.Nicolosi, A. De Marco ,
*Aircraft directional stability and vertical tail design: A review of semi-empirical methods* Vol. 95, November 2017, Pages 140-172 ISSN: 0376-0421, https://doi.org/10.1016/j.paerosci.2017.11.001

[42] Nicolosi F., Della Vecchia P., Ciliberti D., Cusati V.
*Fuselage Aerodynamic Prediction Methods* Aerospace Science and Technology (Elsevier) AESCTE, Vol. 55, Aug. 2016, pp. 332-343, ISSN 1270-9638, http://dx.doi.org/10.1016/j.ast.2016.06.012

[43] Fink, R.
*USAF Stability and Control DATCOM* AFWAL-TR-83-3048. McDonnell Douglas Corporation, Douglas Aircraft Division, for the Flight Controls Division, Air Force Flight Dynamics Laboratory, Wright-Patterson AFB, Ohio. October 1960, revised November 1965, revised April 1978.

[44] van Gent, I., La Rocca, G. and Veldhuis, L., L., *Composing MDAO symphonies: graph-based generation and manipulation of large multidisciplinary systems* 18th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference 5-9 June 2017 Denver, Colorado DOI: https://doi.org/10.2514/6.2017-3663

[45] 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 (2018) 9: 607. https://doi.org/10.1007/s13272-018-0307-2

[46] Aigner, B., van Gent, I., La Rocca, G., Stumpf, E. and Veldhuis, L. L. M. *Graph-based algorithms and data-driven documents for formulation and visualization of large MDO systems* CEAS Aeronautical Journal (2018) 9(4), 695-709. https://doi.org/10.1007/s13272-018-0312-5

[47] Ciampa, P., D., Moerland, E., Seider, D., Baalbergen, E., Lombardi, R., D'Ippolito, R., *A Collaborative Architecture supporting AGILE Design of Complex Aeronautics Products* AIAA AVIATION Forum, 5-9 June 2017, Denver, Colorado, 18th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference

[48] van Gent, I., Ciampa, P. D., Aigner, B., La Rocca, G. and Schut, J. *Knowledge architecture supporting collaborative MDO in the AGILE paradigm* In 18th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference: 5-9 June 2017, Denver, Colorado [AIAA 2017-4139] AIAA. https://doi.org/10.2514/6.2017-4139

[49] Daniel Böehnke, Bjoern Nagel, Volker Gollnick
*An Approach to Multi-Fidelity in Distributed Design Environments* IEEE Aerospace Conference, Big Sky, USA, 2011

[50] La Rocca, G., Langen, T.,H.,M., Brouwers, Y.,H.,A.,
*The design and engineering engine. Towards a modular system for collaborative aircraft design*, 28th Congress of the International Council of the Aeronautical Sciences, ICAS 2012.

[51] M. Fioriti, L. Boggero, S. Corpino, A. Isyanov, A. Mirzoyan, R. Lombardi e R. D'Ippolito,
*Automated Selection of the Optimal On-board Systems Architecture*, in 18th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, Denver (CO), 2017.

[52] Nicolosi, F., Della Vecchia, P., Ciliberti, D.
*Aerodynamic interference issues in aircraft directional control* ASCE's Journal of Aerospace Engineering, Vol. 28, N. 1, January 2015, ISSN 0893-1321, DOI: 10.1061/(ASCE)AS.1943-5525.0000379, 04014048

[53] Roskam, J.,
*Airplane Design, Part VI: Preliminary Calculation of Aerodynamic, Thrust and Power Characteristics*, Roskam Aviation and Engineering Corporation, Published by DAR Corporation 1990, Ottawa, Kansas,

[54] J. Roskam,
*Airplane Aerodynamics and Performance*, University of Kansas, Published by DAR Corporation, 1985, ISBN-10: 1884885446, ISBN-13: 978-1884885440

