# Knowledge management for collaborative design and multi-physical optimization of mechatronic systems

Mehdi Mcharek

université PARIS-SACLAY

NNT : 2018SACLC098

CentraleSupélec

# Gestion des connaissances pour la conception collaborative et l'optimisation multi-physique de systèmes mécatroniques

Thèse de doctorat de l'Université Paris-Saclay
Préparée à CentraleSupélec

École doctorale n°573 approches interdisciplinaires, fondements, applications et innovation (Interfaces)
Spécialité de doctorat: Sciences et technologies industrielles

Thèse présentée et soutenue à Saint Ouen, le 12/12/2018, par

**Mehdi Mcharek**

Composition du Jury :

| | |
|---|---|
| Jean Bigeon<br>Directeur de recherche à l'INP Grenoble | Président |
| Marc Budinger<br>Maître de conférences HDR à l'INSA Toulouse | Rapporteur |
| Nadine Piat<br>Professeur à l'ENSMM Besançon | Rapporteuse |
| Jean-Marc Faure<br>Professeur à Supméca Paris | Examinateur |
| Stanislao Patalano<br>Associate professor à l'unviversité Frederico II Naples | Examinateur |
| Jean-Yves Choley<br>Profeseur à Supméca Paris | Directeur de thèse |
| Cherif Larouci<br>Maître de conférences HDR à l'Estaca | Co-directeur de thèse |
| Moncef Hammadi<br>Maître de conférences à Supméca Paris | Invité |
| Toufik Azib<br>Maître de conférences à l'Estaca | Invité |

Thèse de doctorat

**Titre:** Gestion des connaissances pour la conception collaborative et l'optimisation multi-physique de systèmes mécatroniques

**Mots clés:** Gestion des connaissances, conception collaborative, processus de conception, optimisation multidisciplinaire, systèmes mécatroniques, boitier papillon

**Résumé:** Les produits mécatroniques sont complexes et multidisciplinaires par nature. Les exigences pour les concevoir sont souvent contradictoires et doivent être validées par les différentes équipes d'ingénierie disciplinaire (ID). Pour répondre à cette complexité et réduire le temps de conception, les ingénieurs disciplinaires ont besoin de collaborer dynamiquement, de résoudre les conflits interdisciplinaires et de réutiliser les connaissances de projets antérieurs. De plus, ils ont besoin de collaborer en permanence avec l'équipe d'ingénierie systèmes (IS) pour avoir un accès direct aux exigences et l'équipe d'optimisation multidisciplinaire (OMD) pour valider le système dans sa globalité.

Nous proposons d'utiliser des techniques de gestion des connaissances pour structurer les connaissances générées lors des activités de collaboration afin d'harmoniser le cycle de conception. Notre principale contribution est une approche d'unification qui explique comment IS, ID et OMD se complètent et peuvent être utilisés en synergie pour un cycle de conception intégré et continu. Notre méthodologie permet de centraliser les connaissances nécessaires à la collabora-tion et au suivi des exigences. Elle assure également la traçabilité des échanges entre les ingénieurs grâce à la théorie des graphes. Cette connaissance formalisée du processus de collaboration permet de défi-nir automatiquement un problème OMD.

**Title:** Knowledge management for collaborative design and multi-physical optimization of mechatronic systems

**Keywords:** Knowledge Management (KM), Collaborative design, Design process, Multidisciplinary Design Optimization (MDO), Mechatronic systems, Electronic Throttle Body

**Abstract:** Mechatronic products are complex and multidisciplinary in nature. The requirements to design them are often contradictory and must be validated by the various disciplinary engineering (DE) teams. To address this complexity and reduce design time, disciplinary engineers need to collaborate dynamically, resolve interdisciplinary conflicts, and reuse knowledge from previous projects. In addition, they need to work seamlessly with the Systems Engineering (SE) team to have direct access to requirements and the Multidisciplinary Design Optimization (MDO) team for global validation.

We propose to use Knowledge Management techniques to structure the knowledge generated during collaboration activities and harmonize the overall design cycle. Our primary contribution is a unification approach, elaborating how SE, DE, and MDO complement each-other and can be used in synergy for an integrated and continuous design cycle. Our methodology centralizes the product knowledge necessary for collaboration. It ensures traceability of the exchange between disciplinary en-gineers using graph theory. This formalized process knowledge facilitates MDO problem definition.

"*Knowledge is the life of the mind*" – Abû Bakr As-Siddîq

# Acknowledgements

I dedicate this work to my parents for their unconditional love and support.

I would like to thank my supervisors Moncef Hammadi and Toufik Azib. You always were present for me and this work could not exist without your contributions. I really enjoyed working with you and sharing with you these nice three years!

I address my thanks to my directors Jean-Yves Choley and Cherif Larouci for welcoming me and providing me the appropriate conditions. You have always encouraged and guided me throughout this adventure, thank you!

I would like to thank Nadine Piat and Marc Budinger for accepting to review my PhD manuscript, for the quality of their reports, and their interesting questions. I address my thanks to Jean Bigeon for presiding the defense, as well as Stanislao Patalano and Jean-Marc Faure to examine this thesis. I appreciated your comments and valuable advice. Special thanks to Suzanne Thuron for the organization.

I express my gratitude to the members of the MIMe project, for their motivation and the constructive meetings that have enriched my work. Special thanks to Simon Midrier, Antoine Navarro, Abdoulaye Sow, Antoine Munck, Matthieu Bricogne, Kevin Maquin, Harvey Rowson, Françoise Caron, and Joseph Aracic. It was really a pleasure to collaborate with you. The experimental validations, industrial requirements, beta-testing, interviews were crucial in this work and all the members have made their best to achieve these goals, thank you!

I am grateful to the members of Quartz Laboratory for all the moments that we shared together. Thanks to Christelle, Amel, Régis, Reda, Olivia, and Faida for your support, availability and kindness. I would like to thank Lionel who've left us this year and who delighted us with his smile, enthusiasm and helpfulness. I am also grateful to the members of Estaca'Lab for welcoming me and for the stimulating discussions. Thanks to Sandrine, Georges, Amine, Adriano, Riad, Vincent, Namamoudou and Benoit.

I would like to thank my friends for being present with me. Thank you Achraf for being the best roommate. Thanks to Adel, Ahmad, Issam, Nicolas, Anis, Sandrine, Nourhene, Yassine, Yousri for the energy that you brought to me.

I finish my acknowledgement by thinking sincerely all the members of my family.

# Contents

# List of Figures

# List of Tables

# Introduction

## A. General introduction

Recent advances in design methods promote the development of concurrent engineering to reduce the time and cost of the design cycle. This is particularly necessary for mechatronic design which is involving several disciplines. But companies have also to address associated challenges related to collaboration and reuse. For instance, designers from different disciplines need to collaborate instantaneously and access to the right information at the right moment (Maranzana, Gartiser et al. 2008).

