MDO Architectures Comparison on Analytical Test Case and Aerostructural Aircraft System Design Problem

Francesco Torrigiani
DLR, German Aerospace Center, Systems Architectures in Aeronautics
Research Engineer
Hein-Saß-Weg 22, 21129 Hamburg
Francesco.Torrigiani@dlr.de

Pier Davide Ciampa
DLR, German Aerospace Center, Systems Architectures in Aeronautics
Group Leader
Pier.Ciampa@dlr.de

ABSTRACT

An aircraft system design problem is intrinsically a multidisciplinary problem. If the design configuration is unconventional, sound low-fidelity analysis methods are not available. Complex hi-fidelity tools are often the only solution to obtain reliable results, and for these reasons designers are deeply interested in the interactions and organization of these tools. Inside a Multidisciplinary Design Optimization (MDO) process, different architectures are possible. Analysis and comparison of six MDO architectures is the aim of this paper. The considered architectures are All-At-Once (AAO), Simultaneous Analysis and Design (SAND), Individual Discipline Feasible (IDF), Multidisciplinary Feasible (MDF), Collaborative Optimization (CO), Bi-Level Integrated System Synthesis (BLISS). The comparison is conducted on mathematical benchmark cases and on a simplified aerostructural aircraft design problem. Results expressed in a unified nomenclature are available as open source. Further, the CMDOWS (Common MDO Workflow Schema) developed in the AGILE project is used to translate the neutral description of the MDO problem into an executable implementation and it will be released as open source too. The aim is to promote the discussion on MDO architectures within the MDO research community.

KEYWORDS: *Multidisciplinary Design Optimization (MDO), MDF, IDF, CO, CMDOWS, AGILE, BLISS, SAND, AAO*

NOMENCLATURE

f	= objective function
$\boldsymbol{c}$	= vector of constraint functions
$\boldsymbol{R}$	= vector of disciplinary equations in residual form
$\boldsymbol{c}^c$	= vector of consistency constraint functions
N	= number of disciplines
$\boldsymbol{x}_0$	= vector of shared design variables
$\boldsymbol{x}_i$	= vector of local (discipline i) design variables
$\boldsymbol{y}_i$	= vector of coupling state variables provided by discipline i
$\overline{\boldsymbol{y}}_i$	= vector of local (discipline i) state variables
$\widehat{\boldsymbol{y}}_i$	= vector of target state variables (copy of discipline i coupling state variables)
$\widehat{\boldsymbol{x}}_i$	= system copy of local design variables vector
$\widehat{\boldsymbol{x}}_{0\,i}$	= discipline i copy of shared design variables vector
$\boldsymbol{q}$	= vector of wing parameters (Table 12)
AR	= aspect ratio
S	= wing area
τ	= thickness to chord ratio global factor
λ_{out}	= outboard taper ratio
θ_K	= kink twist
θ_T	= tip twist
F	= mission fuel mass
OEM	= operating empty mass
s	= subscript for structural discipline
m	= subscript for mission discipline

1 INTRODUCTION

Integration is important in every engineering field and for complex system a multidisciplinary approach must be adopted from the early stage of the project development [1-2]. The design of unconventional aircraft configuration is one of the best examples of this kind. The lack of appropriate database pushes the designer to use physics based simulation models, directly inside the design loop; thus, the request for the most efficient way to use these tools.
Multidisciplinary design optimization applies optimization theory to the design of engineering systems. Since the inception of MDO, several architectures have been formalized to solve complex MDO problems for aeronautical systems [3]. These are classified into monolithic (single-level) and distributed (multilevel) architectures.
All-At-Once (AAO), Simultaneous Analysis and Design (SAND), Individual Discipline Feasible (IDF; [4-5]), Multidisciplinary Feasible (MDF; [4]) are monolithic architectures. There is not a complete agreement in literature regarding the labeling of AAO and SAND architectures. Here, the review article of Martins and Lambe [6] is used as reference for the nomenclature. Thus, AAO is the "most general formulation" from which all the others can be casted.
Among distributed architectures there are Collaborative Optimization (CO; [7-8]) and Bi-Level Integrated System Synthesis (BLISS; [9]). The ongoing AGILE project is developing the next generation of MDO processes, and investigating multiple MDO techniques.
This paper addresses multiple MDO problems, whose architectures are described with a unified nomenclature and represented as XDSM [6] diagrams which show both the data connections and process flow. Furthermore, CMDOWS (Common MDO Workflow Schema) [10] process format developed in AGILE is used to store the MDO problems in a neutral format and to transfer the formulations into executable workflow implemented in RCE (Remote Component Environment) [11]. For all the architecture presented and solved in this study the corresponding CMDOWS files and the corresponding XDSM diagrams, are made available as open-source.
In Section 2 a brief introduction to MDO architectures is provided. Section 3 presents the multiple architectures implemented and compared on an analytical benchmark cases: the Sellar problem [12]. In Section 4 an overall aircraft design (OAD) problem is considered. Aerodynamic and structural analyses together with other disciplines are included in the process. Results of both use cases are provided in terms of graphs, as optimization path in the design variables spaces, and evolution of the objective function during iterations. Detailed quantitative data are collected in tables. Both graphs and tables will be part of the aforementioned open-source material. Finally, in Section 5 overall conclusions are proposed together with some outlooks of the research.

1.1 Review of MDO Architectures Benchmark

There are several review works on MDO architectures. One of the first is provided by Haftka et al. [13], and later monolithic architectures have been formalized by Cramer et al. [4]. More recently the work of Martins and Lambe [6] provides an exhaustive collection of MDO architectures both monolithic and distributed. In this survey the use of a unified nomenclature helps the reader to understand differences and similarities between architectures.
One of the issues MDO community has to deal with is the benchmarking of architectures. Some of the first papers on MDO architectures benchmarking have employed analytical test cases [14-16]. Successively comparison of MDO formulations has been conducted on various design cases: the conceptual design of a supersonic business jet in [16]; the design of a reusable launch vehicle in [17]; the seizing of an air flow sensor and of turbine blades in [18]. However, all the models considered in the aforementioned works make us of analytical functions as disciplinary representations. It can be observed there are 3 main challenges concerning the benchmarking of MDO architectures.
The first one is that MDO results strongly depend on the implementation of the architecture: the same architecture applied by two different research groups to the same optimization problem may obtain different optimization results. The same formulation can be coded in several slightly different ways, even if the programming languages used is the same. This problem can be partially solved using a specialized MDO platform like OpenMDAO, iSIGHT or pyMDO.

A second challenge regards the choice of compared architectures. Monolithic methodologies are the most investigated. Distributed ones are seldom considered and among them often CO and CSSO are selected.
Finally it is very common to find comparisons addressing only low dimensional test cases or analytical discipline functions. This makes unreliable the extension of obtained results to realistic design cases, where high fidelity simulations tools are employed.
This work addresses the 3 challenges in the following way.

1. Three different platforms are used in this paper: Matlab, OpenMDAO and RCE. In this way differences due to the implementation are underlined and the most suitable platform for the aircraft design problem is identified.
2. Both monolithic and distributed architectures are studied. Among distributed CO and BLISS are considered.
3. Architectures are implemented on an analytical test case and also on an overall aircraft design (OAD) problem, in which physics based simulation models are employed. Monolithic and distributed architectures are both compared on the aircraft design use case.

2 MDO ARCHITECTURES BACKGROUND

Consider a generic optimization problem formulated as:

$$\min_{\boldsymbol{x}} f(\boldsymbol{x}, \boldsymbol{y}) \tag{1}$$

subject to:

$$\begin{aligned} &\boldsymbol{c}(\boldsymbol{x}, \boldsymbol{y}) \leq \boldsymbol{0} \\ &\boldsymbol{R}_i(\boldsymbol{x}, \boldsymbol{y}) = \boldsymbol{0} \quad \text{for } i = 0, \dots, N \end{aligned} \tag{2}$$

The objective function f is minimized with respect to the design variables $\boldsymbol{x}$. The other arguments of the objective function are the state variables $\boldsymbol{y}$. Then, constraints $\boldsymbol{c}$ and disciplinary (i is the number of disciplines) equations $\boldsymbol{R}_i$ depend by both design and state variables. Note that there is not lack of generality. Equality constraints can be always rewritten as pairs of inequalities, and disciplinary equation can be reordered in residual form.
It is important to distinguish among the analysis and the design processes. Throughout the analysis only the disciplinary equations are considered $\boldsymbol{R}_i$, and there is no difference between design and state variables; these are just variables satisfying a system of equations. Instead, after defining objective and constraints functions, the designer selects some of the disciplinary equations variables and indicates them as design variables. The remaining variables are the state variables. A simple example shows why there is not loss of generality in this definition.
Consider the aerodynamic analysis of a wing. This analysis can be represented by the following equation, already in residual form.

$$a(\boldsymbol{z}, \boldsymbol{p}) = 0 \ , \tag{3}$$

where $\boldsymbol{z}$ are the coordinates of the points on the discretized surface of the wing and $\boldsymbol{p}$ is the pressure on each of these points[1]. The optimization problem consists in minimize the drag coefficient, $\mathrm{C}_D(\boldsymbol{z}, \boldsymbol{p})$, imposing the condition $L(\boldsymbol{z}, \boldsymbol{p}) - \mathrm{W} \leq 0$, where L is the lift and W the aircraft weight. The designer can either choose some of the points $\boldsymbol{z}$ as design variables or define some other overall quantities $\boldsymbol{q}$, for example aspect ratio, wing span, taper ratio, etc.. In the latter case an additional relation connecting the new quantities with the wing discretization points:

$$\mathrm{g}(\boldsymbol{z}, \boldsymbol{q}) = 0, \tag{4}$$

and this is added to the set of disciplinary equations in the optimization problem. Therefore, there is no difference between design and state variables for the disciplinary equations for the analyzer point of view. The differences are connected to the design process: design variables are quantities the designer chooses to be always under the control of the optimizer.
Dealing with multidisciplinary problems means some of the variables are shared among different disciplines' equations. Design variables are denoted by $\boldsymbol{x}_0$ if shared, and by $\boldsymbol{x}_i$ if discipline local. $\boldsymbol{y}_i$

[1] For panel method this relation is a linear system $A(\boldsymbol{z})\boldsymbol{p} - b(\boldsymbol{z}) = 0$.

and $\bar{y}_i$ are respectively discipline shared, also called coupling variables, and discipline local state variables[2].

So far only the optimization problem has been defined, as architecture independent. A brief description of the various architectures is addressed in the following section in conjunction with their application to the first analytical test case. The already mentioned Martins and Lambe survey [3] is the reference for a deeper and complete description on architectures.