[55] J. A. Jr Blackwell
*A Finite-Step Method for Calculation of Theoretical Load Distributions for Arbitrary Lifting-Surface Arrangements at Subsonic Speeds* Washington, D.C., 1969, Document ID: 19690021959

[56] P.M. Sforza,
*Commercial Airplane Design Principles*, Edited by Butterworth-Heinemann, 2014, ISBN: 9780124199538

[57] *FAA Federal Aviation Regulations, FARs, Part 23, Section 149- Minimum control speed*

[58] Trifari, V., Ruocco, M., Cusati, V., Nicolosi, F., De Marco, A.,
*Java framework for parametric aircraft design - Ground performance*, (2017) Aircraft Engineering and Aerospace Technology, 89 (4), pp. 599-608, DOI: 10.1108/AEAT-11-2016-0209

[59] N. Bartoli, T. Lefebvre, S. Dubreuil, R. Olivanti, N. Bons, J. R. R. A. Martins, M.-A. Bouhlel, J. Morlier
*An adaptive optimization strategybased on mixture of experts for wing aerodynamic design optimization*, 18th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, Denver, USA, 2017, DOI: 10.2514/6.2017-4433

[60] J. R. R. A. Martins, A. B. Lambe
*Multidisciplinary design optimization: A survey of architectures* , AIAA Journal 59 (9) (2013) 2049-2075, https://doi.org/10.2514/1.J051895

[61] P. Della Vecchia, L. Stingo, S. Corcione, D. Ciliberti, F. Nicolosi, A. De Marco,
*Game theory and evolutionary algorithms applied to mdo in the agile european project*, in: 18th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, Denver, USA, 2017, DOI: 10.2514/6.2017-4330

[62] F. Daoud, R. Maierl, P. D. Ciampa, X. Gu,
*Aeroelastic shape and sizing optimization of aircraft products supported by AGILE design paradigm*, in: 18th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, Denver, USA, 2017, https://doi.org/10.2514/6.2017-4141

[63] W. Lammen, B. de Wit, J. Vankan, H. Timmermans, T. van der Laan, P. D. Ciampa,
*Collaborative design of aircraft systems - multi-level optimization of an aircraft rudder*, in: 6th CEAS Aerospace Europe Conference, 2017.

[64] P. S. Prakasha, A. Mirzoyan, P. D. Ciampa,
*Collaborative system of systems multidisciplinary design optimization for civil aircraft:agile eu project*, in: 18th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, Denver, USA, 2017, https://doi.org/10.2514/6.2017-4142

[65] K. Deb,
*Multi-Objective Optimization Using Evolutionary Algorithms*, John Wiley & Sons, Inc., New York, NY, USA, 2001. ISBN: 047187339X

[66] F. Nicolosi, A. De Marco, L. Attanasio, P. Della Vecchia,
*Development of a java-based framework for aircraft preliminary design and optimization*, Journal of Aerospace Information Systems 13 (6) (2016) 234–242, https://doi.org/10.2514/1.I010404

[67] F. Nicolosi, P. Della Vecchia, D. Ciliberti,
*An investigation on vertical tailplane contribution to aircraft sideforce*, Journal of Aerospace Science and Technol-ogy 28 (2013) 401–416, DOI: 10.1016/j.ast.2012.12.006

[68] J. Periaux, H. Chen, B. Mantel, M. Sefrioui, H. Sui,
*Combining gametheory and genetic algorithms with application to DDM-nozzle optimization problems*, Finite Elements in Analysis and Design 37 (5) (2001) 417–429, https://doi.org/10.1016/S0168-874X(00)00055-X

[69] P. Della Vecchia, E. Daniele, E. D'Amato,
*An airfoil shape optimizationtechnique coupling parsec parameterization and evolutionary algorithm*, Aerospace Science and Technology 32 (1) (2014) 103–110, https://doi.org/10.1016/J.AST.2013.11.006

[70] Basar, T., Olsder, G.J.,
*Dynamic non cooperative game theory*, In: Classics in Applied Mathematics, 23. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA (1995).