Mechatronic systems encompass a variety of disciplines, including control, electrical and software. They need to be combined to accomplish the entire requisite functionality (Zheng, Bricogne et al. 2014). Each discipline independently focuses on a particular aspect of the system and exploits different Disciplinary Engineering (DE) tools for technical analysis. Such multiplicity of tools and methods renders the mechatronic design quite complex and knowledge intensive. Systems Engineering (SE) approach was proposed to manage this issue. It provides a common communication platform between different stakeholders at the system level, ensuring that each discipline meets the system requirements. Despite these efforts, a benchmark report about mechatronic design challenges shows that 44% of the manufacturers have a problem with understanding and fulfilling requirements (Jackson 2006). To attain the full benefit of SE in concurrent engineering, there is a critical need to create links between system engineers and disciplinary engineers. Research reports a gap between SE and DE, resulting in costly failures to meet system requirements (Gausemeier, Gaukstern et al. 2013). Solutions must support the collaboration between the two levels, allowing system engineers to manage changes in requirements and access dynamic decisions, made during the design cycle, and disciplinary engineers to solve their conflicting objectives and to stay consistent with the system level. Multidisciplinary Design Optimization (MDO) was proposed as a solution to this issue. It provides useful instruments and methodologies to deal with complex design problems. Nevertheless, MDO is challenging to implement in an industrial environment and cannot handle for itself the complexity of the dynamic collaboration and complex reuse of project results (Simpson, Toropov et al. 2008).

In our manuscript, we will study the links between SE, DE and MDO and how to merge them for an integrated and continuous mechatronic design cycle.

## B. Research context

### a. MIMe research project

This thesis is a part of a collaborative French project entitled FUI19-MIMe (Model d'Intégraton et de Simulation Mécatronique), which brings together industrials and academics:

- Small and medium-sized enterprises: DPS, Deltacad, Soyatec, and Eiris
- Manufacturers: PSA and Valeo
- Academics: Supmeca, Estaca, and UTC

The MIMe project aims to enable structured and instantaneous collaboration in mechatronic design.

PSA and Valeo are two major French automotive manufactures. Many meetings were conducted in the context of this project to identify the industrial requirements and the problems relative to mechatronic design. While each work package of the MIMe project focuses on specific objectives, the meetings were valuable to exchange our views and converge in our research.

b. Project objectives

The MIMe project is divided into 5 work packages. Our Ph.D. thesis is situated in the WP4. The different WPs meet regularly to keep the coherence of the global project.

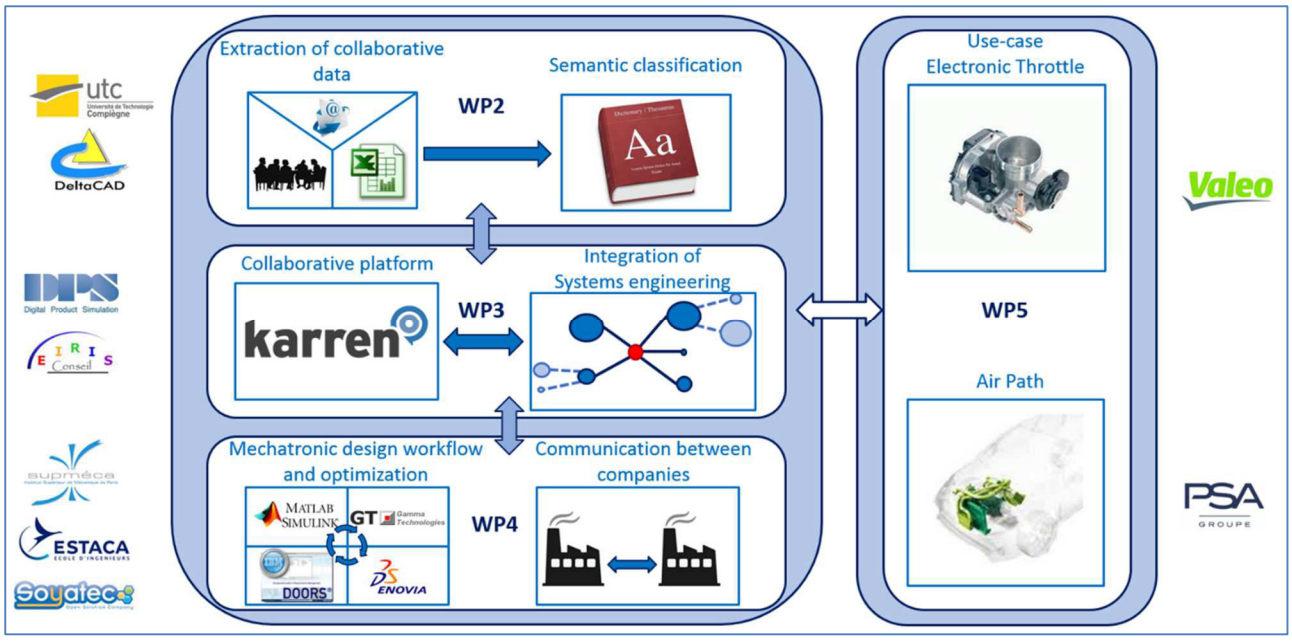



Figure 0-1. MIMe project organization

WP1: DPS

Provides technical, administrative and financial management of the project, as well as the development of a connector framework. This framework enables the automatic creation of connectors between CAD tools and the collaborative platform.

WP2: UTC - DeltaCAD

Determines how to capitalize multidisciplinary knowledge from exchanges made during collaboration to integrate them into a knowledge management structure. This concerns in particular two types of sources: the existing data (in the information systems of companies) and the data generated during the collaboration (emails, meetings..)

WP3: Eiris - DPS

Considers the capitalization of quantified requirements that are critical for tradeoff analysis. This is done by traceability between the system architecture and analysis models

WP4: Supmeca – Estaca - Soyatec

Our WP focuses on the mechatronic design workflow. Our objective is to formalize the collaborative design process and reuse this process efficiently.

WP5: PSA - Valeo

The objective of this WP is to present collaborative intra-company use-case (design teams) and inter-company use-case (suppliers, clients, and partners). The industrial use-cases will focus on the design process of the ETB (Valeo) and its integration into the air loop of a gasoline engine (PSA).

## C. Objectives and manuscript organization

### a. Research objectives

The discontinuity between Systems Engineering (SE), Disciplinary Engineering (DE) and Multidisciplinary Design Optimization (MDO) creates difficulties during the collaboration and the reuse phases. The originality of the proposal is to adopt a Knowledge Management approach bringing together knowledge from the different views, as simple as possible, in a single coherent framework with the following characteristics:

- Understanding the viewpoints of engineers in SE, DE , and MDO to collaborate efficiently
- Reusing efficiently results and decisions made in previous projects
- Automating repetitive design tasks when it is possible

The problematic and the expected contributions are detailed at the end of the Chapter 1.