Here just a short comment on monolithic and distributed formulations is proposed. The main difference between the two approaches is the number of optimization problems defined and solved within the optimization process. Monolithic architectures have a single optimization problem, while in distributed architectures the original problem is decomposed into several smaller optimization problems, disciplines and system subproblems. The reason of this decomposition might depends on the typical structure of the engineering-design environments. It is a common practice, especially in industry, to split the design of a large system among different engineering groups. These groups may be geographically dislocated and may communicate rarely. Therefore, choosing a distributed formulation they can work on their own optimization problem independently, without waiting for the results of other groups as would happen in a monolithic approach.

3 AN MDO ANALYTICAL USECASE: THE SELLAR PROBLEM

The Sellar Problem [12] is a classical benchmark for multidisciplinary methods, and it is also used in AGILE as the test case developing CMDOWS [10]. The statement of the problem follows.

Table 1: Sellar problem statement

Objective	Constraints	Disciplinary eq.
$\min \; x_1^2 + x_{02} + y_1 + e^{-y_2}$ respect to x_{01}, x_{02}, x_1	$1 - y_1/3{,}16 \leq 0$ $y_2/24 - 1 \leq 0$ $-10 \leq x_{01} \leq 10$ $0 \leq x_{02} \leq 10$ $0 \leq x_1 \leq 10$	$y_1 = x_{01}^2 + x_1 + x_{02} - 0{,}2y_2$ $y_2 = \sqrt{y_1} + x_{01} + x_{02}$

There are two disciplines, two shared and one local design variables, two coupling state variables without any local state variables. In the following paragraphs description and application of six architectures to the Sellar problem are provided. For each architecture the corresponding XDSM representation is provided, as well as the implementation schema within the integration environment RCE.

3.1 SAND Architecture for Sellar Problem

The simplest approach is to let the optimizer do all the work. This is the concept behind the SAND architecture. The optimizer controls both design (x_{01}, x_{02}, x_1) and state (y_1, y_2) variables, also the local ones. Disciplinary equations are considered just as other equality constraints of the problem, and their residual values must be exposed to the optimizer.

Table 2: SAND statement for Sellar problem

Objective	Constraints	Disciplinary eq.
$\min \; x_1^2 + x_{02} + y_1 + e^{-y_2}$ respect to $x_{01}, x_{02}, x_1, y_1, y_2$	$1 - y_1/3{,}16 \leq 0$ $y_2/24 - 1 \leq 0$ $-10 \leq x_{01} \leq 10$ $0 \leq x_{02} \leq 10$ $0 \leq x_1 \leq 10$	$y_1 - (x_{01}^2 + x_1 + x_{02} - 0{,}2y_2) = 0$ $y_2 - (\sqrt{y_1} + x_{01} + x_{02}) = 0$

In this approach, that is suitable also for single-discipline problem, the optimizer simultaneously analyses and designs the system. Note that disciplinary equations are not solved explicitly or exactly and this allow the optimizer exploring unfeasible regions during first iterations.

[2] Note that no disciplinary distinction between shared design variables is necessary.

On the other hand, the optimizer has to control also local state variables, that means a huge amount data[3]. Furthermore, discipline analysis simulation tools usually work in a black-box fashion providing state variables value and hiding the residual value of discipline equations. MDF and IDF architectures address this aspect.

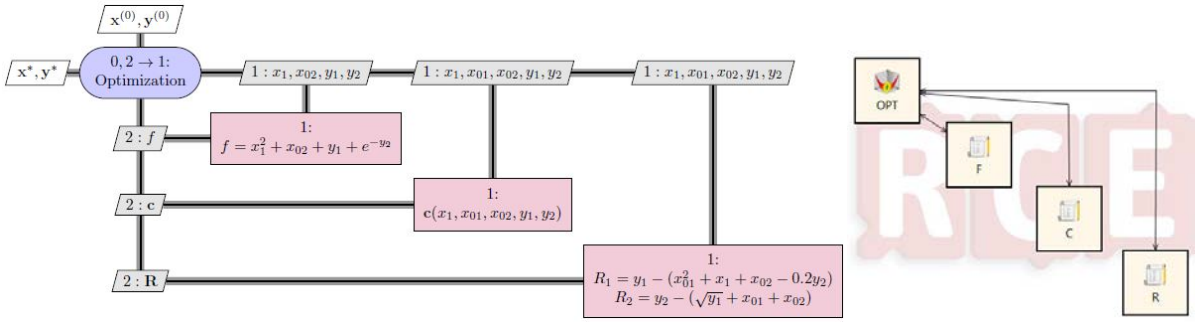

Figure 1: SAND XDSM (left) and RCE workflow (right) for Sellar problem

3.2 MDF Architecture for Sellar Problem

The structure of MDF is rather intuitive. Starting from an initial guess of one of the coupling state variables (y_2^0), single disciplines tools are organized in series to calculate the remaining coupling variables. A feedback branch guarantees the contemporary fulfillment of all the equations once convergence is reached[4]. This loop is the multidisciplinary analysis (MDA) loop. As can be seen from the XDSM graph in Figure 2, the MDA loop receives the design variables (x_{01}, x_{02}, x_1) as input and provides the coupling state variables (y_1, y_2) as output. Hence, the only variables controlled by the optimizer are the design variables, and this is the lowest possible number among all the monolithic architecture.

Table 3: MDF statement for Sellar problem

Objective	Constraints
$\min \; x_1^2 + x_{02} + y_1(x_{01}, x_{02}, x_1) + e^{-y_2(x_{01}, x_{02}, x_1)}$ respect to x_{01}, x_{02}, x_1	$1 - y_1(x_{01}, x_{02}, x_1)/3{,}16 \leq 0$ $y_2(x_{01}, x_{02}, x_1)/24 - 1 \leq 0$ $-10 \leq x_{01} \leq 10$ $0 \leq x_{02} \leq 10$ $0 \leq x_1 \leq 10$

With this architecture, disciplinary equations are all satisfied at each iteration of the optimizer. This is the so called multidisciplinary feasibility of the method which is the main difference with respect to the IDF method. However, the constraints feasibility is not guaranteed by the architecture, because it depends on the optimization algorithm.

The main disadvantage of MDF is that for each optimizer iteration the MDA loop must reach the convergence. Therefore, for each evaluation of the objective function the discipline tools are evaluated several times.

[3] Of course not here where there are not local variables. For instance using a panel method all nodes and pressure values must be controlled by the optimizer.

[4] Omitting the feedback branch is also possible, and in this case the architecture is said MDF not converged. In this case one relies much more on the initial guess.

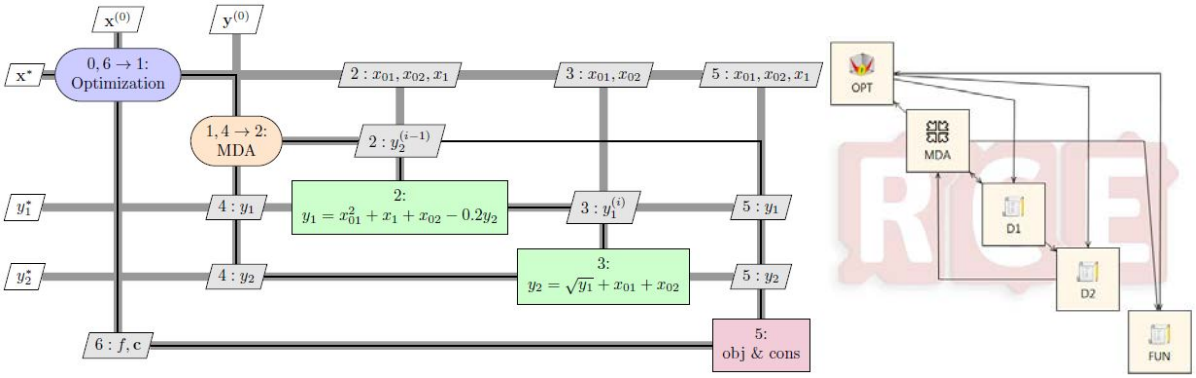

Figure 2: MDF Gauss-Seidel XDSM (left) and RCE workflow (right) for Sellar problem

Inside the MDA loop, several different organizations of discipline tools are possible. The series type sequence is the most intuitive one and is borrowed from numerical linear algebra method Gauss-Seidel. Another possible sequence, inspired from numerical linear algebra method as well, is the Jacobi method. With Jacobi convergence, the multidisciplinary analysis modules are organized in parallel and a feedback branch departs from each of them, as shown in Figure 3.

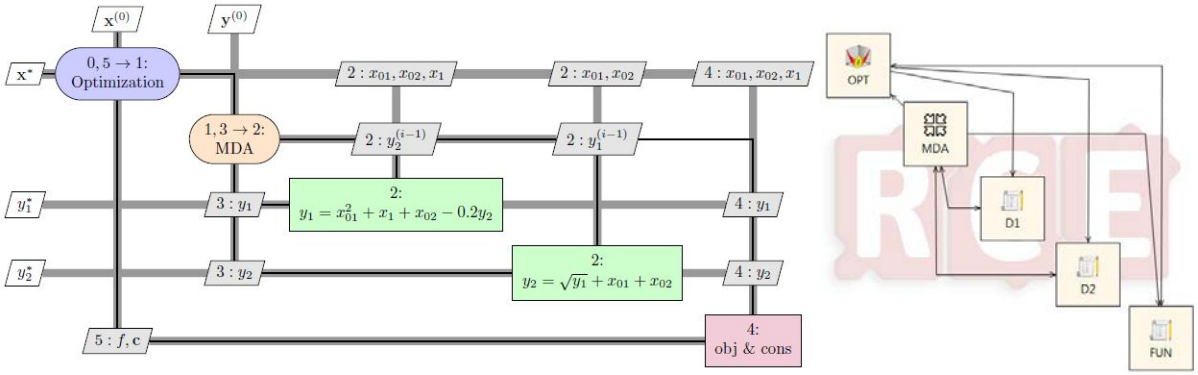

Figure 3: MDF Jacobi XDSM (left) and RCE workflow (right) for Sellar problem

3.3 IDF Architecture for Sellar Problem

In order to reduce the number of discipline evaluations needed by the MDF, the MDA loop is eliminated in the IDF architecture. Copies of coupling design variables are provided in order to use the different discipline tools independently. Each disciplinary module calculates its own coupling state variables $\boldsymbol{y}_i$ (here, for example for the first discipline, means y_1), using copies of other disciplines coupling state variables, called target variables and identified by the hat $\hat{\boldsymbol{y}}_{j\neq i}$ (for the first discipline means $\hat{y}_2$, copy of y_2). Local and coupling state variables are not exposed, while target variables are. Consistency constraints are added to ensure the equality of coupling state variables and their copies.

