An agent-supported approach for the collaborative design of complex systems

Moncef Hammadi, Amir Guizani, Jean-Yves Choley, Andreas Kellner, Peter Hehenberger

An agent-supported approach for the collaborative design of complex systems

Moncef Hammadi,* Amir Guizani and Jean-Yves Choley

Laboratoire Quartz EA 7393, Supmeca, 3 rue Fernand Hainaut
93400 Saint-Ouen, France
moncef.hammadi@supmeca.fr

Andreas Kellner and Peter Hehenberger

Institute of Mechatronic Design
and Production, Johannes Kepler Universität
Altenberger Straße 69, 4040 Linz, Austria
peter.hehenberger@jku.at

Received (Day Month Year)
Accepted (Day Month Year)

A novel approach for partitioning and coordinating the collaborative design optimization of complex systems is described. A partitioning metric has been formulated to select the best partitioning solutions among the total possibilities of dividing the complex design optimization problem. Then, an agent-supported approach is used for the coordination of the collaborative design optimization. The approach has been applied to the case of a preliminary design of an electric vehicle, to demonstrate how various agents can effectively communicate with each other to provide support to the collaborative design optimization of complex systems.

Keywords: Complex design optimization; collaborative design; agent-supported approach.

1. Introduction

The design process of engineered complex systems requires the synergy between several engineering disciplines. A complex system is defined as a system of a large number of parts that interrelate in a non-simple manner [1]. In the field of engineering design optimization of multidisciplinary systems, such as electric vehicles, engineers have to deal with a big number of design variables, design constraints and design objectives, which are interrelated in non-simple manner. For this purpose, the complex system design should be optimized as a whole to maximize its added value. However, some issues are facing the design synergy of complex systems. Indeed, the whole design work has to be distributed on multidisciplinary development teams to cut down the problem complexity and the computing cost. Design teams have to collaborate between each other to reduce the design errors and inconsistencies. In addition, several modeling and simulation tools are required to verify and

*Corresponding author.

validate the design. Furthermore, different abstraction levels of models have to be considered during the design process phases of the complex system. Consequently, the complex system design should be divided into partitions easier to manage in a collaborative and distributed design context.

Dealing with the design integration issues, several previous works proposed approaches and frameworks to integrate collaborative design optimization ([2], [3], [4], [5], [6]). Most these works are based on multidisciplinary design optimization (MDO) techniques [7] and more recently agent-based paradigm [8]. However, research activities are still needed in the area of collaborative distributed design to reduce costs and development time of complex products.

A new approach is proposed in this paper for partitioning and coordinating the design of complex systems to facilitate the collaborative optimization of the design. In this approach, the designer defines the maximum number of partitioning solutions, next he selects the best solutions using a partitioning metric. Then, an agent-supported technique is proposed to coordinate the collaborative optimization. The approach has been applied to the case of a preliminary design of an electric vehicle to demonstrate how various agents can effectively communicate with each other to provide support to the collaborative design optimization.

In this paper, Section 2 gives an overview of the work related to our research. Section 3 describes our approach for partitioning and coordinating the optimization of complex system design. Section 4 presents an application to a case of preliminary design of an electric vehicle to illustrate how the design optimization is supported by the developed agents. Finally, conclusions are given in section 5.

2. Related works

The most employed method to optimize a whole complex system is the MDO approach. MDO uses optimization methods to solve design problems incorporating a number of disciplines. Several MDO techniques have been developed in this field such as concurrent subspace optimization [9], multidisciplinary feasible design [2], collaborative optimization CO [3], multi-objective collaborative optimization MOCO [10], analytical target cascading ATC [11] and bi-level integrated system synthesis [12]. Such approaches can be applied to the optimization of large multidisciplinary systems such as planes or vehicles, but also for optimizing smart complex products such as power converters [13].

For instance, ATC is an MDO technique based on propagating system targets, representing requirements or desirable properties, through a hierarchical representation of the system to achieve a feasible system design satisfying these targets [14]. ATC technique uses, in most of cases, quadratic penalty functions at both system

level and local sub-problem level to coordinate the overall distributed design. However, the distributed set of sub-problems must be solved for a given set of weights, because of the non-separable nature of the quadratic penalty function [7]. This makes coordination difficult for the collaborative design optimization of large complex systems.

Coordination of design tasks, and therefore design analysis with simulation, can however be managed using agent technology.
A software agent is a program whose behavior can be characterized with autonomy, cooperation and learning [15]. Synergy between agent paradigm and simulation can be seen from different viewpoints. For instance, agent simulation concerns the use of simulation techniques to simulate the behavior of agent systems. Agent-based simulation is the use of agent techniques, as intelligent piece of software, to generate simulation models. Whereas, agent-supported simulation deals with the use of intelligent agents to enrich simulation environments [16].
Several agent-oriented applications were developed last years in different industrial applications including manufacturing, process control, transportation and telecommunication [17].

More recently, agent-based techniques are being used increasingly in practice for the distributed multidisciplinary design of complex systems. For instance, a prototyping environment for integrating Web and agent technologies for MDO was presented by Wan et al. [18]. Agents were utilized to encapsulate the functionalities related to the design process and product performance simulation. Likewise, an agent approach using a negotiation mechanism based on a price schedules decomposition algorithm, originally developed in economics, was proposed by Lin et al. [19]. The mechanism searches for globally optimal designs, where no participant is necessary to own full knowledge of the entire design space. Furthermore, multi-agent design paradigm has also been used to tackle the issue of multi-level modeling in mechatronic design [20]. Analysis models belonging to different abstraction levels were encapsulated into agents linked with relationships and managed with design rules. Moreover, a comparative study between multi-agent systems and multi-disciplinary optimization approaches was proposed in the domain of multidisciplinary collaborative building design by Ren et al. [21]. In this last study, it was concluded that agent technology could be adopted for the collaboration process in MDO to make the process more efficient. However, much work is required to develop sophisticated theory and approaches in order to make agent techniques more powerful and mature for the distributed multidisciplinary design optimization of complex systems.