### a. Outline of the manuscript

Following this introduction, the thesis manuscript is organized in 5 chapters (Figure 0-2)

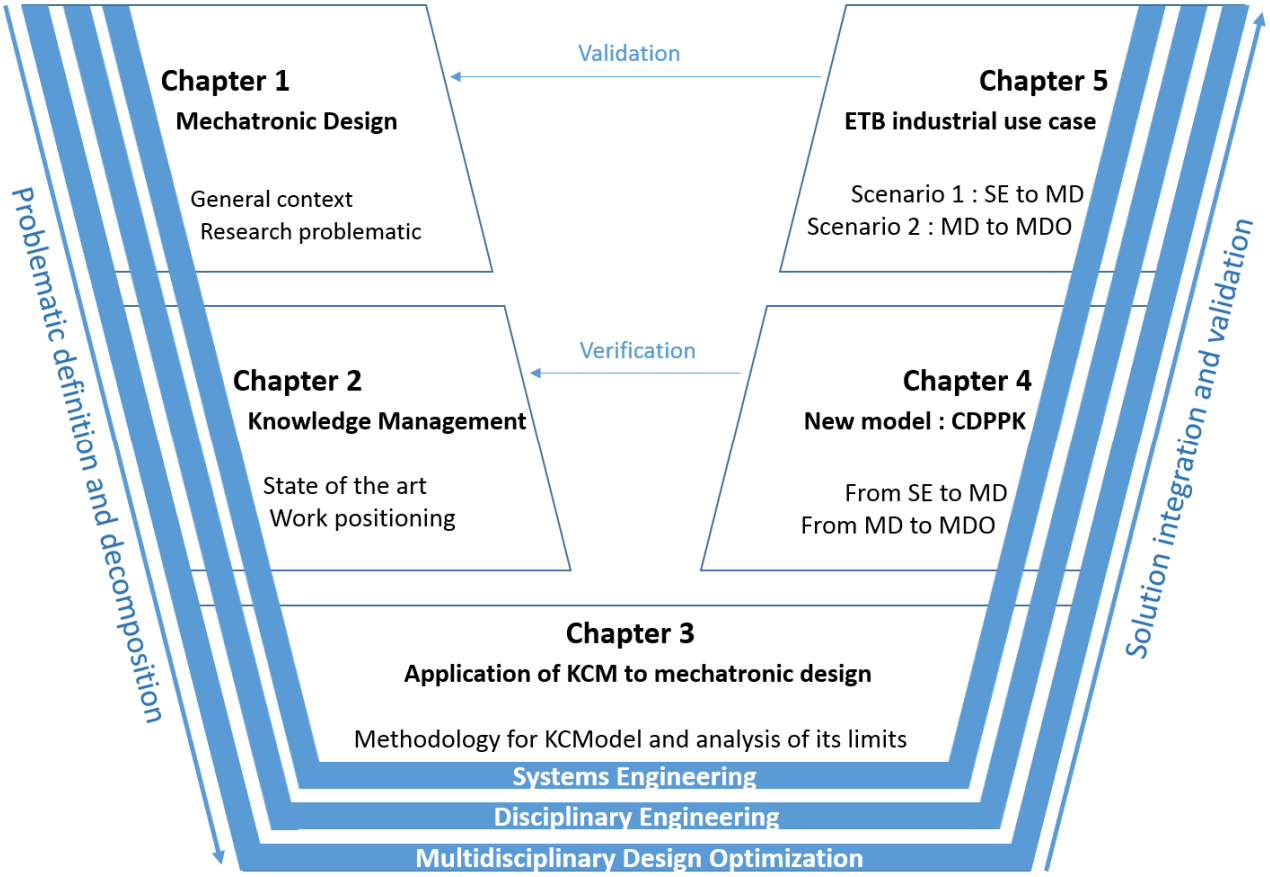



Figure 0-2. Organization of the manuscript chapters

- Chapter 1: This chapter introduces the mechatronic design and the difficulties that the designers encounter. We define SE, DE, and MDO with illustrative examples.
- Chapter 2: This chapter is a state of the art concerning existing solutions to support SE, DE, and MDO. Accordingly, our problematic and approach are positioned.
- Chapter 3: The Knowledge Configuration Model (KCM), outlined in chapter 2, is analyzed in detail in this chapter. We propose a methodology to use KCM in mechatronic design.
- Chapter 4: Based on the limits of KCM, our new model is presented in this chapter. Collaborative Design Process and Product Knowledge (CDPPK) model and associated methodology are explained to answer the research problematic.
- Chapter 5: Presents the validation of the proposal. We define in details the ETB design lifecycle in industry. The multidisciplinary models implemented and the experimental test bench are presented. The CDPPK is applied in two collaborative scenarios to validate our point.

Finally, we close this manuscript with a general conclusion. The first part is a summary of the conducted work with analysis and limits. The second part is about the perspectives and future work. Appendixes are also provided for more explanations.

# Abbreviations

| | |
|---|---|
| AAO: | All At Once |
| CAD: | Computer Aided Design |
| CAE: | Computer Aided Engineering |
| CDP: | Collaborative Design Process |
| CDPPK: | Collaborative Design Process and Product Knowledge |
| CE: | Concurrent Engineering |
| CFD: | Computational Fluid Dynamics |
| CM: | Configuration Management |
| CSCD: | Computer Supported Collaborative Design |
| DA: | Design Analysis |
| DE: | Disciplinary Engineering |
| DE: | Disciplinary Engineering |
| DoE: | Design of Experiment |
| DPK: | Design Product Knowledge |
| DV: | Design Variable |
| ETB: | Electronic Throttle Body |
| FEM: | Finite Element Method |
| ICE: | Information Core Entity |
| ISO: | International Organization for Standardization |
| IT: | Information Technology |
| KBE: | Knowledge Based Engineering |
| KBS: | Knowledge Based System |