Table 4: IDF statement for Sellar problem

Objective	Constraints	Consistency
$\min\ x_1^2 + x_{02} + y_1(x_{01}, x_{02}, x_1, \hat{y}_2) + e^{-y_2(x_{01}, x_{02}, \hat{y}_1)}$ respect to $x_{01}, x_{02}, x_1, \hat{y}_1, \hat{y}_2$	$1 - y_1(x_{01}, x_{02}, x_1, \hat{y}_2)/3{,}16 \leq 0$ $y_2(x_{01}, x_{02}, \hat{y}_1)/24 - 1 \leq 0$ $-10 \leq x_{01} \leq 10$ $0 \leq x_{02} \leq 10$ $0 \leq x_1 \leq 10$	$\hat{y}_1 - y_1 = 0$ $\hat{y}_2 - y_2 = 0$

In this method only single discipline feasibility is guaranteed during iterations. Global disciplines feasibility is obtained only at the end of the optimization process. This architecture is a tradeoff between MDF and SAND. The number of required evaluations is reduced, respect to MDF, introducing additional variables controlled by the optimizer (target variables). However, the number of variables exposed to the optimizer is lower in IDF than in SAND (in IDF local state variable are not exposed), and that is achieved solving explicitly the discipline equations (as in MDF).

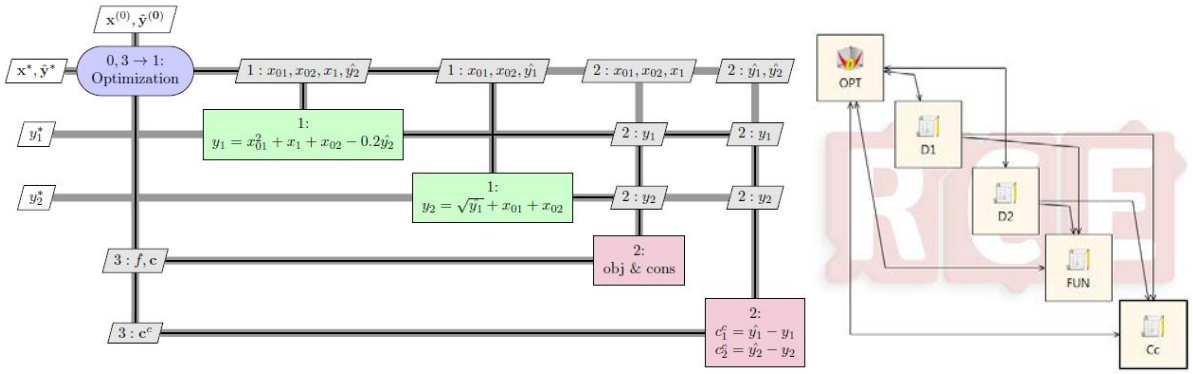

Figure 4: IDF XDSM (left) and RCE workflow (right) for Sellar problem

3.4 AAO Architecture for Sellar Problem

The AAO architecture is obtained introducing target variables and consistency constraints in SAND architectures. Like in SAND the optimizer controls all the variables, which here are: design, state and target variables. Again disciplinary equations are considered as constraints.

Table 5: AAO statement for Sellar problem

Objective	Constraints	Disciplinary eq.	Consistency
$\min\ x_1^2 + x_{02} + y_1 + e^{-y_2}$ respect to $x_{01}, x_{02}, x_1, y_1, y_2, \hat{y}_1, \hat{y}_2$	$1 - y_1/3{,}16 \leq 0$ $y_2/24 - 1 \leq 0$ $-10 \leq x_{01} \leq 10$ $0 \leq x_{02} \leq 10$ $0 \leq x_1 \leq 10$	$y_1 - (x_{01}^2 + x_1 + x_{02} - 0{,}2\hat{y}_2) = 0$ $y_2 - (\sqrt{\hat{y}_2} + x_{01} + x_{02}) = 0$	$\hat{y}_1 - y_1 = 0$ $\hat{y}_2 - y_2 = 0$

Actually AAO is never solved in practice in this form, because the introduction of target variables does not provide any advantages and they can be easily eliminated using consistency constraints (which are linear in this formulation). Doing this, SAND architecture is obtained again. The reason for defining AAO architecture is strictly formal. It is the most general formulation, in which all possible variables and constrains are presented; removing some of them all the other architecture can be obtained.

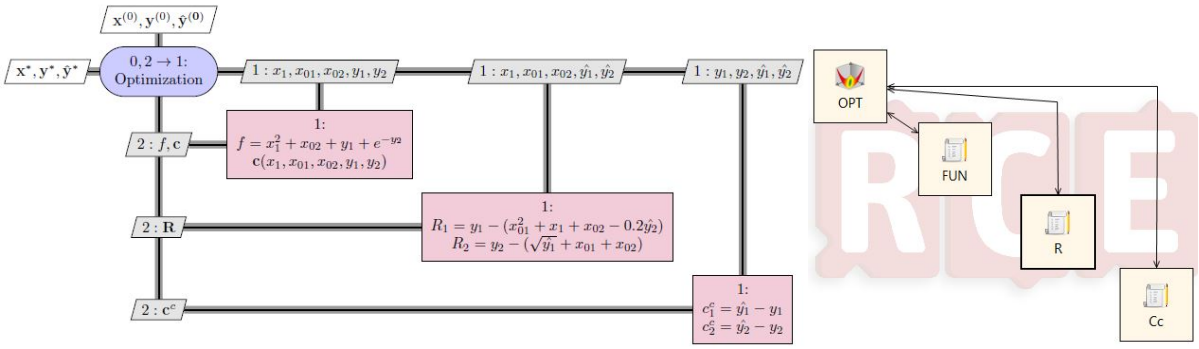

Figure 5: AAO XDSM (left) and RCE workflow (right) for Sellar problem

3.5 CO Architecture for Sellar Problem

This distributed architecture exploits and extends the concept of IDF. In this formulation single discipline tools not only are executed independently using copies of coupling state variables ($\hat{y}_1, \hat{y}_2$), but they are wrapped in their own discipline optimization subproblem. For this reason each disciplinary subproblem needs its own copies of the design variables ($\hat{x}_{01\_1}, \hat{x}_{02\_1}$ for first discipline and $\hat{x}_{01\_2}, \hat{x}_{02\_2}$ for second discipline).

Furthermore copy of local design variables ($\hat{x}_1$) are needed only if these variables are explicitly employed in objective or constraints functions of the system subproblem (as the case of Sellar problem).

The purpose of the discipline optimizations is to enforce the consistency of state and design variables with their copies. Whereas, the system level optimization subproblem collects all the discipline optimized consistency constraints and minimizes the original objective function. The CO system subproblem statement for the Sellar problem is the following.

Table 6: CO system subproblem statement for Sellar problem

System Subproblem		
Objective	Constraints	Consistency
$\min\ \hat{x}_1^2 + x_{02} + \hat{y}_1 + e^{-\hat{y}_2}$ respect to $x_{01}, x_{02}, \hat{x}_1, \hat{y}_1, \hat{y}_2$	$1 - \hat{y}_1/3{,}16 \leq 0$ $\hat{y}_2/24 - 1 \leq 0$ $-10 \leq x_{01} \leq 10$ $0 \leq x_{02} \leq 10$	$J_1^* = 0$ $J_2^* = 0$

The asterisk on the consistency constraints functions J_i indicates they are optimized by the respective discipline subproblem. For this reason the system optimization is also called post-optimal calculation. The discipline subproblems are shown in Table 7

Table 7: CO discipline subproblems statement for Sellar problem

Discipline 1 Subproblem	
Objective	Constraints
$\min J_1 = \left\|\hat{x}_{01\_1} - x_{01}\right\|^2 + \left\|\hat{x}_{02\_1} - x_{02}\right\|^2 + \left\|x_1 - \hat{x}_1\right\|^2 + \left\|\hat{y}_1 - y_1\left(\hat{x}_{01\_1}, \hat{x}_{02\_1}, x_1, \hat{y}_2\right)\right\|^2$ respect to $\hat{x}_{01\_1}, \hat{x}_{02\_1}, x_1$	$0 \leq x_1 \leq 10$
Discipline 2 Subproblem	
Objective	Constraints
$\min J_2 = \left\|\hat{x}_{01\_2} - x_{01}\right\|^2 + \left\|\hat{x}_{02\_2} - x_{02}\right\|^2 + \left\|\hat{y}_2 - y_2\left(\hat{x}_{01\_2}, \hat{x}_{02\_2}, \hat{y}_1\right)\right\|^2$ respect to $\hat{x}_{01\_2}, \hat{x}_{02\_2}$	none

As in IDF the discipline feasibility is not guaranteed during iterations of system optimization subproblem, but only at the end of it. The main disadvantage of this form of CO is the poor convergence rate. Indeed, the gradient of consistency constraints at an optimal point is a null vector. This slows down convergence for the most common gradient-base optimization algorithm. Several ways of solving this issue were proposed. One is the use of linear consistency constrains for each variable and copy of it, but then the problem would have more equality constraints than variables. Another possible solution is relaxing equality constraints. They are replaced by inequalities with a relaxation tolerance, where the tolerance is a small constant number.