Our work comes in this direction in order to improve the use of agent technologies in the collaborative design process of complex engineering systems. Indeed, the multidisciplinary design of complex systems should be optimally partitioned

in order to develop efficient software agents that facilitate the coordination and communication between agents during the design optimization process. For this, we propose a new formal approach for partitioning and coordinating the design of complex systems.

3. Description of the approach for partitioning and coordinating the design of complex engineering systems

To obtain the optimal design of a complex system in a distributed collaborative design context, we propose firstly to divide the optimization problem into partitions easier to solve, then secondly we coordinate the overall design optimization using agent technology.

3.1. *Optimal partitioning of the design optimization*

A solution for maximizing the added value of a complex product is to formulate and solve an optimization problem. Our approach for partitioning the complex design is based on the specification of the optimization formulation of the complex system design by specifying the objectives, the constraints and the design variables.

We consider a problem of a complex design optimization defined with a set of m objective functions F, n constraints G, r design variables X, s analysis models A and t partitioning possibilities P, respectively noted as: $F = \{f_1, f_2, .., f_m\}$; $G = \{g_1, g_2, .., g_n\}$; $X = \{x_1, x_2, .., x_r\}$, $A = \{a_1, a_2, .., a_s\}$ and $P = \{p_1, p_2, .., p_t\}$; m, n, r, s and $t \in \mathbb{N}$.

The optimization problem of the design can be formulated as the following:

$$\begin{aligned} &min\ F(X) \\ &w.r.t\ X \\ &s.t\ G(X) \end{aligned} \tag{1}$$

The objective of our partitioning approach is to find the optimal decompositions combining subsets of objectives, constraints, design variables and analysis models.

Analysis models are used to evaluate the objective functions and constraints which are interrelated with the design variables. Analysis models depend on several design factors such us the different use cases to be evaluated and the hierarchical decomposition structure of the complex system into subsystems and components. They also can be dependent on the degree of coupling between the design variables, constraints and objective functions related to the different subsystems.

A Design partitioning configuration is defined with the association of subsets of objective functions, design constraints, design variables and analysis models. Objective functions, constraints and design variables can be associated to analysis models according to some criteria related to the overall computation time, the computation time of every analysis model and the coupling degree between constraints, objective functions and design variables. The coupling degree can be measured as the amount of information to be exchanged between the different variables.

The first step of our partitioning method is to define the maximum partitioning possibilities. For this, we need to draw a network graph defining all the possible partitioning solutions. An example of such networks will be treated in the example of next section.
In the second step, we search the optimal way of partitioning the optimization problem. For this we define, for every partitioning solution, a metric that we call *partitioning metric* defined as:

$$P_m = \frac{1}{N_a} \cdot \frac{1}{C_c} \cdot \sum_{j=1}^{N_a} \frac{Co_j}{Cx_j} \tag{2}$$

where: N_a is the number of analysis models present in one partitioning solution, C_c is the degree of complexity of coordination between analysis models in one partitioning solution, Co_j is the degree of coupling between design variables within one analysis model j and Cx_j is the degree of complexity to implement one analysis model j.
The degree of coupling between design variables within one analysis model Co_j can be determined by summing all the local degrees of coupling between each two design variables that represent the objective functions and constraints. The local degrees of coupling can be determined based on the experience of the designers using the analysis models or by using parametric sensitivity analysis of the design variables. We choose the local degrees of coupling to be between 0 for non-coupled variables and 1 for strongly coupled variables. The global coupling degree of the partitioning solution Co_j can however be greater than 1.
The degree of complexity to implement one analysis model Cx_j can also be determined based on the experience of designers. We chose values of Cx_j to be between 0 for an easy to develop and implement analysis models and 10 for very complicated ones. However, a judicious choice of the maximum value of Cx_j may be the value that normalizes all the partitioning metrics. So that all the values of the partitioning metric are less than one, which makes easier the comparison between partitions.
The degree of complexity to coordinate between analysis models C_c in a given partitioning configuration is proportional to the number of analysis models N_a used in the same partition. For this, we choose C_c to be simply equal to N_a.

Once all the parameters of the partitioning metric are determined, P_m can easily be calculated using equation 2. In the case of large complex design problems with a

big number of design variables, one optimization problem can be formulated based on maximizing P_m to find the optimal partitioning solutions.

After the partitioning phase, designers develop the analysis models and define the optimization workflow. However, a coordination between analysis models is required to reach the overall optimization objectives. For this, we propose an agent-supported approach to coordinate the optimization process.

3.2. *Coordination of the global optimization using agent-supported approach*

The design optimization workflow is mainly made up of optimization activities controlled by software agents. An optimization activity is the task of calling an optimizer controlling an analysis model, or a list of coupled analysis models, to execute an optimization related to one partition according to predefined design constraints and objectives. We propose to use two types of agents, a coordinating agent (or coordinator) and several design agents. Both types of agents can exchange messages between each other and participate in the design optimization process. The messages exchanged can be the orders to execute optimization or the orders to change input parameters related to design variables or constraint limits. They can also be the orders to retrieve results and send them to other agents. Based on the local optimization results sent by the design agents, the coordinating agent searches for the global optimal design that satisfies all local optimizations.
The optimization process workflow is not established beforehand, but the coordinating agent adapts the process depending on the evolution of the optimization process. The coordinating agent organizes therefore the negotiation process with the design agents to reach a trade-off about the optimal design.