KC: Knowledge Configuration

KCM: Knowledge Configuration Model

KM: Knowledge Management

LH: Limp Home

MDO: Multidisciplinary Design Optimization

PDM: Product Data Management

PIDO: Process Integration and Design Optimization

PLM: Product Lifecycle Management

SDM: Simulation Data Management

SE: Systems Engineering

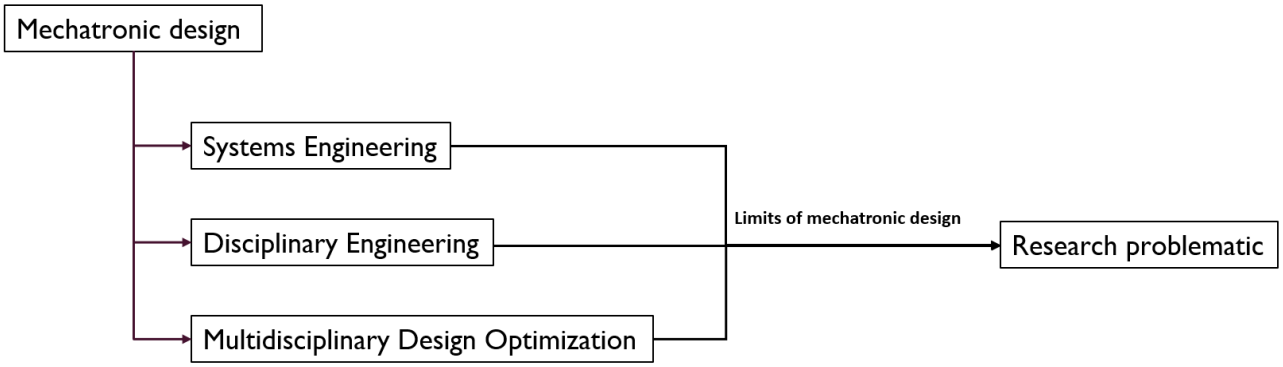
SysML: System Modeling Language

UML: Unified Modeling Language

UPC User Process Configuration

# Chapter 1 Mechatronic systems design

*In this chapter mechatronic design is introduced. We focus in this work on Systems Engineering (SE), Disciplinary Engineering (DE), and Multidisciplinary Design Optimization (MDO). Based on their limits, the research problematic emerges. The Electronic Throttle Body (ETB) is an example of mechatronic systems that will be used for illustration.*



## 1.1 Mechatronic systems

### 1.1.1 Definition of mechatronics

The term “mechatronics” originally emerged from the Yaskawa Electric Corporation in Japan (Kyura and Oho 1996). It combines the two words “mechanics” and “electronics”. The French Standard Organization AFNOR, normalized the definition of the mechatronics (NF E01-010):”a synergistic combination of mechanical, electrical, control and computer” as illustrated in Figure 1-1. The design of mechatronic systems is complex because of the increasing integration level and the wider range of collaborators involved (Tomizuka 2000). Mechatronics can be considered as a philosophical approach to design performant devices through a mechanism of simulating interdisciplinary ideas. The performance of mechatronic products results from the combination of precision mechanical and electrical engineering and real time programming integrated into the design process (Shetty, Manzione et al. 2012).

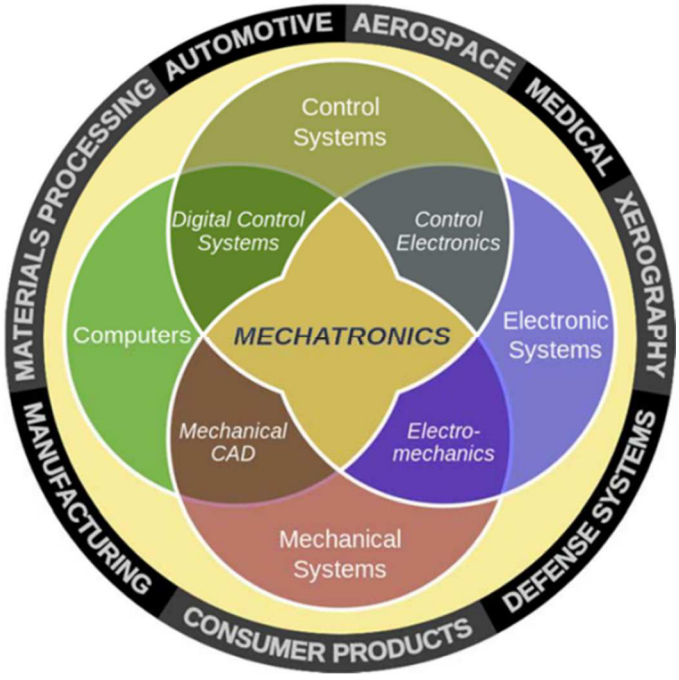



Figure 1-1. Mechatronic applications (Karnopp, Margolis et al. 2012)

### 1.1.2 Mechatronics system structure

There is a need to understand the fundamental working principles of mechatronic systems before approaching the design procedure of a mechatronic product. The general scheme (Figure 1-2) presents the architecture of a mechatronic system.

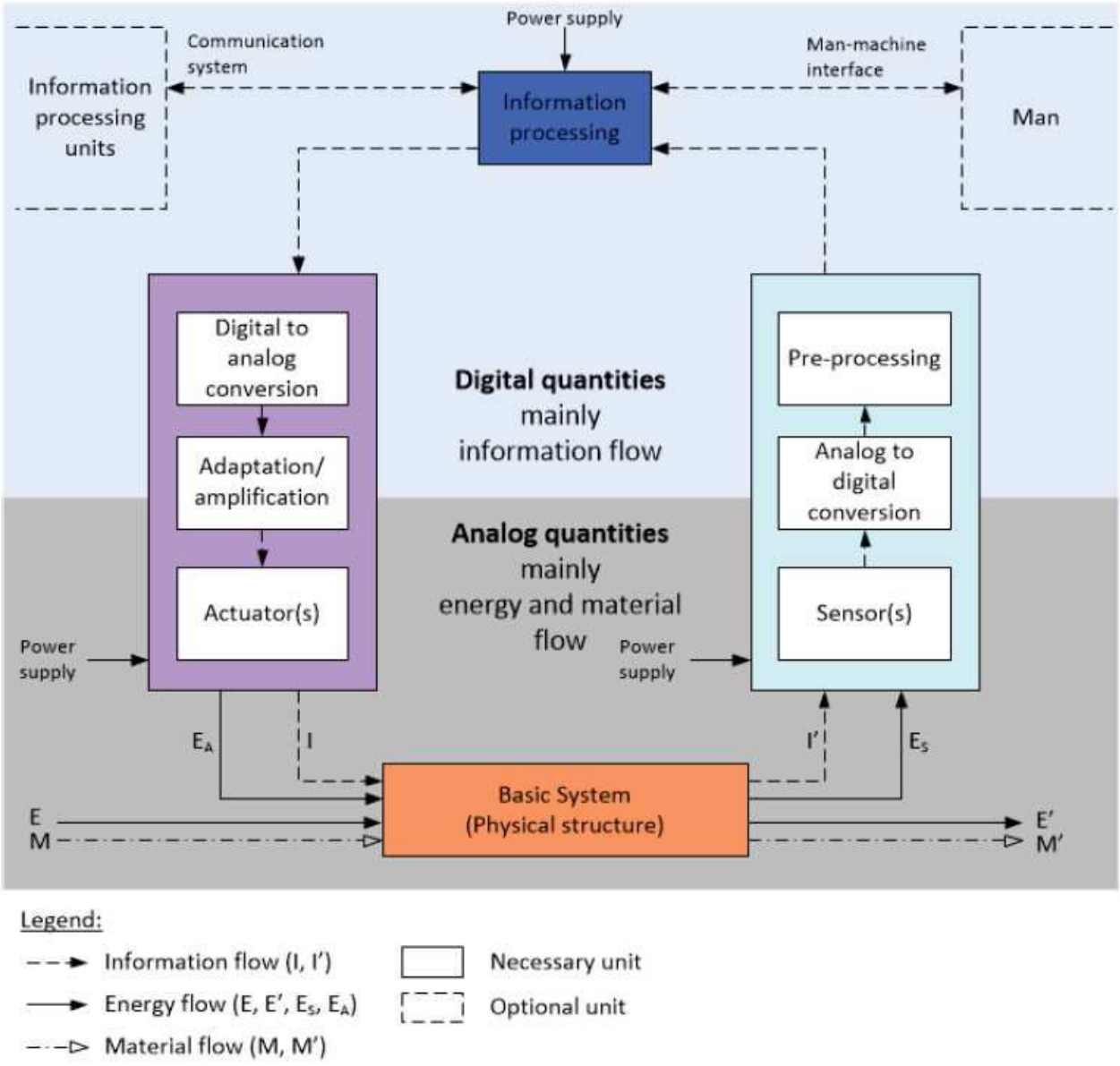



Figure 1-2. Mechatronic system architecture (adapted from (Krause, Jansen et al. 2007))