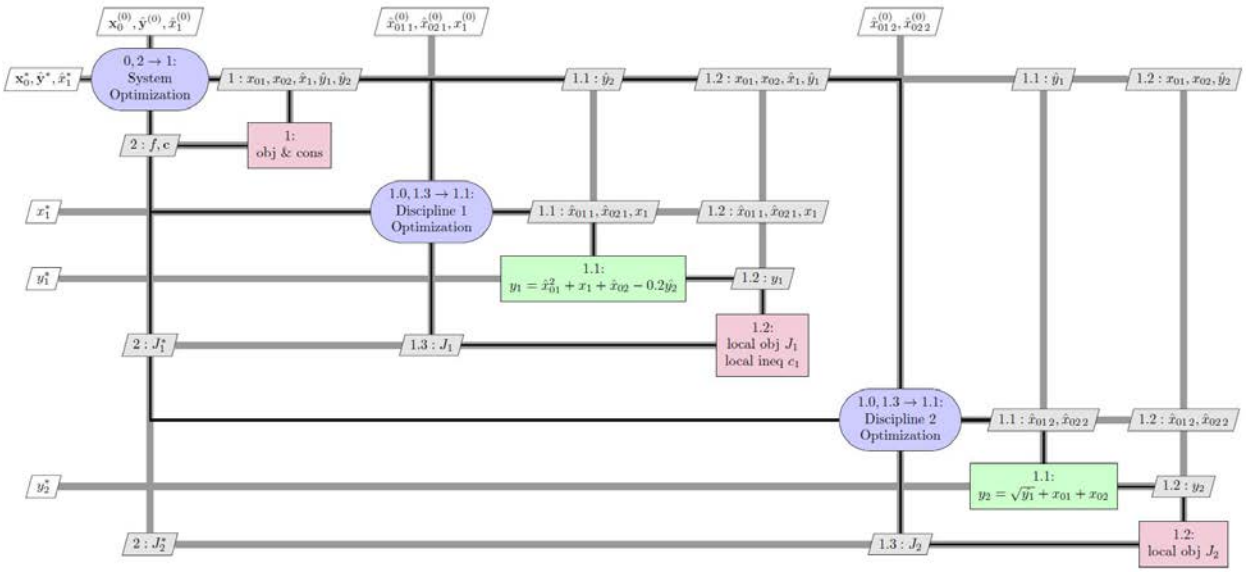

Figure 6: CO XDSM for Sellar problem

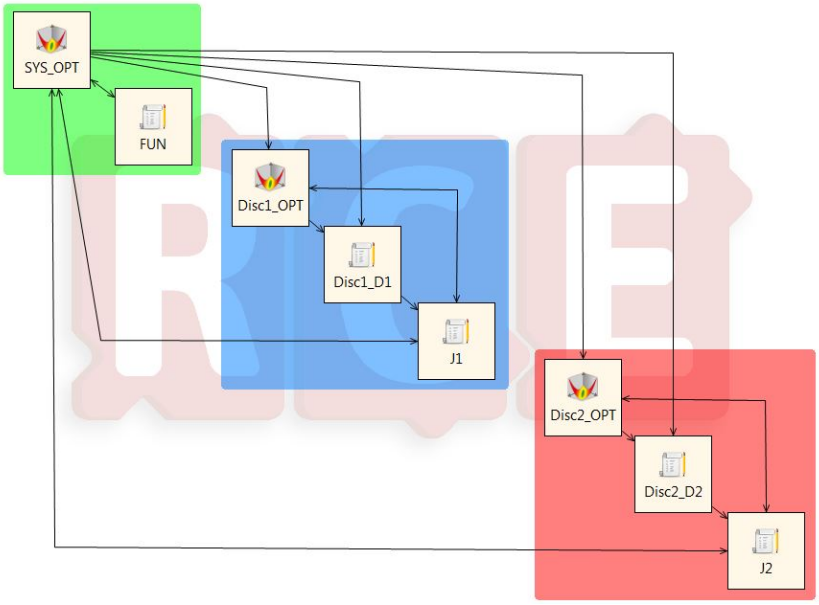

Figure 7: CO RCE workflow for Sellar problem

3.6 BLISS Architecture for Sellar Problem

The driving concept here is building a path in the design space, using a series of linear approximations of the original design problem. User defined bounds on the approximated design step are used to prevent design point moving too far respect to the approximation accuracy. This is the same idea of thrust-region optimization algorithm.

For each approximation point the local linear approximation of the objective function is minimized respect to the design variables step[5] $\Delta \boldsymbol{x}$. The optimization is accomplished in two times. First, discipline optimization subproblems define the optimal step in local discipline design variables direction. Then, the system level subproblem determines the optimal step in the remaining directions, which are the shared design variables directions. As in thrust region algorithm, when two (or more) steps give the same value of design variables and objective function, the minimum is reached. The system optimization subproblem follows.

[5] Note: approximation point for objective and constraints functions remains constant during the optimization. This is indicated by the subscript 0.

Table 8: BLISS system subproblem statement for Sellar problem

System Subproblem	
Objective	Constraints
$\min \left(f_0^*\right)_0 + \left(\frac{\partial f_0^*}{\partial x_{01}}\right)\Delta x_{01} + \left(\frac{\partial f_0^*}{\partial x_{02}}\right)\Delta x_{02}$ respect to $\Delta x_{01}, \Delta x_{02}$	$(\boldsymbol{c}_0^*)_0 + \left(\frac{\partial \boldsymbol{c}_0^*}{\partial x_{01}}\right)\Delta x_{01} + \left(\frac{\partial \boldsymbol{c}_0^*}{\partial x_{02}}\right)\Delta x_{02} \leq 0$ $\Delta x_{01\_L} \leq \Delta x_{01} \leq \Delta x_{01\_U}$ $\Delta x_{02\_L} \leq \Delta x_{02} \leq \Delta x_{02\_U}$

Note: the linear approximation point used in the system subproblem is actually moved in local discipline design variables direction, and this legitimize the term of post-optimal calculation also for this architecture[6] (thus the use of the asterisk).
Although the involved disciplines are two, the discipline subproblem for Sellar problem in BLISS architecture is only one. This is a peculiarity of BLISS: the number of discipline subproblems is equal to the number of discipline local design variables. Thus, only the first discipline subproblem is defined.

Table 9: BLISS first discipline subproblem statement for Sellar problem

Discipline 1 Subproblem	
Objective	Constraints
$\min \left(f_0\right)_0 + \left(\frac{\partial f_0}{\partial x_1}\right)\Delta x_1$ respect to Δx_1	$(\boldsymbol{c}_0)_0 + \left(\frac{\partial \boldsymbol{c}_0}{\partial x_1}\right)\Delta x_1 \leq 0$ $\Delta x_{1\_L} \leq \Delta x_1 \leq \Delta x_{1\_U}$

The subproblems statements are reported in general form without using the explicit expressions for the derivatives of f and $\boldsymbol{c}$; though it would be possible for an analytical problem like this one. This representation reflects better the actual implementation, for which a finite difference approach is used to make the method suitable also for complex problems.
Several differences can be observed by comparing BLISS with CO. First a new set of constraints has been introduced. These are the bounds of the approximation step, and are defined outside the MDO process by the user.
Other structural differences are evident looking at the graphic representation of the architectures. CO presents the disciplines subproblems organized in parallel respect each other and nested on the system subproblem, Figure 6-7. Furthermore, in CO analysis modules are inside the discipline optimization subproblem, which is responsible of enforcing consistency among original and copied variables. Whereas in BLISS analysis tools are outside any optimization subproblems, and in series with them, Figure 8-9. This means that an IDF-style method to deal with copied variables is not possible since there would be no optimizer to guarantee consistency. Therefore, an MDF structure is the only solution, and the analysis modules are wrapped in an MDA loop.
Advantages and disadvantages of BLISS come both from the thrust-region like structure. Gradients of objective and constraints functions are immediately available (and constant for the optimizer), since they are necessary for the definition of the optimization problems themselves. On the other hand all the unreliability connected to the linear approximation is unavoidable. This issue can be only partially solved by easy-to-handle user-defined bounds for design step, especially if the user has not a clear idea of the problem nonlinearity level.

[6] Even if in this architecture discipline subproblems are not nested on the system subproblem, but in series with it.

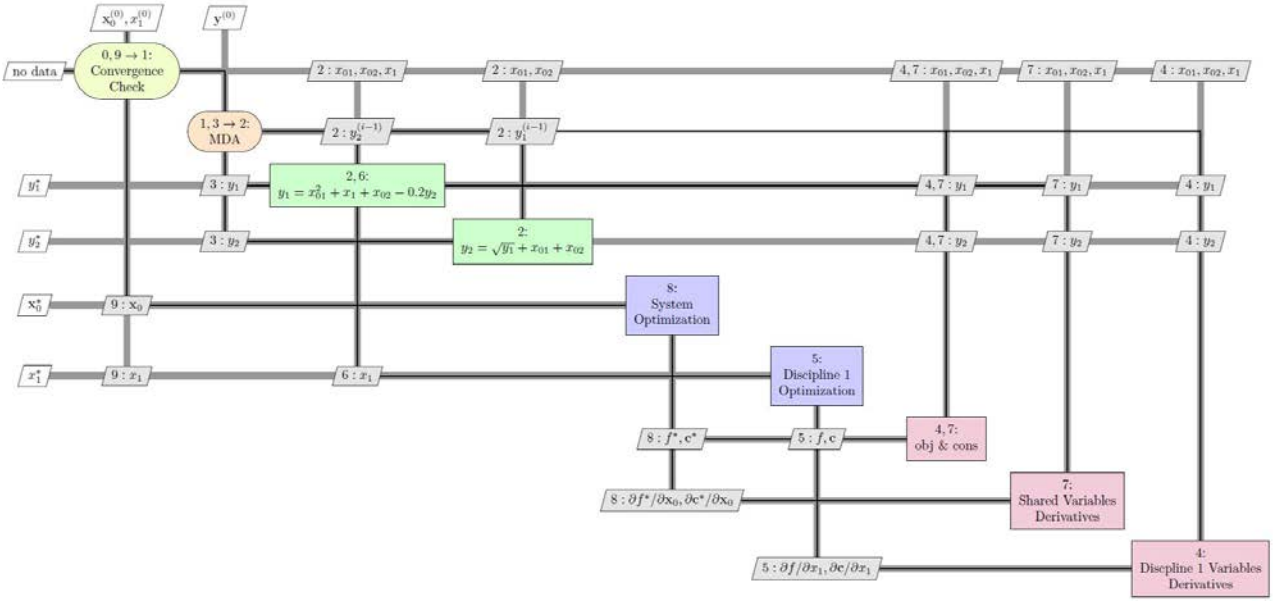

Figure 8: BLISS XDSM for Sellar problem

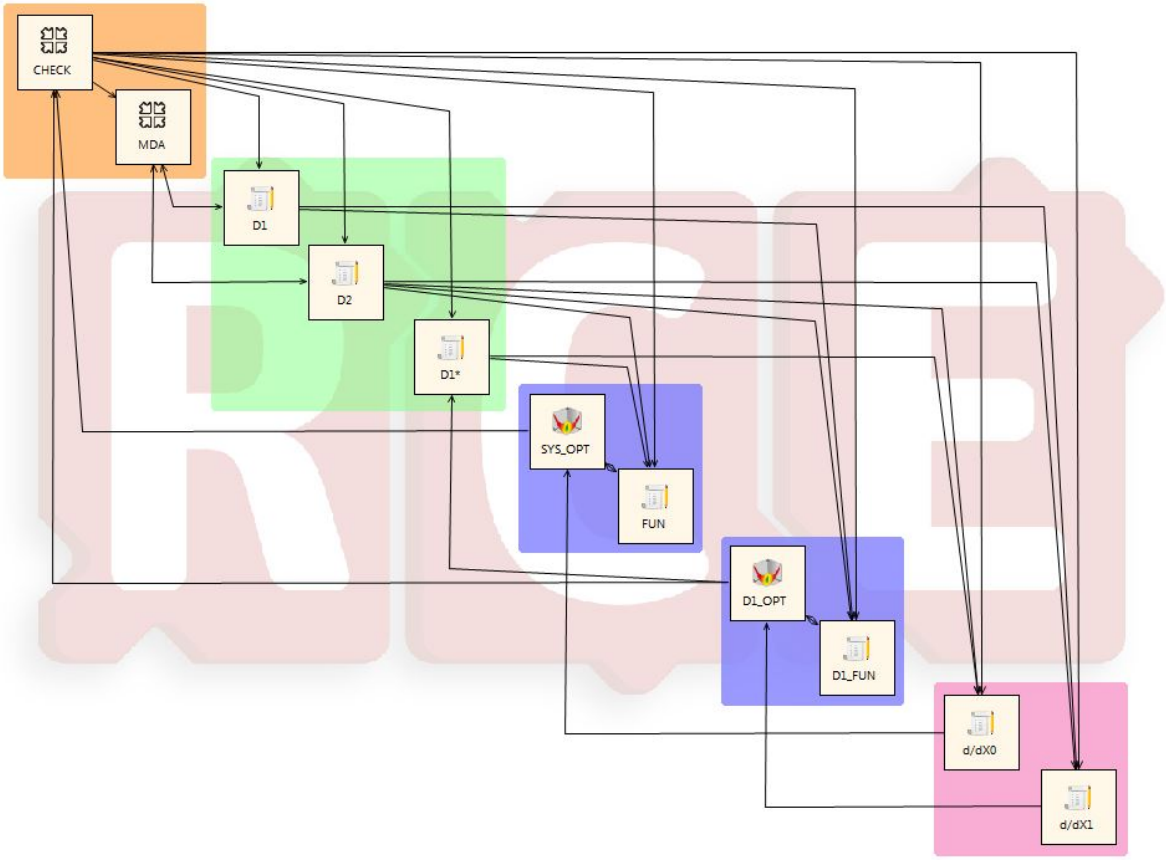

Figure 9: BLISS RCE workflow for Sellar problem

3.7 Architectures Comparison Results for Sellar Problem

All the results from the MDO architectures described in the previous subsections are here compared. The optimization history results are shown in two kinds of graphs: objective function evolution and optimization path in the design domain. The latter shows the sequence of iteration points chosen by the optimizer in the design variables space. This kind of representation is possible because the design space has only three dimensions.