For example, in a hierarchically decomposed engineered system, made of subsystems and components, the coordinating agent starts the first optimization to search the target points to be used by design agents. Then the design agents solve the local optimizations and send the solutions back to the coordinating agent. The coordinating agent measures the discrepancy between the solutions coming from the different design agents and then suggests new targets, by solving a new optimization problem, until reaching the final solutions.
In the case of non-hierarchically decomposed engineered system, e.g. multi-use case decomposition (decomposition of the optimization problem according to the use cases of the system), the design agents start parallel optimizations. Then the coordinating agent defines the order of optimizations to be performed by the design agents to reach the final solutions, according to the solutions found by the design agents. In this case, the coordinating agent proposes an execution order of optimization by starting with the design agent having the less number of solutions and progressing to the design agents which have more solutions. Because every design

agent in the research process will filter the solutions by applying the local design constraints. Therefore the number of research iterations will rapidly decrease when the research starts with the design agent having less solutions. Consequently, the computing time of the overall optimization process will be reduced.

Figure 1 shows a sequence diagram that illustrates how design agents exchange messages and data during optimization process with three design agents and a coordinator. In order not to overload the diagram with all the operations, we have

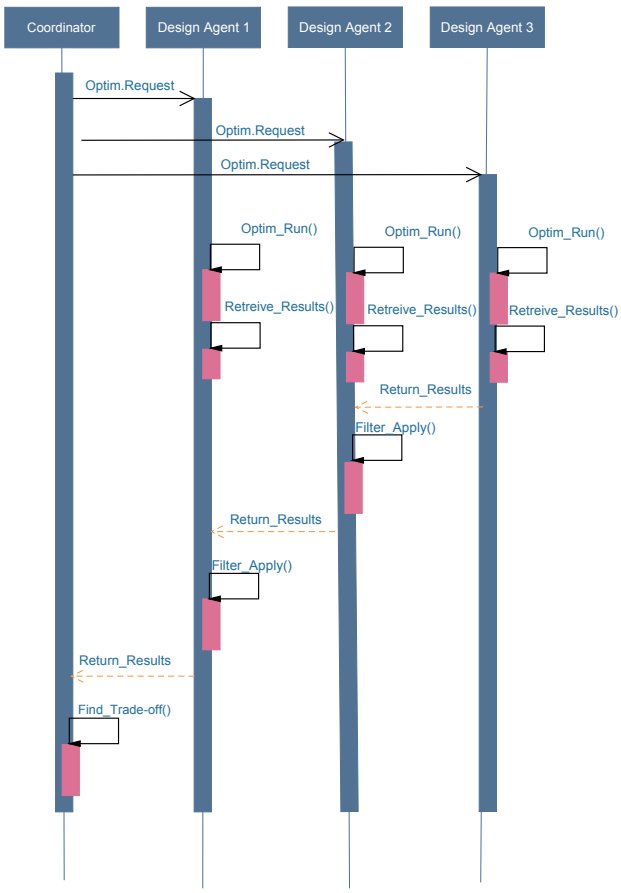

Fig. 1. Example of agent activities during optimization process

chosen to show only some of them to facilitate the description.

In this generic example, the *coordinating agent* starts by sending an optimization request to the three design agents to perform activities of optimization. This step

could be accompanied with a sending of target values to be taken into consideration in the optimizations (not represented on the diagram). Every design agent invokes the optimization task by running the optimizer that controls the simulation tool supporting the analysis model. After retrieving results, every design agent sends a message with the number of solutions found to the *coordinating agent*. Then the *coordinating agent* defines an order of exchange of results to be filtered and keep only solutions satisfying the three design agents. In the case represented in figure 1, it is supposed that the *Design Agent 3* has the smallest number of solutions and *Design Agent 1* has the maximum number of solutions. So *Design Agent 3* sends the results found to *Design Agent 2* to be filtered. In the same way filtered results are sent to *Design Agent 1* and finally to the *coordinating agent*. Based on these results, the *Coordinating agent* searches for the common solutions that satisfies the three design agents (Find-Trade-off). In the case no solutions are found, the *coordinating agent* proposes new target values or suggests some decisions about the modification to bring in the optimization problem, such as modifying the interval limits of the design variables.

3.3. *Implementation of the agent-supported platform*

The Java Agent DEvelopment framework (JADE) [a] has been chosen to implement the agent-supported approach for the collaborative design optimization. JADE is an open-source framework for agent-based applications compatible with the Foundation for Intelligent Physical Agents (FIPA) [b] standards and specifications. JADE is also a cross-platform and works on multiple operating systems, enabling agent platforms to be distributed across different machines.

Agents in JADE are managed with containers and platforms. The first container is called the main container. It holds two specific agents: the Agent Management System (AMS) and the Directory Facilitator (DF). AMS is a registry directory that assigns a unique and valid identifier to each registered agent, container and platform; enabling correct location and invocation. The DF is a registry directory often compared to the "Yellow Pages" phone book that offers query services.
The communication between agents within the same agent platform or across platforms are coordinated by the Message Transport System (MTS). The coordinating agent (CA) resides in a facilitator container, and each design agent (DA) is deployed in a separate container. The containers can be distributed separately in the network environment and recognized by different IP addresses.
Figure 2 gives an overview about the JADE platform for the collaborative design optimization.
Users can interact with the JADE platform through graphical interfaces, either