The basis of mechatronic systems is the physical structure. Information on the state of the mechanical product has to be obtained by measuring energy flow and/or information flow. Together with the reference variables (coming from man-machine interface), the measured variables are the inputs for an information processing, which controls the system. The generated signal is converted to an analog signal for the actuator. Mechatronic systems make new functions possible that could not be possible without this integration like fault diagnosis, adaptation to changes, autonomous systems…

In the future, growth in mechatronic systems will be fueled by the growth in the constituent domains. We can cite for example: cyber communication, 3D printing for integrated prototypes, images recognition in sensors, and machine learning in control systems. Mechatronic systems are present in many areas and some products will be presented in the next paragraph for illustration.

### 1.1.3 Mechatronic products

Mechatronic products are divided into micro-mechatronics and macro-mechatronics. Micro-mechatronics deals with miniaturization and precision applications like piezoelectric components, micro-actuators, and micro-sensors. Macro-mechatronics are shown in Figure 1-3, they are components present in daily products, machinery, transport…

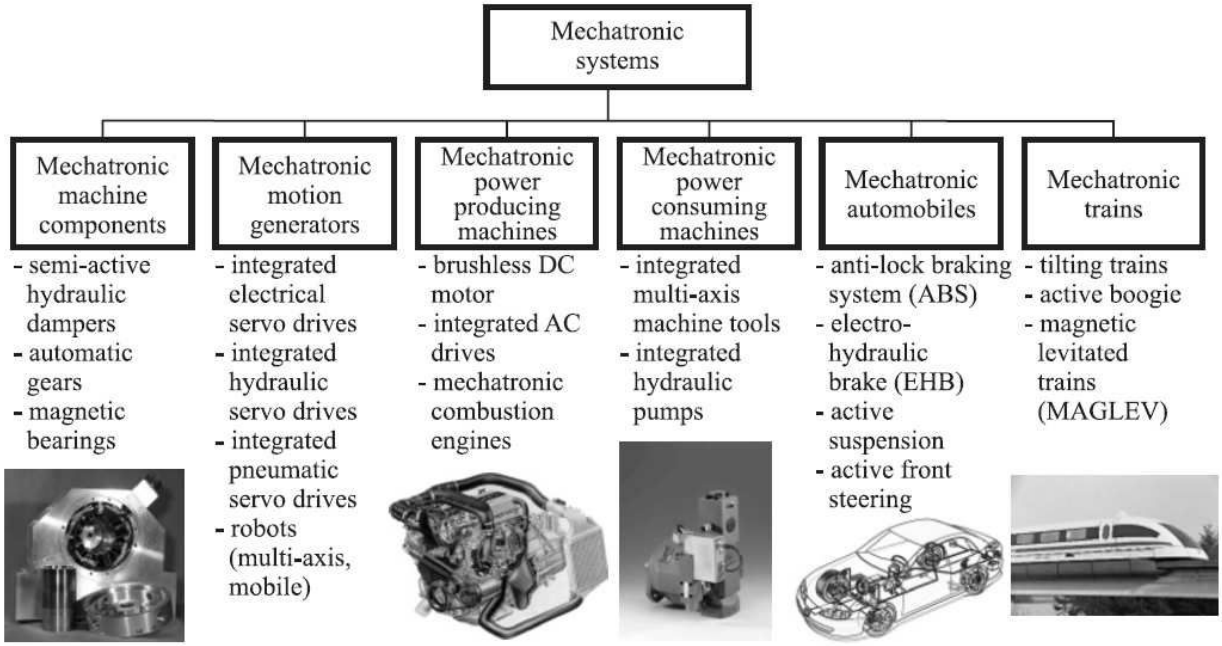



Figure 1-3. Examples for mechatronic components (Hehenberger and Zeman 2007)

Mechatronics has a high impact on the automotive area. In our project, PSA and Valeo gave as their point of view on this subject and how they collaborate to design mechatronic systems.

These systems improve the drivability and reduce the emissions. They introduced functionalities in automobiles that were not feasible with purely mechanical systems. All mechatronics management remains invisible for the customer and allows correction of systems in real time (engine management, stability control, braking control…). The modern Spark Ignition (SI) engines are equipped with mechatronic components that control air-to-fuel ratio (George and Pecht 2014). The process of varying air intake into the engine cylinder is accomplished employing the Electronic Throttle Body (ETB). As shown in Figure 1-4, the throttle valve varies the quantity of air flow into the engine and thereby cylinder charge, which determines the engine torque (Rossi, Tilli et al. 2000). The ETB offers in acceleration maneuvers, a smoother vehicle behavior and ensures security by controlling the engine operation range (engine speed limitation, idle speed...) (Corno, Tanelli et al. 2011).

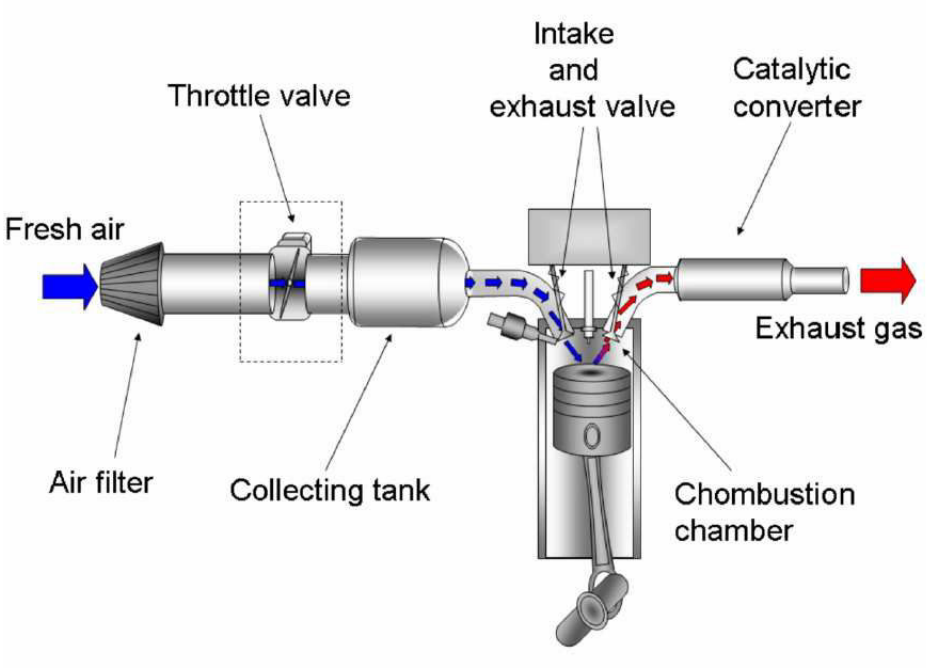



Figure 1-4. Electronic throttle body environment (Nentwig and Mercorelli 2008)