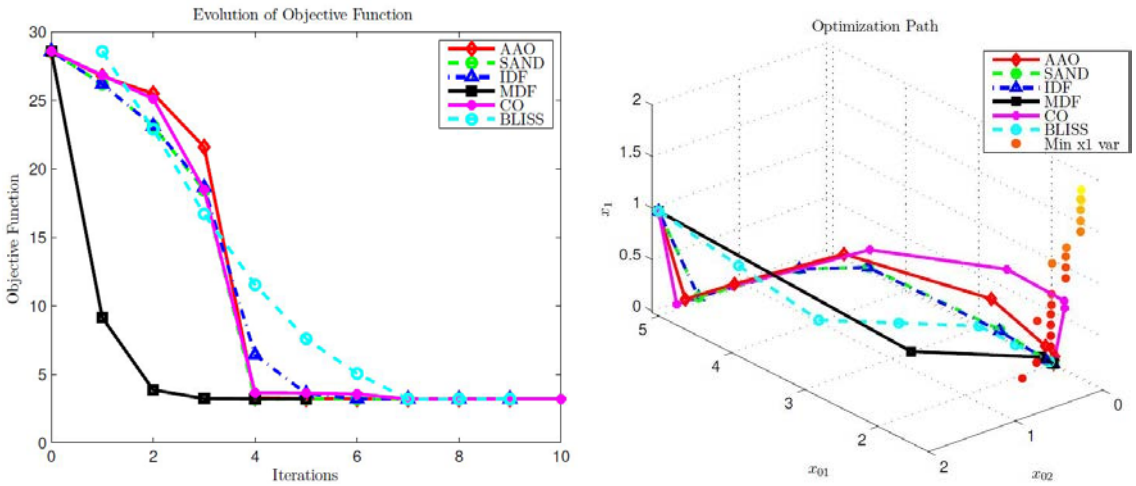

Figure 10: Objective function evolution (left) and optimization path (right) for Sellar problem

The cloud of dots on the right corner of the optimization path graph, Figure 10, is made by minimum points analytically calculated keeping x_1 fixed. As can be seen, MDF has the steepest descendant behavior; it also chooses a radically different path respect to the other architectures. In this particular case the performances of distributed architectures are basically equivalent to the monolithic ones. The number of iterations, the shape of objective function evolution and of the optimization path of CO and BLISS are similar to the monolithic architectures ones, Figure 10.
However, as it is shown by the graphs in Figure 11, changing the starting point seriously affects the performance of CO. Instead, BLISS solution is strongly dependent on the approximation step length. Figure 11 shows also a BLISS with the half of the previous step length.

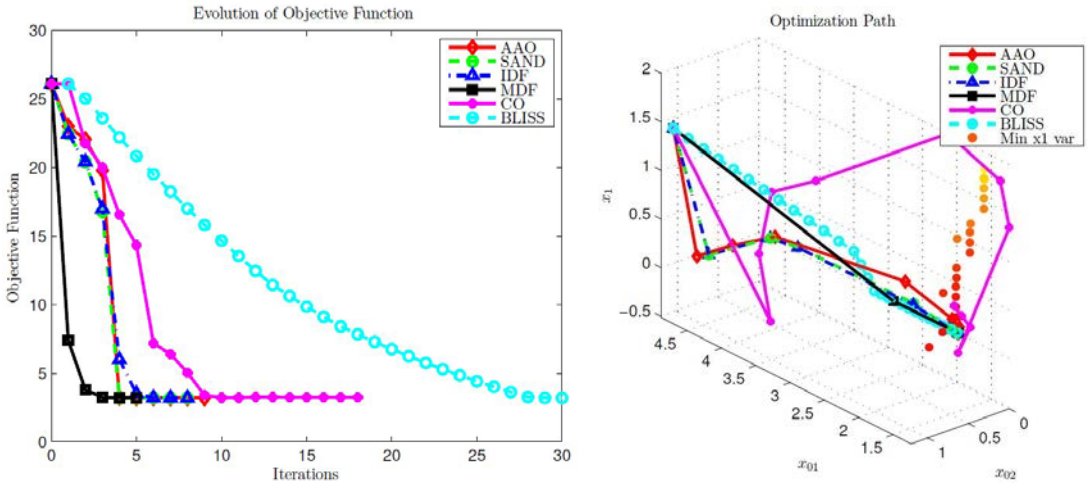

Figure 11: Objective function evolution (left) and optimization path (right) for Sellar problem with different starting point and approximation step length for BLISS

For the case represented in Figure 11 a table of quantitative data is reported in Table 10. Due to the different structure the indications on the table have slightly different meanings for each architecture. For AAO, SAND and IDF there is no ambiguity: iterations are the optimizer iterations. For MDF the optimizer iterations are reported, but the MDA iterations are implicitly considered in the number of discipline calls. For CO the iterations of the system subproblem are reported, but the discipline subproblems iterations are included in the discipline calls. For BLISS the iterations are the number of approximation points calculated before the optimum is reached. Again, iterations of MDA are included in the objective and discipline functions calls.
Finally, for all the architectures (except BLISS), gradients are never provided to any optimizer. Finite difference method is used and, thus, functions are evaluated several times within single optimizer iteration.

Table 10: Summary of architectures comparison on Sellar problem

	AAO	SAND	IDF	MDF	CO	BLISS
Optimal obj.value	3.1834	3.1834	3.1834	3.1834	3.233	3.1834
Iterations	9	8	8	5	18	52
Obj.fun calls	80	72	72	48	176	243
Disc. 1 calls	0	0	72	152	2642	348
Disc. 2 calls	0	0	72	152	797	348

The table underlines the difference between monolithic and distributed architectures. For the latter the number of discipline tool calls is definitely greater, especially for CO. However, must be pointed out that the purpose of developing distributed architectures is not to decrease calculations but to make disciplinary groups independent. In an industrial environment, it may effectively decreases the development lead time.
A clear difference exists among distributed formulations. CO performs few system iterations, each one with many disciplinary calls. BLISS disciplinary calls for single iteration are fewer but the number of global iterations is higher.
Moreover, the number of discipline calls in BLISS is equal for both disciplines, like in monolithic architectures and unlike CO. This is expected because in BLISS there is no optimizer driving the discipline analyses. All the disciplines are inside the same MDA converger loop and optimization subproblems are in series with it.
The implementations of monolithic architectures are compared also on different optimization platforms. The previous results are all obtained with Matlab optimization-environment, and are also compared with OpenMDAO and RCE environments.
Regarding the optimization algorithm adopted, for both Matlab and OpenMDAO implementation, a sequential quadratic algorithm is used while for RCE the built-in DAKOTA Coliny COBYLA algorithm (an extension of simplex algorithm for linear and nonlinear constraints).

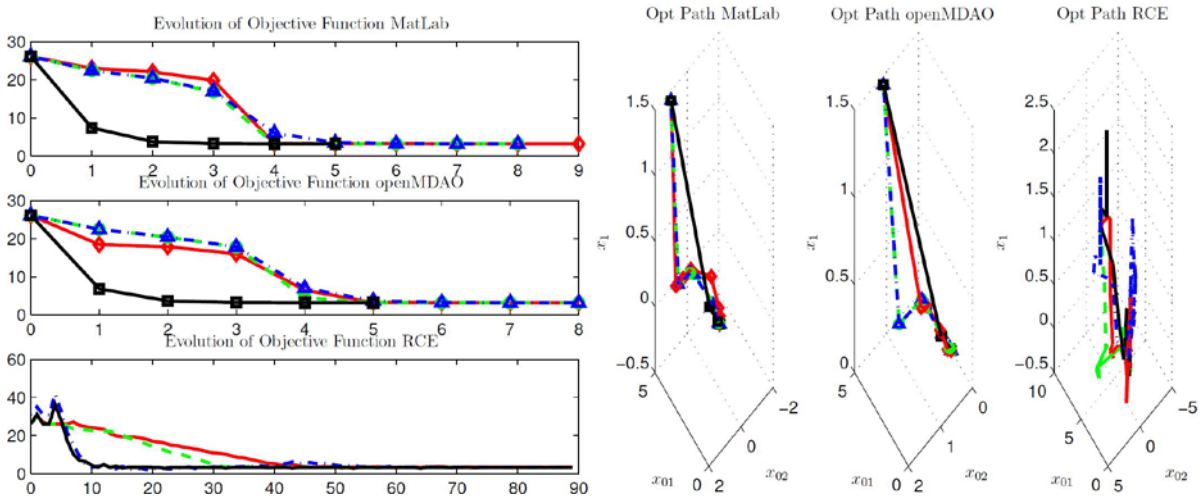

Figure 12: Objective function evolution (left) and optimization path (right) for the monolithic architectures (AAO-red, SAND-green, IDF-blue, MDF-black) of Sellar problem obtained with the three different optimization platforms

Figure 12 shows that the different optimization algorithm of RCE implementation results in an increased number of iterations. However the general behavior of the architectures is confirmed: MDF has the steepest descent, while the others have a similar behavior (except IDF in RCE implementation which is close to MDF).

4 AEROSTRUCTURAL AIRCRAFT SYSTEM DESIGN PROBLEM

In this section three architectures, MDF, IDF and CO, are applied to an overall aircraft design problem. A brief description of the tools used is provided in the MDF subsection.
The different tools are organized in a RCE workflow. RCE [11] is an open source workflow-driven integration environment developed by DLR to execute collaborative MDO problems. Tools and the workflow itself are the same ones used in AGILE project [19] for other tasks.