[a] http://jade.tilab.com/
[b] http://www.fipa.org/

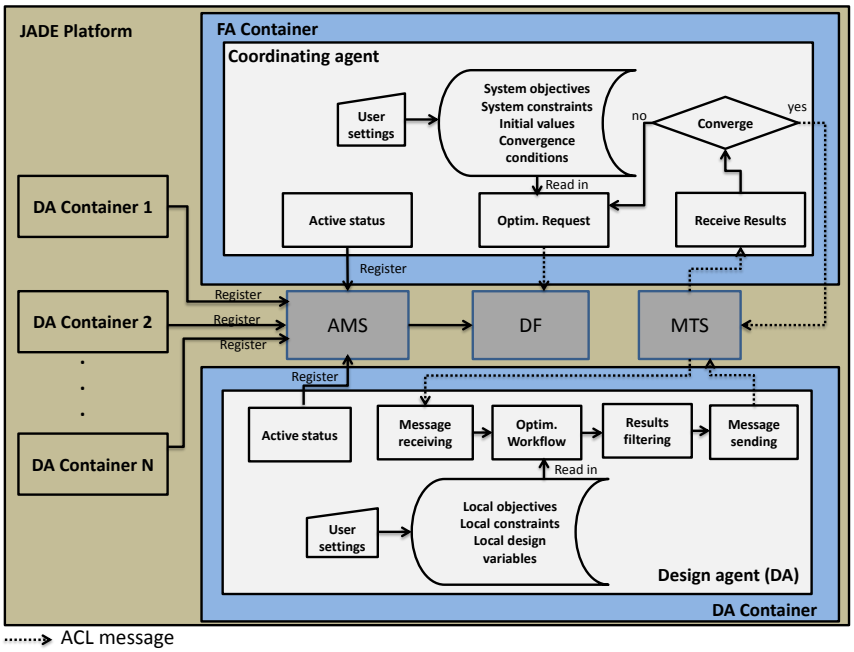

Fig. 2. Framework of the agent-supported design optimization

for the setting of the optimization problem parameters or, if needed, during the optimization process. The optimization workflow settings are divided between an administrator (or system designer) and the designers interacting with each design agent. The administrator introduces the system constraints in the coordinating agent and the designers specify the constraints and objectives relative to each design agent.

4. Application to a case of preliminary design of an electric vehicle

In this application, we begin by specifying the design optimization problem of an electric vehicle. Next, we apply our methodology to the partitioning of the design optimization to define the analysis models required. Then we present the mathematical formulation of the analysis models to be used for the optimization computing. After that, we apply the agent-supported method for the coordination of the design optimization.

4.1. *Specification of the design problem*

We consider the case of a preliminary design of an electric vehicle. The objective of this study is to optimize the electric motor and a one-ratio gearbox to fulfill the performance requirements related to three test cases: the acceleration test, gradeability (inclined road) and maximal velocity.

We define two objective functions f_1 and f_2 which are both depending on the two design variables x_1 and x_2, respectively corresponding to the back-electromotive-force (EMF) of the electric machine constant and the one-ratio gearbox. We also define three constraints g_1, g_2 and g_3 which are respectively related to the acceleration test, gradeability test and maximal velocity test. The two functions to be minimized and the three constraints can therefore be expressed as:

- f_1: The maximum torque at the input of the gearbox should be minimized to remain less than 250 $N.m$.
- f_2: The maximum peak of electric power required by the vehicle should be minimized to remain less than 200 kW.
- g_1: The vehicle velocity on a 0% grade road after 10 seconds of start-up ($V_{10@0\%}$) shall be grater or equal to 100 km/h.
- g_2: The vehicle velocity in a road with a maximum grade equal to 30% after 10 seconds of start-up ($V_{10@30\%}$) shall be grater or equal to 20 km/h.
- g_3: The vehicle maximum velocity on a horizontal road ($V_{max@0\%}$) shall be grater or equal to 120 km/h.

4.2. *Partitioning of the design optimization problem*

To simplify the study we will consider, for optimization computing, one only Modeling/Simulation tool (t_1) and one only optimization tool (t_2) with one optimizer (optimizing algorithm o_1).
Figure 3 shows, in these conditions, the network graph of maximum partitions.

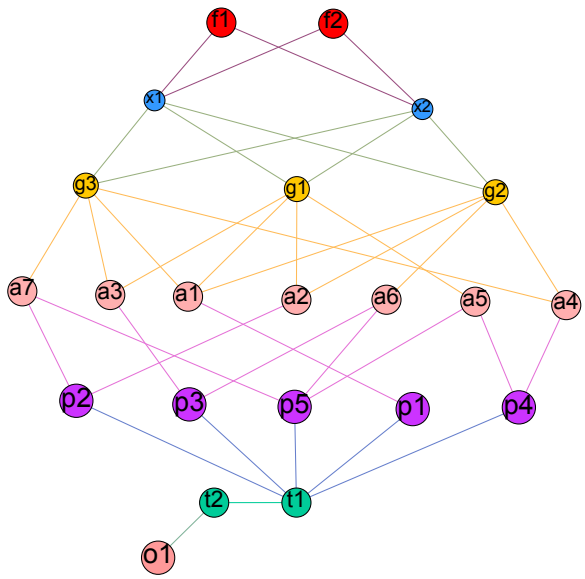

Fig. 3. The network graph of maximum partitions of the electric vehicle optimization problem

In order not to overload the graph, the links between the analysis models and objective functions are not represented in the graph, wherein the two objective functions are related to all analysis models.
We have 7 possibilities for defining the analysis models (a_j for $j = 1..7$) and 5 partitioning solutions (p_t for $t = 1..5$). For instance, the case of p_1 corresponds to an all-at-one (AAO) case, in which the three constraints and the two objective functions are evaluated by the same analysis model (a_1). The second partition solution p_2 is made of two analysis models: a_2 (to evaluate g_1 and g_2) and a_7 (to evaluate g_3). The partitioning solution p_3 is also made of two analysis models: a_3 (to evaluate g_1 and g_3) and a_6 (to evaluate g_2). Likewise, the partitioning solution p_4 is made of two analysis models: a_4 (to evaluate g_2 and g_3) and a_5 (to evaluate g_1). The last partition p_5 is made of three analysis models: a_5, a_6 and a_7 to evaluate respectively g_1, g_2 and g_3.
Figure 4 shows the five partitioning possibilities and their links with the 7 possible analysis models.