The ETB encompasses the aspects of a mechatronic system by containing mechanical parts, electric power, electronic sensor and a control system. Figure 1-5 details the composition of the ETB. A typical ETB includes a DC motor that actuates a gearbox and a valve. A potentiometer is generally used to measure the angle for control feedback. A failsafe system with two springs is used when the control system fails, it keeps the valve at the Limp Home position (between 10 deg and 14 deg) in order to provide the necessary flow to keep the engine running (Rossi, Tilli et al. 2000).

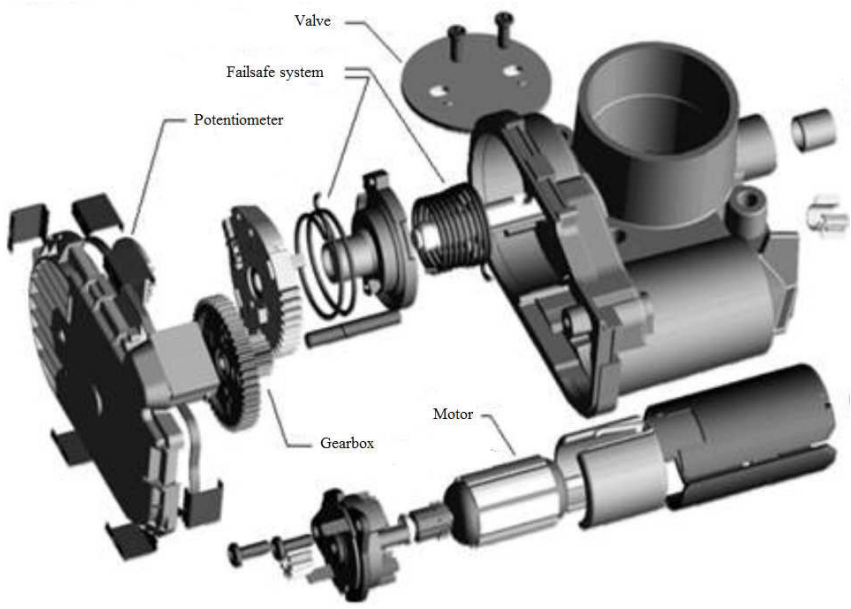



Figure 1-5. Electronic Throttle Body composition

As reported by PSA and Valeo, the design of such system creates new challenges for the industrials because of its multi-physical aspect and the implication of different design teams (control, mechanic, electric, thermal, and fluid). Moreover, the nonlinearities of the system bring challenges in the design and the verification phases. A successful design requires a successful collaboration between multi-disciplinary teams. The ETB example will be used in this manuscript in the different chapters with different levels of maturity to concretize the design process of such components and validate our approach. The next section details the design process of mechatronic systems.

## 1.2 Design process for mechatronic systems

The ABET (Accreditation Board for Engineering and Technology, Inc.) definition of engineering design: “Engineering design is the process of devising a system, component, or process to meet desired needs. It is a decision-making process (often iterative), in which the basic sciences, mathematics, and engineering sciences are applied to convert resources optimally to meet these stated needs (Commission 1999)”. Like any design, the mechatronic design is an iterative procedure with defined steps. However, it is more complex than mono-disciplinary processes because it requires continuous integration and collaboration. In this section, the main design cycles are analyzed.

### 1.2.1 Sequential cycle

It is not uncommon to find companies that consider that mechatronics system design as a sequential cycle (Figure 1-6). This cycle consists of three consecutive phases: modeling and simulation, prototyping and deployment. First, the requirements are analyzed and the conceptual design starts based on customer needs. The conceptual design aims to define the optimum configuration of the whole system without going into detail on its subsystems. The main functions are identified and an architecture is created for the system. Then, a first modular mathematical model is created with the fundamental behavior of the subsystems. After, a detailed model which is an extension of the first one provides more functions and accuracy for analysis. The control system is developed and the final step of this phase consists in the optimization of the system with its controller. If the requirements are designer can move the prototyping phase otherwise a new iteration is necessary to adjust the model.

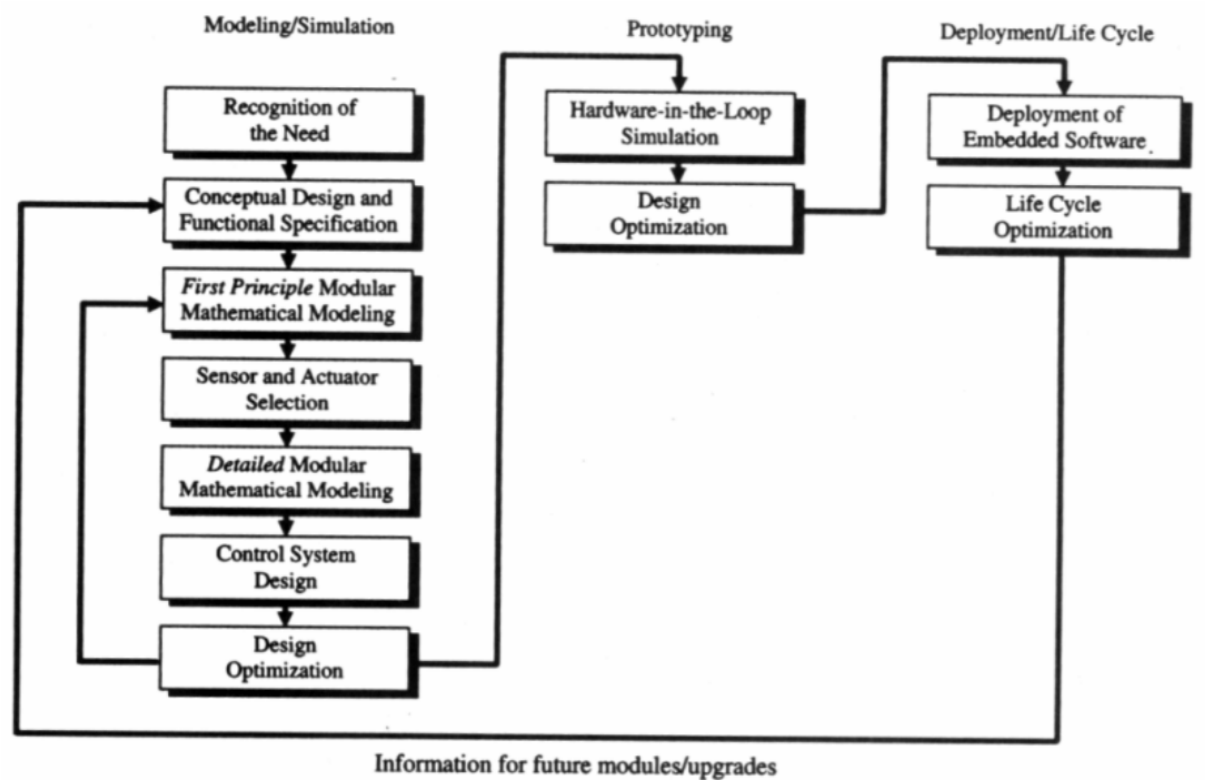



Figure 1-6. Mechatronic sequential process (Shetty, Manzione et al. 2012)