Another feature that accelerates the development of an aircraft design workflow is the use of CPACS data format to exchange information between disciplinary tools. CPACS [20] is a data schema written in xml containing all the information related to the aircraft. All the tools read and write on the same CPACS file; hence, the number of necessary connections is decreased as well as the possibility of store inconsistent or duplicated information.
The design and optimization problem is formalized in the following statement.

Table 11: OAD problem statement

Objective	Constraints	Disciplinary eq.
$\min F$ respect to $\boldsymbol{q}$	$s(\boldsymbol{q}) - span_{max} \leq 0$ $F - F_{max} \leq 0$	$[OEM, F_{max}] = \boldsymbol{D}_{struct}(\boldsymbol{q}, F)$ $F = D_{mission}(\boldsymbol{q}, OEM)$

The objective to minimize is the fuel mass calculated by the mission analysis tool. Design variables are indicated by the vector $\boldsymbol{q}$; they are defined in the Table 11 and in Figure 13.

Table 12: OAD design variables

$\boldsymbol{q}$	definition	boundaries	init.	ref.
AR	$= b^2/S$	$[7; 13]$	8	9
S	wing area	$[70m^2; 95m^2]$	$90m^2$	$81m^2$
τ	$= \frac{(t_{R,K,T}/c_{R,K,T})}{(t_{R,K,T}/c_{R,K,T})_{ref}}$	$[0.5; 2]$	1	1
λ_{out}	$= c_T/c_K$	$[0.25; 0.45]$	0.3	0.39
θ_K	kink twist	$[-1.5°; 2.5°]$	0°	0.01°
θ_T	tip twist	$[-3°; 1°]$	−1°	−1.3°

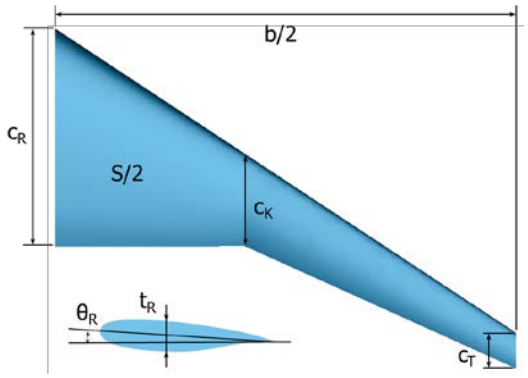

Figure 13: Wing parameters definition

All the design variables are shared among disciplines; there are not local design variables. Although design variables refer to the wing, the problem defined in Table 10 is an aircraft design problem because tools analyze the whole aircraft.
Several disciplines are considered in the workflow and all of them are actually coupled. However, for the sake of brevity, this paper underlines the coupling between two of them: structural and mission analysis. This is one of the strongest couplings in the aircraft design process and, all the architectures features are still well represented. Future works will highlight more distributed connections among disciplines.
The coupling state variables are the fuel mass (F), and the operating empty mass (OEM). Local state variables are not explicitly indicated but they are numerous. For instance, the aerodynamic tool is a panel method with 1000 panels, that means approximately 4000 scalar local state variables. Only one local design variable is exposed: the maximum fuel mass (F_{max}), which is obtained from the tank volume calculated by the structural tool.
Finally two constraints are considered: maximum wing span and maximum fuel. The maximum span is defined by the user at the start of the process and kept fixed throughout the design process. Here $span_{max} = 28\,m$, that is the standard limitation for the chosen aircraft category. The reference aircraft for all the MDO architectures is a large regional jet (same class of E190): 90 passengers and a range of 3500 km.

4.1 MDF Architecture for OAD Problem

In realistic design problems the huge number of local state variables hampers the use of SAND. Hence the first adopted monolithic architecture is MDF.

Table 13: MDF statement for OAD problem

Objective	Constraints
$\min F(\boldsymbol{q})$ respect to $\boldsymbol{q}$	$s(\boldsymbol{q}) - span_{max} \leq 0$ $F(\boldsymbol{q}) - F_{max}(\boldsymbol{q}) \leq 0$

First the Gauss-Seidel version of the MDF is considered. In this formulation the different disciplinary modules are organized in series. Thus, each tool writes its output on the CPACS file that it has received from the previous tool and sends it to the next tool.

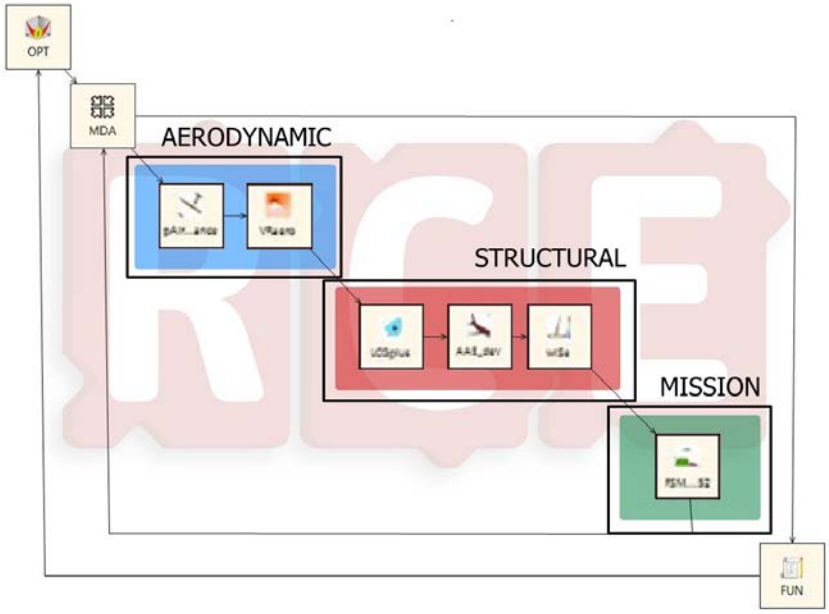

Figure 14: RCE workflow of MDF Gauss-Seidel architectures for the OAD design problem

The workflow's structure is represented in Figure 14. First, the aerodynamic block calculates the aircraft polars for different Mach and Reynolds numbers. Viscous considerations are added to results obtained by a lifting surface method (results of this tool are shown on the left wing in Figure 15).
Thereafter, the structural block calculates flight envelops, calculates the corresponding loads, and the wing box structure is seized according to the critical points of the envelope. A FEM is used for calculating stress in the wing box (see the right wing in Figure 15). Secondary masses, such as actuators system, control surfaces, etc., are also accounted in the process and represented by colored points on the right wing in Figure 15. Therefore, the entire mass breakdown and the operating empty mass are updated.
Finally, the mission analysis block provides the fuel mass necessary to fly the mission. The mission profile is chosen by the user and kept constant throughout the optimization process.

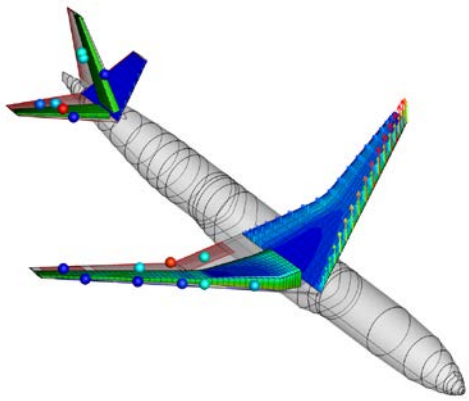

Figure 15: Models used for aerodynamic, structure and actuators system analysis

After the mission calculations, fuel mass and OEM, which are the coupling variables, are sent to the MDA converger. Once convergence on the coupling variables is reached, converged values of fuel mass and OEM are used to obtain objective and constraint functions. When the optimizer receives these values the iteration is concluded. At the next iteration the optimizer provides new design variables, which are the wing parameters: aspect ratio, area, thickness to chord ratio, taper and twist. With this data the CPACS aircraft representation is updated and sent to the MDA loop. The loop starts once also a starting value of the fuel mass is provided.
For the Jacobi version of the MDF the structural and the mission clusters are parallel, whereas the aerodynamic block is placed before the bifurcation, see Figure 16. Hence, a starting value of both fuel mass and OEM must be provided in order to start the MDA loop.

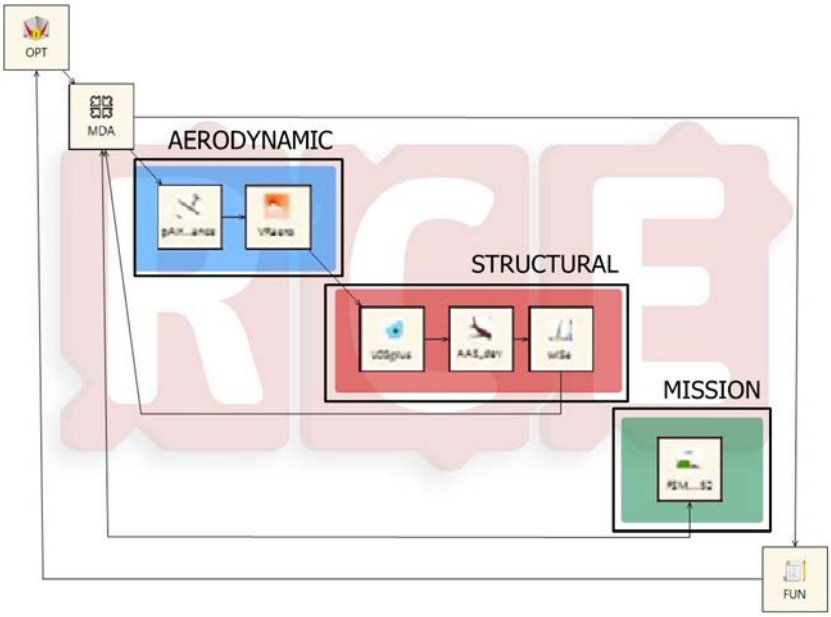

Figure 16: RCE workflow of MDF Jacobi architectures for the OAD problem

Here the trends of the objective and the constraint functions are reported for both the MDF variations, Figure 17. Only the constraint on span is represented because it is driving the optimization, whereas the constraint on tank volume is never active.
The adopted optimization algorithm spends the first 6 iterations (because this is the number of the design variables in this formulation) perturbing objective and constraints functions in a single design variable direction. After this collection of gradient information, the descendant path starts.