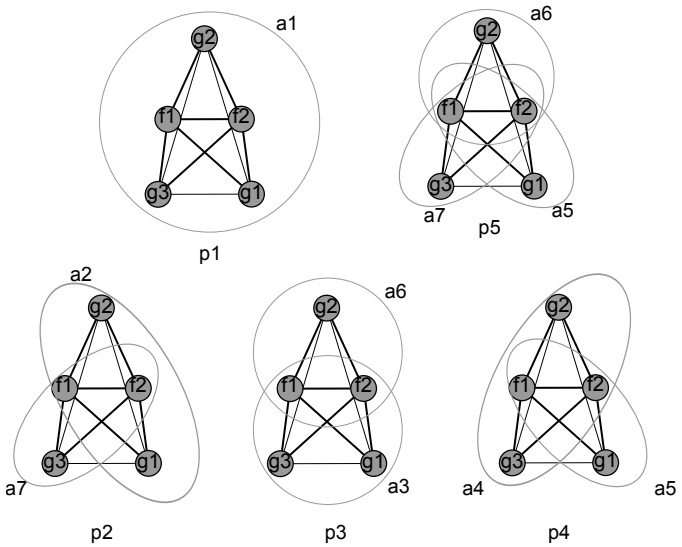

Fig. 4. Five partitioning solutions

In this figure, three line thicknesses are used to distinguish between coupling degrees between objective and constraint functions.

Table 1 gives the coupling values between objective functions and constraints. The two objective functions are strongly coupled with the three constraints, so that we assigned 1 as coupling value. The constraint g_2 has a loose coupling with g_1 and g_3, because they are not related to the same grade slope. However, g_1 is moderately coupled with g_3, since they are assigned to the same grade slope, but for different time instants. It should be remarked that coupling values are elaborated depending

on sensitivity analysis of design variables.

Table 1. Coupling values between objective functions and constraints

	f_1	f_2	g_1	g_2	g_3
f_1	x	1	1	1	1
f_2	1	x	1	1	1
g_1	1	1	x	0.1	0.5
g_2	1	1	0.1	x	0.1
g_3	1	1	0.5	0.1	x

The degree of coupling within every analysis model and the degree of complexity of implementing the analysis models are given in table 2.

Table 2. Degree of coupling and degree of complexity of analysis models

	a_1	a_2	a_3	a_4	a_5	a_6	a_7
C_o	7.7	5.1	5.5	5.1	3	3	3
C_x	10	10	1	10	1	1	1

Results show that the analysis model (a_1) has the higher degree of coupling (7.7), which reflects the high level of integration. A high level of integration is required when the interaction between the design variables, objective functions and constraints has an important influence on the system behavior. However, in the case of complex systems with hundreds of design variables, it is often impossible to have only one analysis model for the optimization of all the system.

To search for the optimal partitioning solutions, we calculate the values of the partitioning metric using equation 2 for the five solutions. These values are given in table 3.

Table 3. Partitioning metric values

	p_1	p_2	p_3	p_4	p_5
N_a	1	2	2	2	3
C_c	1	2	2	2	3
P_m	0.77	0.8775	2.125	0.8775	1

Results in the last table show that the all-at-one solution (P_1) has the lowest $P_m = 0.77$, which reflects the difficulty to compute $V_{10@0\%}$ and $V_{10@30\%}$ in the same analysis model. The solution P_1 reflects also the low distributivity level of design tasks. Likewise, solutions P_2 and P_4 have low P_m values due to the same reasons as P_1, although both of them have two analysis models. Despite the biggest

number of analysis models, P_5 is not the best solution, because of the complexity of coordination. P_3 is however the best solution, with its best compromise between decomposition (two analysis models) and integration.
Figure 5 shows the network representation of the best partitioning solution P_3.

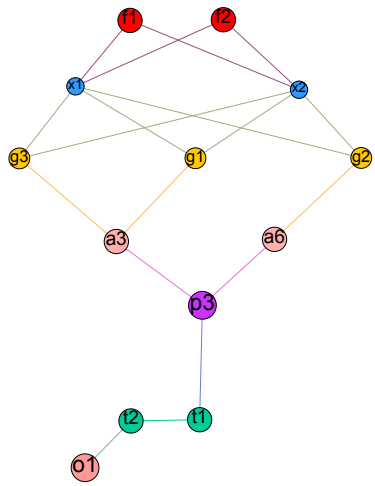

Fig. 5. Optimal partitioning solution P_3

Based on the partitioning study, the mathematical modeling of the analysis models can now be elaborated.

4.3. *Mathematical modeling of analysis models*

The acting forces on the electric vehicle are shown in figure 6.