The complexity of multidisciplinary design showed the limits of this approach. Downstream tasks are penalized because they are more and more constraints by the choices made during upstream tasks. In fact, the multidisciplinary optimum cannot be reached by successive monodisciplinary calculations. Moreover, the duration of the process is not optimal because the tasks are carried out one after another. Therefore, V-model which is more adapted for concurrent engineering was widely adopted by industrials (Shetty and Kolk 2010). Then, it was standardized for mechatronics in VDI 2206 guideline that will be presented in the next paragraph.

### 1.2.2 VDI 2206 cycle

The VDI 2206 guideline provides a design process which is more integrated than the sequential design process. This cycle is divided into two phases: A top-down phase for system decomposition and a bottom-up for system integration (Figure 1-7). Even if the base of this process is sequential, the V-form highlights the continuous feedback connections between the two phases. The disciplines are treated in parallel to stress the iterations that occur during the cycle and facilitate concurrent engineering. Five main macro steps can be considered in this process (Dick, Hull et al. 2017):

Step1: Product Requirements

The customer requirements are identified in this step. A good understanding of customer requirements is the key to provide a suitable system (Wiesner, Freitag et al. 2015). Requirements should be clear and concise because this step is essential to have a first estimation of the resources and time allocation.

Step2: Conceptual Design

Based on the requirements, system engineers identify the different functions of the system. This provides a greater understanding of the system complexity. For each function, a functional subsystem is defined (Peluso 2015). Therefore, we can obtain different architectures depending on the chosen functional subsystems. Functional and architectural design can be combined into conceptual design (Kellner, Hehenberger et al. 2015). This step is important because it has a direct impact on the main parameters, properties, and cost.

Step3: Domain-specific Design

The detailed design step is between the two phases. The previously defined architecture is transformed into technical solutions with associating physical components to the functional subsystems. The architecture needs then to be modelled using embodiment tools. In this step, the system begins to take shape and first analyses are made to model the system (Törngren, Qamar et al. 2014).

Step4: Verification & Integration

Several analyses and simulations are made in this step to test the system against physical laws. The goal is to recompose the system and evaluate the integration of the different components (Petty 2009). The properties depending on the nature of the components (thermodynamic, vibration, dimensions...) and the requirements are verified and analyzed.

Step5: Product Validation

In this final step, the requirements related to the global system are validated. This validation can be experimental or virtual using analysis tools and multidisciplinary design optimization (Hammadi, Choley et al. 2012).

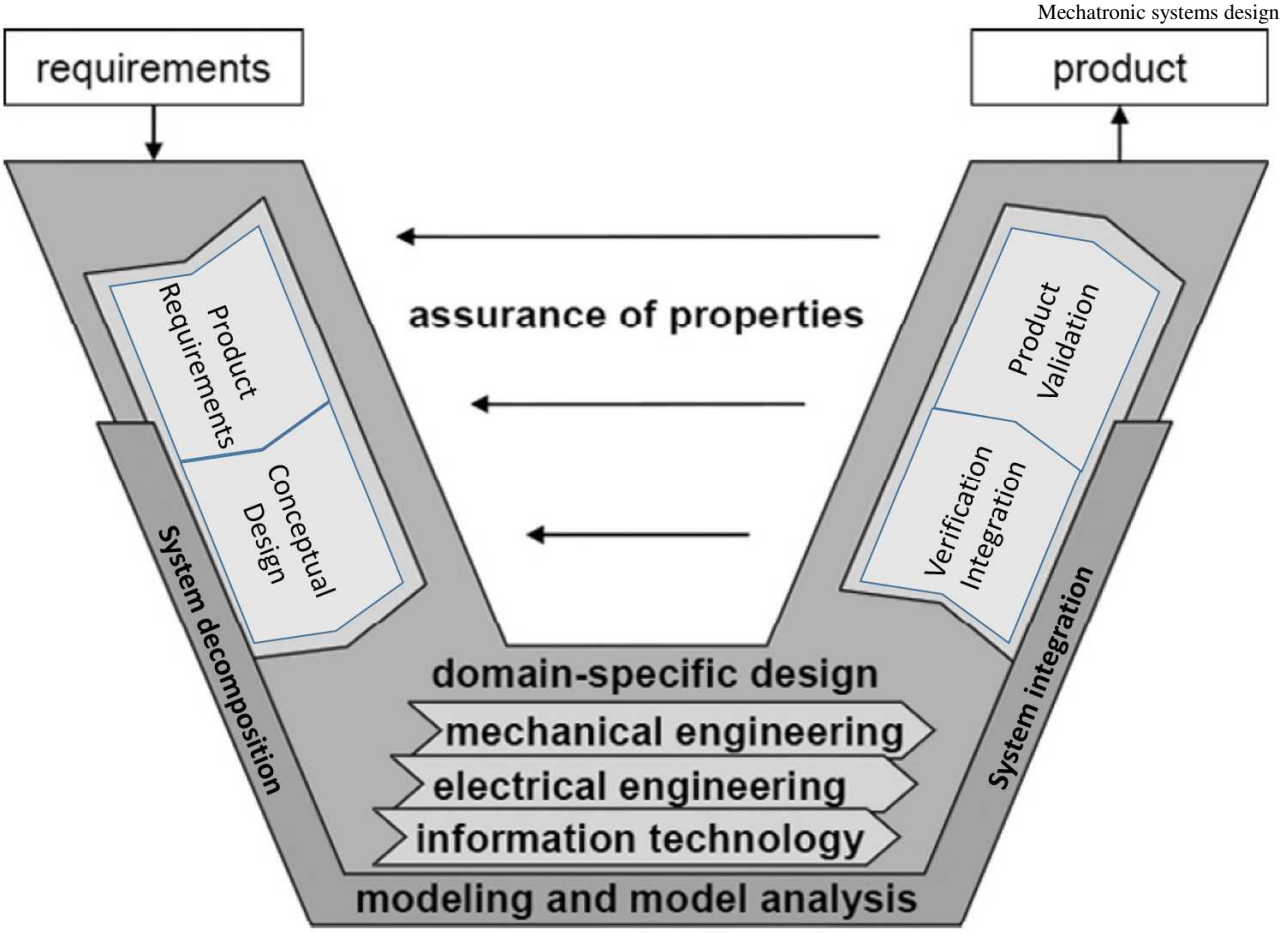



Figure 1-7. Design cycle of mechatronic systems (adapted from VDI Guideline)