[71] Haupt, R.L., Haupt, S.E.,
*Practical Genetic Algorithms*, Wiley-Interscience, Hoboken, New Jersey, 2004, ISBN-10: 0471455652 ISBN-13: 978-0471455653

[72] D'Amato, E., Daniele, E., Mallozzi, L., Petrone, G., Tancredi, S.,
*A Hierarchical Multi-modal Hybrid Stackelberg-Nash GA for a Leader with Multiple Followers Game*, In: Dynamics of Information Systems: Mathematical Foundations, Springer Proceedings in Mathematics & Statistics, (ed. Sorokin, A.; Murphey, R.; Thai, M.T.; Pardalos, P.M., Vol. 20) 267-280, Springer Verlag USA (2012), DOI: 10.1007/978-1-4614-3906-6_14.

[73] D'Amato, E., Daniele, E., Mallozzi, L., Petrone, G.,
*Equilibrium strategies via GA to Stackelberg games under multiple follower's best reply*, International Journal of Intelligent Systems, **27**, Wiley Subscription Services, Inc., A Wiley Company, (2012), https://doi.org/10.1002/int.21514.

[74] Fudenberg, D., Tirole, J.,
*Game Theory*, The MIT Press, Boston, USA, 1991, ISBN-10: 0262061414 ISBN-13: 978-0262061414.

[75] Nash, J.F.
*Non-Cooperative Games*, In: The Annals of Mathematics, Second Series,Princeton University, NJ, USA, **54**, 286-295 (1951), DOI: 10.2307/1969529

[76] Deb, K., Agrawal, S., Pratap, A., Meyarivan, T.,
*A Fast Elitist Non-Dominated Sorting Genetic Algorithm for Multi-Objective Optimization: NSGA-II*, In: Parallel Problem Solving from Nature – PPSN VI, edited by Marc Schoenauer and K. Deb and G. Rudolph and Xin Yao and Evelyne Lutton and Juan Julian Merelo and Hans-Paul Schwefel, Springer, Berlin, 849–858 (2000).

[77] Wang, J.F., Periaux, J., Sefrioui, M.,
*Parallel evolutionary algorithms for optimization problems in aerospace engineering*, In: Journal of Computational and Applied Mathematics, no.149, pagg. 155-169 (2002), https://doi.org/10.1016/S0377-0427(02)00527-7.

[78] *https://it.mathworks.com/help/gads/how-the-genetic-algorithm-works.html*, Matlab online documentation

[79] *https://it.mathworks.com/help/gads/examples/performing-a-multiobjective-optimization-using-the-genetic-algorithm.html*, Matlab online documentation

[80] L. Stingo, P. Della Vecchia, G. Cerino, F. Nicolosi, A. De Marco,
*MDO Applications to conventional and novel turboprop aircraft within AGILE European project*, 31st Congress of the International Council of the Aeronautical Sciences, ICAS 2018.

[81] Melin, T.
*Using Internet Interactions in Developing Vortex Lattice Software for Conceptual Design*, Ph.D. Thesis, Department of Aeronautics, Royal Institute of Technology, KTH, Stockholm, Sweden, 2003

[82] Drela, M.
*A User's Guide to MSES 3.05*, MIT Department of Aeronautics and Astronautics, July 2007.

[83] Palacios, F., Colonno, M.R., Aranake, A.C., Campos, A., Copeland, S.R., Economon, T.D., Lonkar, A.K., Lukaczyk, T.W., Taylor, T.W.R., Alonso, J.
*(SU2): An open-source integrated computational environment for multi-Physics simulation and design*, In Proceedings of the 51st AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition, Grapevine, TX, USA, 7–10 January 2013; AIAA 2013-0287.