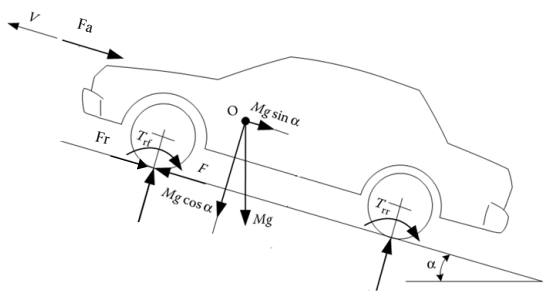

Fig. 6. Acting forces on the electric vehicle

The total resistive force applied on the vehicle is defined with:

$$F_t = F_a + F_r + F_p. \tag{3}$$

F_a is the resistive aerodynamic drag force defined with:

$$F_a = \frac{1}{2}.\rho.C_x.S.V^2. \tag{4}$$

Where ρ is the air density, C_x is the aerodynamic drag coefficient, S is the vehicle frontal area and V is the vehicle velocity.
F_r is the rolling resistive force defined with:

$$F_r = f_r.M.g.cos\,\alpha. \tag{5}$$

Where f_r is rolling coefficient, M is the vehicle mass, g is the gravitational acceleration constant and α is the grade angle.
F_p is the resistive force due to the grade angle of the road defined with:

$$F_p = M.g.sin\,\alpha. \tag{6}$$

The vehicle velocity is calculated using the dynamic equation:

$$M.\frac{dV}{dt} = F - F_t. \tag{7}$$

F is the tractive force due to the electric motor that can be calculated with:

$$F = C_m.\frac{R_g}{r_w}. \tag{8}$$

C_m is the resultant torque, R_g is the gearbox ratio and r_w is the wheel radius. The electric motor torque is calculated with the dynamic equations of the electric motor:

$$C_m - C_r = J_e.\frac{d\omega_m}{dt}. \tag{9}$$

$$C_m = K_m.i. \tag{10}$$

$$E - R.i - L.\frac{di}{dt} = K_m.\omega_m. \tag{11}$$

Where C_r is the equivalent torque of the resistive forces, J_e is the equivalent moment of inertia of the vehicle, K_m is the electric constant of back-electromotive-force (EMF), i is the electric current, E is the input electric voltage, R the internal resistance of the electric motor, L is the electric motor inductance and ω_m is the electric motor speed, which is related to the vehicle velocity with:

$$\omega_m = V.\frac{R_g}{r_w}. \tag{12}$$

After developing the mathematical models of the electric vehicle, the system has been modeled using Modelica language [22] and solved with the OpenModelica[c]

[c]https://www.openmodelica.org/

algebraic differential equation solver. Table 4 gives the parameters that have been used during optimization tests. The default values of the gear ratio (R_g) and EMF (K_m) are respectively 2.0 and 1.6.

Table 4. Electric vehicle parameters

	Description	***value***	***unit***
M	Vehicle mass	*1540*	kg
r	Air density	*1.2*	$kg.m^3$
S	Vehicle frontal area	*1.8*	m^2
f_r	Rolling coefficient	*0.013*	-
C_x	Aerodynamic drag coefficient	*0.2*	-
a	Grade angle	*0%, 30%*	-
g	Gravitational acceleration constant	*9.81*	$m.s^{-2}$
r_w	Wheel radius	*0.28*	m
R	Internal resistance of the motor	*0.2*	Ω
L	Internal inductance of the motor	*0.06*	H
J_m	Moment of inertia of the motor	*0.1*	$Kg.m^2$
R_g	Gearbox ratio	$3 \leq R_g \leq 13$	–
K_m	Back-electromotive-force	$0.2 \leq K_m \leq 1$	$N.m.A^{-1}$

The basic analysis models for the different partitioning solutions can therefore be determined using the Modelica model developed, by modifying the model inputs or the grade angle parameter.
The objective of the next section is to coordinate the optimization of the electric constant of back-electromotive-force (K_m) of the electric motor and the gearbox ratio (R_g) using the multi-agent approach.

4.4. *Application of the agent-supported approach for the collaborative optimization*

Among the five partitioning possibilities previously identified, the solution $P_3(P_m = 2.125)$ is the best one to be considered. This solution requires two design agents and a coordinator. However, for illustrative reasons, we will deal with the solution $P_5(P_m = 1)$, which requires three design agents and a coordinator. The approach is the same for both cases but it is more detailed for the latter.

The multi-objective optimizations are performed in this study using the Non-dominated Sorting Genetic Algorithm-II (NSGA-II) [23] with ModelCenter software [d]. Results of optimization are stored in Excel format, therefore the developed agents are able to process them.

Figure 7 shows the Pareto fronts obtained by optimization in the case of P_5. The order in which the optimizations were performed begins with the acceleration test, then the maximum velocity test and ends with the gradeability test.

[d] http://www.phoenix-int.com/software/phx-modelcenter.php

After the multi-objective optimization, controlled by the first design agent responsible for the acceleration performance test-case, the Pareto-front represented with figure 7-a is generated. For this optimization, the first limits of the design variables are: $0.2 \leq K_m \leq 1.0$ and $3 \leq R_g \leq 13$. The optimal solutions obtained in figure 7-a give new limits which are $0.26 \leq K_m \leq 0.5$ and $5.9 \leq R_g \leq 13$. These limits are then used for the second optimization relative to the maximum velocity (figure 7-b). In the same way, the agent responsible of the second optimization determines the new design variable limits, which are, in this case, the same as previous limits ($0.26 \leq K_m \leq 0.5$ and $5.9 \leq R_g \leq 13$). These intervals are then sent to the design agent controlling the optimization of the gradeability test. The results of this optimization are shown in figure 7-c. The new design variable limits are $0.298 \leq K_m \leq 0.404$ and $9.54 \leq R_g \leq 13$.
The new interval limits are sent back to the two first design agents to filter the results previously obtained. This gives the results shown in figure 7-d for the acceleration test-case and figure 7-e for the maximum velocity test-case.