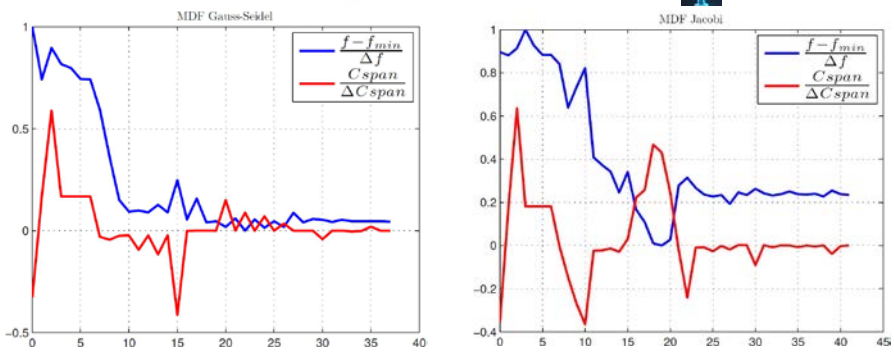

Figure 17: Objective function (blue) and span constraint function (red) of MDF Gauss-Seidel (left) and Jacobi (right) architectures for the OAD problem

Dimensionless values are reported, since the aim of these graphs is showing the trends. Gauss-Seidel version has a steeper and more regular slope than Jacobi. Regularity considerations are also shown by the constraints trend; in Jacobi version violations are stronger and more frequent (positive values of constraints function means constraint violation).

4.2 IDF Architecture for OAD Problem

In this architecture there is not MDA converger loop. Copies of couplings variables, $\hat{F}$ and $O\hat{E}M$, are created to allow independent discipline analyses. Consistency constraints guarantee discipline feasibility, but only at the end of the optimization process.

Table 14: IDF statement OAD problem

Objective	Constraints	Consistency
$\min F(\boldsymbol{q}, O\hat{E}M)$ respect to $\boldsymbol{q}, \hat{F}, O\hat{E}M$	$s(\boldsymbol{q}) - span_{max} \leq 0$ $F(\boldsymbol{q}, O\hat{E}M) - F_{max}(\boldsymbol{q}, \hat{F}) \leq 0$	$O\hat{E}M - OEM = 0$ $\hat{F} - F = 0$

Like in the Jacobi version of MDF, also in IDF there is a bifurcation in the workflow, see Figure 18. Again structural and mission blocks work in parallel, while the aerodynamic block is placed upstream to the bifurcation.

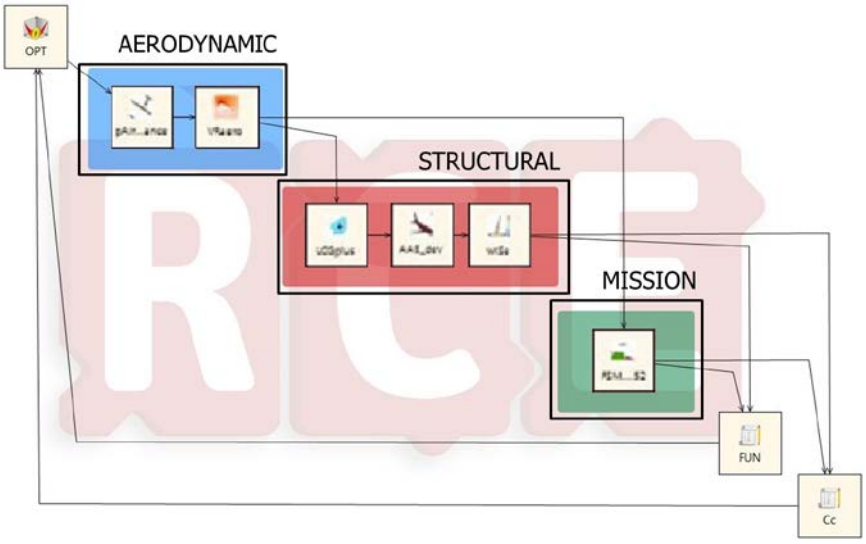

Figure 18: RCE workflow (right) of IDF architectures for the OAD problem

Figure 19 underlines that for this architecture global disciplinary feasibility is reached only at the end of the optimization process.

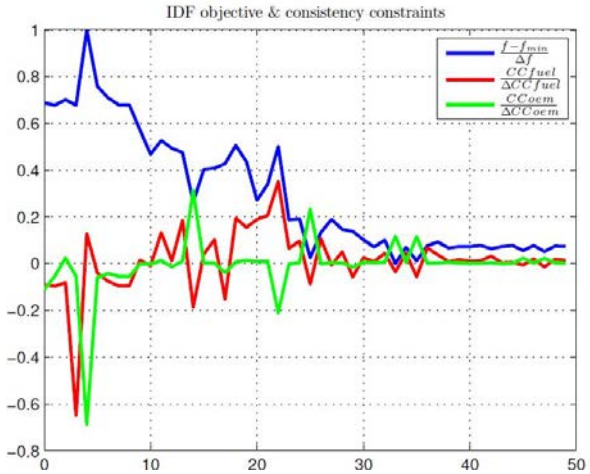

Figure 19: Objective function (blue), fuel (red) and OEM (green) consistency constraints of IDF architecture for the OAD problem

4.3 CO Architecture for OAD Problem

Among the two analyzed distributed architectures CO is chosen for the OAD problem. The aim here is to underline the effect of disciplines separation in distributed architecture. BLISS is discarded because disciplines analyses do not run independently (see paragraph on BLISS of Subsection 3.1). The system subproblem statement is the following.

Table 15: CO statement for OAD problem

System Subproblem		
Objective	Constraints	Consistency
$\min \hat{F}$ respect to $\boldsymbol{q}, \hat{F}, O\hat{E}M$	$s(\boldsymbol{q}) - span_{max} \leq 0$ $\hat{F} - F_{max}(\hat{\boldsymbol{q}}_s, \hat{F}) \leq 0$	$J_s^* = 0$ $J_m^* = 0$
Discipline 1 Subproblem		
Objective		Constraints
$\min J_s = \left\|\hat{\boldsymbol{q}}_s - \boldsymbol{q}\right\|^2 + \left\|O\hat{E}M - OEM(\hat{\boldsymbol{q}}_s, \hat{f})\right\|^2$ respect to $\hat{\boldsymbol{q}}_s$		none
Discipline 2 Subproblem		
Objective		Constraints
$\min J_m = \left\|\hat{\boldsymbol{q}}_m - \boldsymbol{q}\right\|^2 + \left\|\hat{F} - f(\hat{\boldsymbol{q}}_m, O\hat{E}M)\right\|^2$ respect to $\hat{\boldsymbol{q}}_m$		none

In this architecture each discipline tool is wrapped in its own discipline optimization subproblem. For this reason not only copies of coupling state variables ($\hat{f}, O\hat{E}M$) are needed, but also copies of design variables are defined for each discipline ($\hat{\boldsymbol{q}}_s$ for structure and $\hat{\boldsymbol{q}}_m$ for mission). In this way each discipline subproblem guarantees single discipline feasibility (minimizing respectively J_s and J_m).

After disciplines optimization the obtained value of consistency constraints (the so called post-optimal value: J_s^* and J_m^*) is sent to the system optimizer. The system optimization subproblem is responsible of minimizing the original objective function respect to design variables ($\boldsymbol{q}$) and the copies of coupling state variables ($\hat{f}, O\hat{E}M$).

Note that in this particular problem there are no local design variables, hence it is not necessary to define copies of them for the system subproblem.

The typical structure of the CO architecture, with discipline subproblems nested on the system subproblem, is shown on Figure 20.

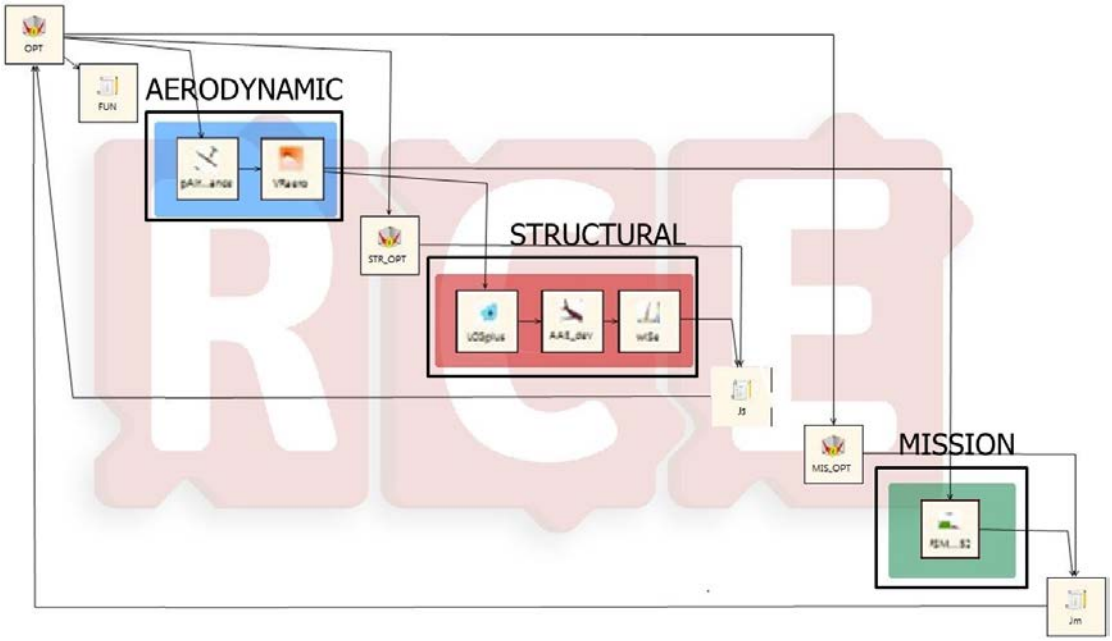

Figure 20: CO RCE workflow for the OAD problem

Although CO is considered a distributed version of IDF, comparison of Figure 21 with Figure 19 shows an important difference among them. In IDF consistency is enforced through constraints, hence it is reached just at the end of the process. In CO consistency is demanded to discipline subproblems optimization, that means it is satisfied from the first iteration.

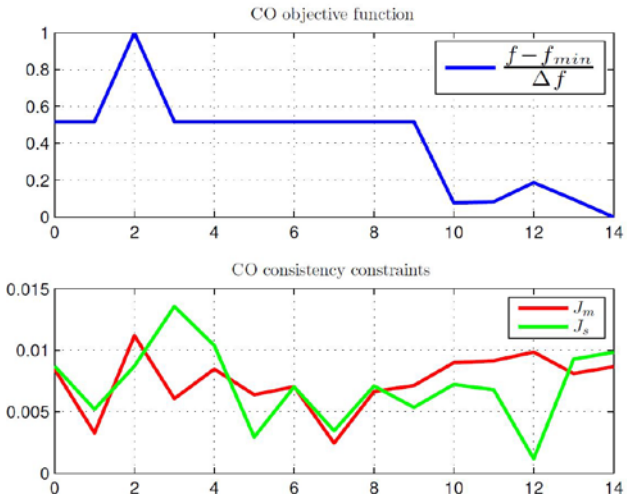

Figure 21: Objective (blue), fuel (red) and OEM (green) consistency functions of CO architecture for the OAD problem

Figure 21 shows also some other important features of CO implementation in RCE. As said in Subsection 4.1, the chosen algorithm spends the first iterations just collecting gradient information in each design variables direction. For CO the design variables of the system subproblem are the wing parameters, $\boldsymbol{q}$, and the copies of the coupling state variables, $\hat{F}$ and $O\hat{E}M$, 8 in total. The objective

function depends explicitly just on $\hat{F}$, thus objective function is constant for the first 8 iterations, except for the iteration where the perturbation direction is exactly $\hat{F}$ (that is the second iteration).