[84] Della Vecchia, P., Stingo, L., Nicolosi, F., De Marco, A., Daniele E. and D'Amato, E,
*Application of Game Theory and Evolutionary Algorithm to the Regional Turboprop Aircraft Wing Optimization*, In Evolutionary and Deterministic Methods for Design Optimization and Control With Applications to Industrial and Societal Problems, Springer International Publishing AG, part of Springer Nature, 2019, Volume 49 Series editor Eugenio Oñate, Universitat Politècnica de Catalunya Barcelona, Spain, DOI: https://doi.org/10.1007/978-3-319-89890-2_26

[85] Della Vecchia, P., Nicolosi, F., Ruocco, M., Stingo, L., De Marco, A.,
*An improved high-lift aerodynamic prediction method for transport aircraft*, CEAS Aeronautical Journal, ( ), 1-10 , 2018, DOI: 10.1007/s13272-018-0349-5

[86] P. D. Ciampa, T. Zill and B. Nagel,
*Aeroelastic Design and Optimization of Unconventional Aircraft Configurations in a Distributed Design Environment*, in 53rd AIAA SDM, Honolulu, USA, 2012, https://doi.org/10.2514/6.2012-1925.

[87] P. D. Ciampa, T. Zill and B. Nagel,
*A Hierarchical Aeroelastic Engine for the Preliminary Design and Optimization of the Flexible Aircraft*, in 54th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference, Boston, USA, 2013, https://doi.org/10.2514/6.2013-1820.

[88] P. S. Prakasha, P. D. Ciampa, L. Boggero e M. Fioriti,
*Assessment of airframe-subsystems synergy on overall aircraft performance in a Collaborative Design*, in 17th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, Washington D.C. (USA), 2016, https://doi.org/10.2514/6.2016-3667.

[89] Lammering, T., Franz, K., Risse, K., Hoernschemeyer, R., and Stumpf, E.,
*Aircraft Cost Model for Preliminary Design Synthesis*, 50th AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition, American Institute of Aeronautics and Astronautics, Reston, Virigina, 2012, DOI: 10.2514/6.2012-686.

[90] Dallara, E. S., Kroo, I. M., and Waitz, I. A.,
*Metric for Comparing Lifetime average Climate Impact of Aircraft*, AIAA Journal, Vol. 49, No. 8, 2011, pp. 1600–1613, DOI: 10.2514/1.J050763

[91] Ruijgrok G.J.J. and Van Paassen D.M.,
*Elements of Aircraft Pollution*, (2005) Delft: Delft University Press, ISBN print 978-90-407-2602-6

[92] Kollat, J. B., and Reed, P. M.,
*Comparison of Multi-Objective Evolutionary Algorithms for Long-Term Monitoring Design*, Advances in Water Resources, 29(6):792-807, 2006.

[93] Sierra, M. R. and Coello Coello, C. A.,
*Improving PSO-based multi-objective optimization using crowding, mutation and $\epsilon$-dominance*, Evolutionary Multi-Criterion Optimization, Berlin, Germany, 505-519, 2005.

[94] P. Ciampa and B. Nagel,
*Towards the 3rd generation MDO collaboration Environment*, in 30th Congress of the International Council of the Aeronautical Sciences, Daejeon, 2016.

[95] Ciampa P.D., Prakasha, P., S., Torrigiani, F., Walther, J., Lefebvre, T., Bartoli, N., Timmermans, H., Della Vecchia, P., Stingo, L., Rajpal, D., van Gent, I., La Rocca, G., Fioriti, M., Cerino, G., Maierl, R., Charbonnier, D.,

Jungo, A., Aigner, B., Anisimov, K., Mirzoyan, A. and Voskuijl, M., *Streamlining Cross-Organizational Aircraft Development: Results from the AGILE Project*, AIAA Aviation 2019 Forum, 17-21 June 2019, Dallas, Texas, DOI: https://doi.org/10.2514/6.2019-3454

[96] Della Vecchia, P., Ciampa, P., Prakasha, P., S., Aigner, B., van Gent, I., *MDO Framework for university research collaboration : AGILE Academy Initiatives & Outcomes*, Multidisciplinary Analysis and Optimization Conference, June 25-29, 2018 Atlanta, Georgia https://doi.org/10.2514/6.2018-3254