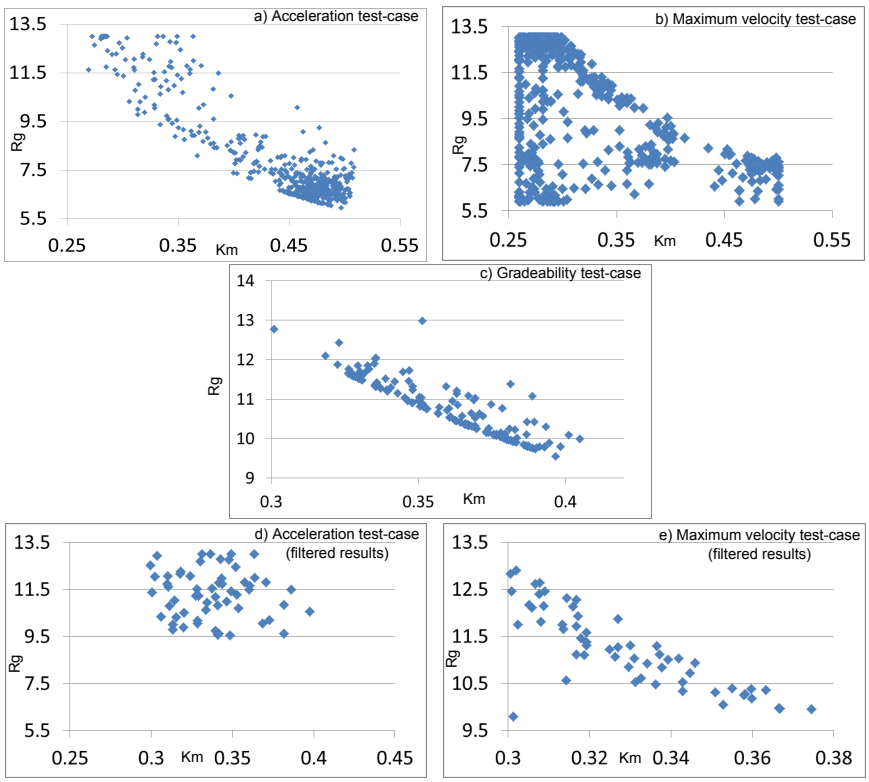

Fig. 7. Pareto fronts $[R_g - K_m]$ for three test cases: a) Acceleration; b) Maximum velocity; c) Gradeability; d) Acceleration with filtered results in $[0.298 \leq K_m \leq 0.404$ and $9.54 \leq R_g \leq 13\ ]$; e) Maximum velocity with filtered results in $[0.298 \leq K_m \leq 0.404$ and $9.54 \leq R_g \leq 13\ ]$.

The filtered optimization results are therefore sent to the coordinator. Figure 8-a shows the superposed optimization results. The coordinator searches for the common solutions belonging to the three sets of results. The obtained results constitute the optimal Pareto front of the overall design problem (figure 8-b).

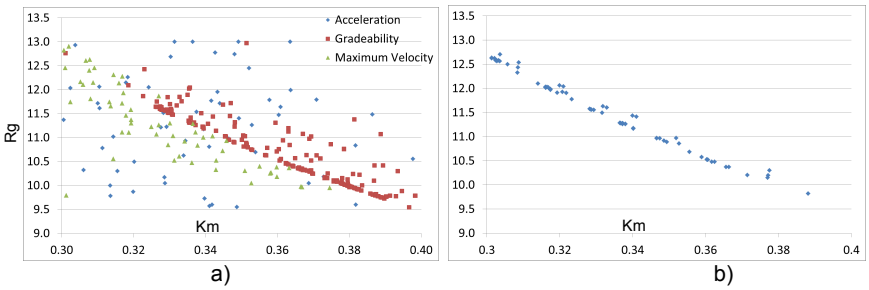

Fig. 8. Optimization results: a) Superposition of filtered results; b) final Pareto front

The obtained Pareto front shows an inverse-linear correlation between the gearbox ratio R_g and the motor constant K_m. The limit optimal solutions are $[K_m = 0.301,\ R_g = 12.62]$ and $[K_m = 0.388,\ R_g = 9.82]$. One intermediate optimal solution is $[K_m = 0.34,\ R_g = 11.16]$. This gives a large choice to designers to make decision about the electric machine and the gearbox. However, by adding other constraints such as cost, the choice could be further limited. These constraints can easily be added to the multi-agent platform by adding new design agents related to cost constraints.
The coordinating agent can also be programmed to help in making decision on the choice of solutions among the Pareto front by adding other design rules to filter the set of solutions found.

With this relatively simple example, we showed that it is possible to divide the complex optimization problem into partitions and perform an optimization of the overall system design by coordination using agents. However, the proposed approach can be more efficient for more complex design problems necessitating several design teams with different modeling software tools and multidisciplinary fields.

One novelty in this approach is the introduction of the specification of the optimization in the upstream design phase; the optimization itself is performed at the end. Conversely to conventional design methods that leaves the optimization phase at the end of the design process. This reduces the number of modifications and adaptations required on analysis models to succeed the optimizations.

5. Conclusion

Our main contribution in this paper is to propose a novel approach for the design partitioning and coordination of complex systems. The partitioning phase begins with defining the maximum number of partitioning possibilities by combining the elements of the optimization problem in terms of objective functions, design constraints, design variables and analysis models.Then, by using a partitioning metric, the designers select the best partitioning solutions. This step can be automated by formulating an optimization problem for the selection of optimal configurations by maximizing the value of the partitioning metric. After that, an agent-supported approach is proposed for the coordination of the design optimization. This approach has been implemented using JADE platform and applied to the case of a preliminary design of an electric vehicle.