4.4 Architectures Comparison Results for OAD Problem

The optimal aircraft solutions obtained for each of the MDO architectures investigated are shown in Figure 22.

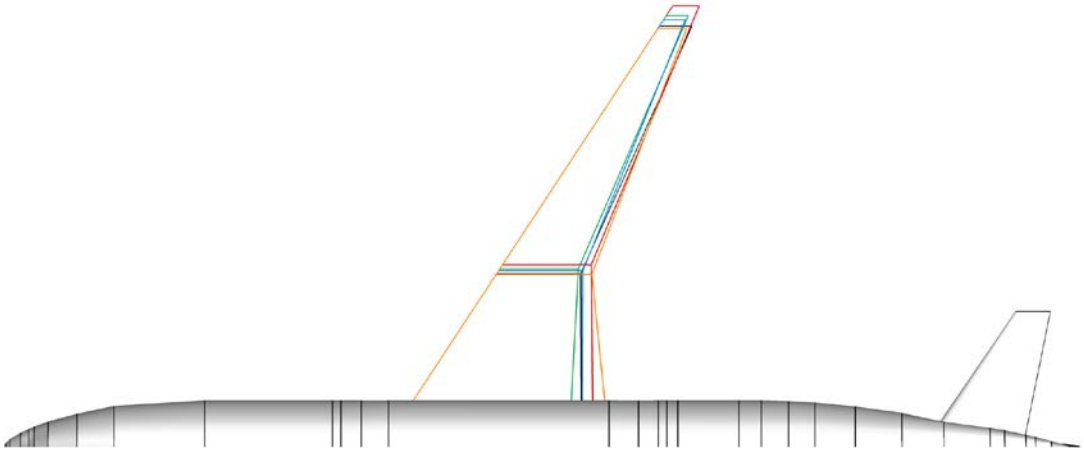

Figure 22: Optimal shape obtained with MDF Gauss-Seidel (green), MDF Jacobi (blue), IDF (red) and CO (orange) architectures (baseline in black)

Quantitative results for the comparison are provided in terms of objective function graph, Figure 23, and Table 16.

Table 16: Summary of architectures comparison on OAD problem

	MDFgs	MDFj	IDF	CO (interrupted)
Optimal obj.value	5141 (-9.4%)	5260 (-7.3%)	5144 (-9.4%)	5068 (-10.7%)
AR	11.10	11.10	11.10	7.82
$S\ [m^2]$	69.66	69.66	69.66	91.56
τ	0.93	0.61	0.94	1.13
λ_{out}	0.25	0.27	0.26	0.31
θ_K [°]	-0.0010	-0.0002	0.0005	0.0001
θ_T [°]	-1.10	-0.92	-0.97	-1.02
Iterations	38	42	50	15
Obj.fun calls	76	84	50	15
Struct. tool calls	76	84	50	418
Miss. tool calls	76	84	50	423

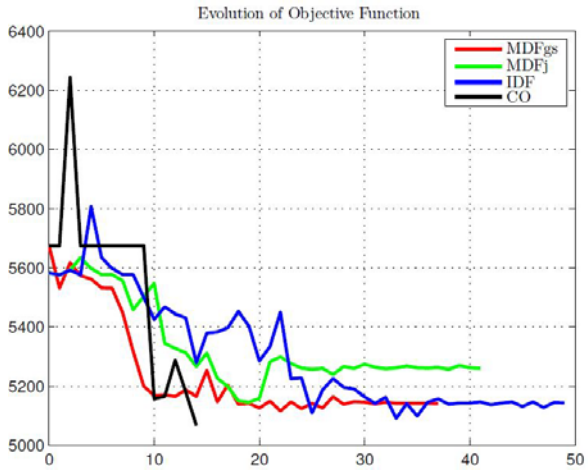

Figure 23: Objective functions of MDF Gauss-Seidel (red), MDF Jacobi (green), IDF (blue) and CO (black) for the OAD problem

Some trends found in the analytical test case are here confirmed. For instance MDF (Gauss-Seidel version in particular) has still the steepest descent among monolithic architectures, even if is a little bit more expensive in terms of calculation respect to IDF.
The number of calculations for CO is definitely greater than any of the monolithic architectures, and this is only partially compensated by the considerations that disciplines calculations run simultaneously. Approximately 450 evaluations of each analysis tool were spent for these iterations of CO architecture. Then, considering the good point reached, in terms of objective function and of consistency constraints, the process was manually interrupted. The consistency functions, J_s^* and J_m^*, for the last point before interruption have both a value of 10^{-2}.

5 CONCLUSIONS

In this paper a comparison among six multidisciplinary optimization architectures is presented. The architectures investigated include AAO, SAND, IDF and MDF among monolithic; CO and BLISS among distributed. The study is part of the AGILE project, which is developing the next generation MDO processes, and investigating multiple MDO techniques.
The study addresses 3 main challenges concerning the benchmarking of MDO architectures:

1. Implementation platforms' dependency: the MDO architectures have been implemented into three different platforms (Matlab, OpenMDAO and RCE), and results compared.
2. Scarcity of results on distributed architectures: the comparison here presented is not limited to monolithic architectures. Two distributed architectures are also considered: CO and BLISS.
3. Low dimensionality of use cases: in the presented study an overall aircraft design problem is also selected as use case. Such a use case included physics based simulation models typical for preliminary aircraft design, panel method for aerodynamic and FEM for structure analysis.

The comparison underlines the strong dependency of architectures' performance not only on the problem under investigation, but also on the analysis modules available, on the design environment of the organizations involved, and on the settings selected by the designer. For each of the MDO architecture (both the analytical and the OAD problem), the formulation and implementation are described. Quantitative results comparing the different architectures are also reported and discussed. Furthermore, each of the problem is represented and made available to the MDO community [21] as XDSM representation, and in the neutral format CMDOWS (Common MDO Workflow Schema) developed in AGILE project. The use of CMDOWS allows and facilitates the comparison among MDO architectures in order to choose the most suitable formulation for each design problem and organizational situation considered.
The insight gained by means of a realistic use case as an OAD problem will be exploited further in future comparisons of MDO architectures. Future works will also address more distributed coupling between disciplines, as well as the extension of the comparison among other distributed architectures.

6 ACKNOWLEDGEMENTS

The research presented in this paper 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 Programme (H2020-MG-2014-2015) under grant agreement n◦ 636202. The authors are grateful to the partners of the AGILE Consortium for their contribution and feedback.

REFERENCES

1. B. Cameron, E. Crawley, D. Selva; 2015; *System Architecture Strategy and Product Development for Complex Systems*; Pearson Education Limited
2. J. Sobieszczanski–Sobieski, A. Morris, M.J.L. van Tooren, G. La Rocca, W. Yao; 2015; *Multidisciplinary Design Optimization Supported by Knowledge Based Engineering*; John Wiley & Sons, Ltd.
3. AIAA Technical Committee; 1991; "Multidisciplinary Design Optimization (MDO), White Paper on Current State of the Art"

4. E.J. Cramer, J.E. Dennis, P.D. Frank, R.M. Lewis, G.R. Shubin; 1993; "Problem formulation for multidisciplinary optimization"; Center for Research on Parallel Computation Rice Univ., CRPC-TR93334
5. H.S. Lee; 2004; "Sequential approximate individual discipline feasible method using enhanced two-point diagonal quadratic approximation method"; *Master Thesis* (in Korean); Hanyang University
6. J.R.R.A. Martins, A. B. Lambe; 2013; "Multidisciplinary Design Optimization: A Survey of Architectures"; *AIAA Journal*; **51**(9)
7. R.D. Braun; 1996; "Collaborative Optimization: An Architecture for Large-Scale Distributed Design"; *Ph. D. Thesis*; Stanford University
8. R.D. Braun, P.J. Gage, I.M. Kroo, I.P. Sobieski; 1996; "Implementation and Performance Issues in Collaborative Optimization"; *6th AIAA/USAF/NASA/ISSMO Multidisciplinary Analysis and Optimization Symposium*
9. J. Sobieszczanski–Sobieski, J.S. Agte, R.R.Jr.Sandusky; 2000; "Bilevel Integrated System Synthesis"; *AIAA Journal*; **38**(1); pp. 164 - 172
10. I. van Gent, G. La Rocca, M.F.M. Hoogreef; 2017; "CMDOWS: A Proposed New Standard To Store And Exchange MDO Systems"; *6th CEAS Conference*; Bucharest; October
11. http://rcenvironment.de/
12. R.S. Sellar, S.M. Batill, J.E. Renaud; 1996; "Response Surface Based, Concurrent Subspace Optimization for Multidisciplinary System Design"; *34th Aerospace Sciences Meeting and Exhibit*
13. R.T. Haftka, J. Sobieszczanski–Sobieski, S.L. Padula; 1992; "On Options for Interdisciplinary Analysis and Design Optimization"; *Structural Optimization*; **4**(2); pp. 65 - 74
14. K.F. Hulme, C.L. Bloebaum; 2000; "A Simulation-Based Comparison of Multidisciplinary Design Optimization Solution Strategies Using CASCADE"; *Structural and Multidisciplinary Optimization*; **19**(1); pp. 17 - 35
15. S.I. Yi, J.K. Shin, G.J. Park; 2008; "Comparison of MDO methods with mathematical examples"; *Structural and Multidisciplinary Optimization*; **35**; pp. 391 – 402
16. R.E. Perez, H.H.T. Liu, K. Behdinan; 2004; " Evaluation of Multidisciplinary Optimization Approaches for Aircraft Conceptual Design"; *10th AIAA/ISSMO Muldisciplinary Analysis and Optimization Conference*
17. N.F. Brown, J.R. Olds; 2006; "Evaluation of Multidisciplinary Optimization Techniques Applied to a Reusable Launch Vehicle"; *Journal of Spacecraft and Rockets*; **43**(6); pp. 1289 - 1300
18. J.T. Allison, M. Kokkolaras, P.Y. Papalambros; 2007; "On Selecting Single-Level Formulations for Complex System Design Optimization"; *Journal of Mechanical Design*; **129**(9); pp. 898 - 906
19. P.D. Ciampa, B. Nagel; 2017; "The AGILE Paradigm: the next generation of collaborative MDO"; *18th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*; Denver; June, 5 - 9
20. http://cpacs.de/
21. https://www.agile-project.eu/