The presented approach simplifies the complexity of the design process of complex systems and ameliorates the collaborative design. It also reduces the cost of the design of complex systems by distributing the work on several posts instead of using high performance computers. However, it should be mentioned that for simple design problems, the computing time of optimization problems could be longer with the agent approach compared to classic MDO method, because of the exchange of messages. Nevertheless, computing time could be reduced for very complex problems, using the capacity of agents to adapt the optimization process with the intermediate results found during the optimization process. Therefore, the approach should be reserved for complex systems having real collaborative distributed design constraints, to guarantee a good compromise between the integration required for coupled disciplines and decomposition required for the collaborative distributed design.

Acknowledgment

This work has been supported by the Austrian COMET-K2 programme of the Linz Center of Mechatronics (LCM), and was funded by the Austrian federal government and the federal state of Upper Austria.

References

1. H. A. Simon, *The sciences of the artificial.* MIT press, 1996, vol. 136.
2. E. J. Cramer, J. Dennis, Jr, P. D. Frank, R. M. Lewis, and G. R. Shubin, "Problem formulation for multidisciplinary optimization," *SIAM Journal on Optimization*, vol. 4, no. 4, pp. 754–776, 1994.
3. K. IM, "Development and application of the collaborative optimization architecture in a multidisciplinary design environment," 1995.
4. J. Sobieszczanski-Sobieski and R. T. Haftka, "Multidisciplinary aerospace design op-

timization: survey of recent developments," *Structural optimization*, vol. 14, no. 1, pp. 1–23, 1997.

5. C. D. McAllister, T. W. Simpson, K. Hacker, K. Lewis, and A. Messac, "Integrating linear physical programming within collaborative optimization for multiobjective multidisciplinary design optimization," *Structural and Multidisciplinary Optimization*, vol. 29, no. 3, pp. 178–189, 2005.
6. L. Li and J. Liu, "An efficient and flexible web services-based multidisciplinary design optimisation framework for complex engineering systems," *Enterprise Information Systems*, vol. 6, no. 3, pp. 345–371, 2012.
7. J. R. Martins and A. B. Lambe, "Multidisciplinary design optimization: a survey of architectures," *AIAA journal*, vol. 51, no. 9, pp. 2049–2075, 2013.
8. M. Luck, P. McBurney, and C. Preist, "A manifesto for agent technology: Towards next generation computing," *Autonomous Agents and Multi-Agent Systems*, vol. 9, no. 3, pp. 203–252, 2004.
9. J. Sobieszczanski-Sobieski, "Optimization by decomposition: a step from hierarchic to non-hierarchic systems," *NASA STI/Recon Technical Report N*, vol. 89, p. 25149, 1989.
10. R. Tappeta and J. Renaud, "Multiobjective collaborative optimization," *Journal of Mechanical Design*, vol. 119, no. 3, pp. 403–411, 1997.
11. N. Michelena, P. Papalambros, H. Park, and D. Kulkarni, "Hierarchical overlapping coordination for large-scale optimization by decomposition," *AIAA journal*, vol. 37, no. 7, pp. 890–896, 1999.
12. J. Sobieszczanski-Sobieski, J. S. Agte, and R. R. Sandusky, "Bilevel integrated system synthesis," *AIAA journal*, vol. 38, no. 1, pp. 164–172, 2000.
13. M. Hammadi, J.-Y. Choley, O. Penas, J. Louati, A. Rivière, and M. Haddar, "Layout optimization of power modules using a sequentially coupled approach." *International Journal of Simulation Modelling (IJSIMM)*, vol. 10, no. 3, 2011.
14. H. M. Kim, N. F. Michelena, P. Y. Papalambros, and T. Jiang, "Target cascading in optimal system design," *Journal of mechanical design*, vol. 125, no. 3, pp. 474–480, 2003.
15. H. S. Nwana, "Software agents: An overview," *The knowledge engineering review*, vol. 11, no. 03, pp. 205–244, 1996.
16. T. Ören and L. Yilmaz, "Synergies of simulation, agents, and systems engineering," *Expert Systems with Applications*, vol. 39, no. 1, pp. 81–88, 2012.
17. P. Leitão, "Agent-based distributed manufacturing control: A state-of-the-art survey," *Engineering Applications of Artificial Intelligence*, vol. 22, no. 7, pp. 979–991, 2009.
18. Y. D. Wang, W. Shen, and H. Ghenniwa, "Webblow: a web/agent-based multidisciplinary design optimization environment," *Computers in Industry*, vol. 52, no. 1, pp. 17 – 28, 2003, knowledge Sharing in Collaborative Design Environments.
19. Y.-I. Lin, Y.-W. Chou, J.-Y. Shiau, and C.-H. Chu, "Multi-agent negotiation based on price schedules algorithm for distributed collaborative design," *Journal of Intelligent Manufacturing*, vol. 24, no. 3, pp. 545–557, 2013.
20. M. Hammadi, J.-Y. Choley, and F. Mhenni, "A multi-agent methodology for multi-level modeling of mechatronic systems," *Advanced Engineering Informatics*, vol. 28, no. 3, pp. 208 – 217, 2014.
21. Z. Ren, F. Yang, N. Bouchlaghem, and C. Anumba, "Multi-disciplinary collaborative building designa comparative study between multi-agent systems and multi-disciplinary optimisation approaches," *Automation in Construction*, vol. 20, no. 5, pp. 537–549, 2011.
22. P. Fritzson, *Principles of object-oriented modeling and simulation with Modelica 2.1.*

John Wiley & Sons, 2010.

23. K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: Nsga-ii," *Evolutionary Computation, IEEE Transactions on*, vol. 6, no. 2, pp. 182–197, 2002.