### 1.2.3 Mechatronic design cycle in the industry

The VDI 2206 is, in practice, a spiraling process between decomposition phase and integration phase. We have initially incomplete requirements which gain completion during the decomposition phase. Also, conflicts in the integration phase lead to changes in the architecture defined by system engineers. The principle is to carry out in parallel different activities related to the product design. To study the mechatronic design cycle from an industrial perspective we decompose into SE, DE, and MDO (Figure 1-8). The predominant field in the first phase is SE which manages the decomposition activities (Requirements management, Functional modelling…). For the second phase, the field of DE is predominant. By DE we mean all the formal analyses and calculations made on CAD tools in the different disciplines involved in the mechatronic design (Embodiment, simulation…). Finally, MDO was recently created to resolve complex multidisciplinary problems through optimization algorithms. SE, DE, and MDO are explained in the next three sections.

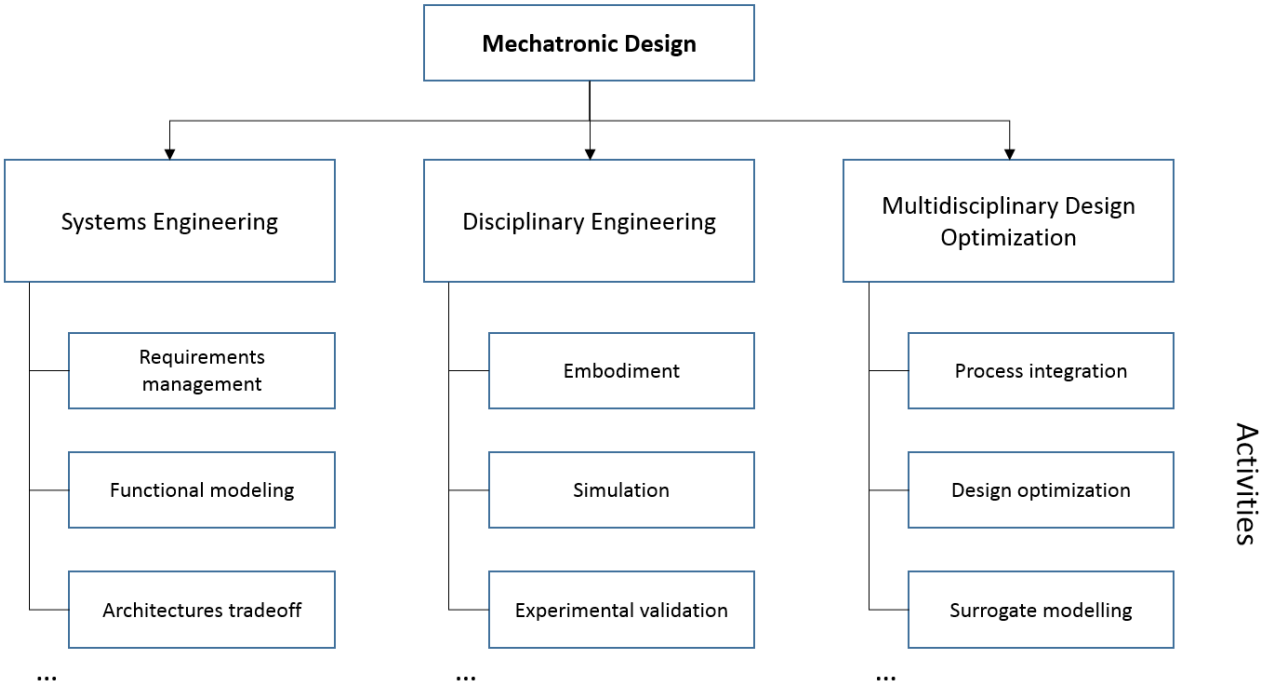



Figure 1-8. SE, DE, and MDO activities in mechatronic design

## 1.3 Systems Engineering (SE)

International Council on Systems Engineering (INCOSE) defines SE as "An interdisciplinary approach and means to enable the realization of successful systems. It focuses on defining customer needs and required functionality early in the development cycle, documenting requirements, then proceeding with design synthesis and system validation while considering the complete problem. It integrates all the disciplines and specialty groups into a team effort forming a structured development process that proceeds from concept over production to operation. SE considers both the business and the technical needs of all customers with the goal of providing a quality product that meets the user needs" (Friedenthal, Griego et al. 2007). In this section, we will explore this field and its implication in the definition and decomposition phase.

### 1.3.1 Definition and decomposition phase

SE was developed to address the challenges of coordinating and managing the design cycle. It is typically viewed as a separate branch of engineering addressing both technical and management activities with the goal of balancing all project objectives. The system engineer intervenes concretely in the first part of the design cycle to define the system based on the requirements (Lightsey 2001). Therefore tools belonging to SE are tools for creating a common system model that integrates all partial models and their relationships (Biahmou 2015). SE focuses more on the process of designing a system rather than on the solutions to the design problem itself. It focuses on the system-level to

evaluate the properties of the system under consideration in order to ensure that the final system meets the design requirements.

There are various languages for SE (SysML, eFFBD, ISM…). Only the SysML language will be considered in this chapter because it is one of the most commonly used in industry.

### 1.3.2 SysML Language

The System Modelling Language (SysML), comes from an effort from the International Council on Systems Engineering (InCoSE) and the Object Management Group (OMG) to integrate languages of different disciplines into one interdisciplinary language. SysML is the most widespread system modeling language in the industry and it derived from UML2.0 (Friedenthal, Moore et al. 2014). This language offers nine different diagram types representing various aspects of the system.

- **Requirement diagrams** are used to model the requirements, their organization and their relationships with the elements of the system.
- **Activity diagrams** allow modelling the chronological order of activities and decisions to model a process for example.
- **Sequence diagrams** are used to model the flow of control between actors and systems (blocks) or between parts of a system.
- **State machine diagrams** describe the states of a system and their changes in response to events.
- **Use case diagrams** describe the usage of a system by its actors (environment) and high-level services or features that the system has to offer.
- **Block definition diagrams** represent the structure of the system and provide an option for modelling the system hierarchy. The system consists of various blocks (modular units of the system), which are interconnected by connectors, which specify relationships between model elements, both within and across the boundary of the system.
- **Internal block diagrams** Represent the internal organization of a block and of its sub-blocks, and in particular the interconnections between the ports.
- **Parametric diagrams** represent physical aspects of the system, using constraint blocks in a specialized variant of an internal block diagram. Constraint blocks include a constraint (mathematical equation) and the associated parameters.
- **Package diagrams** are used to organize the model (e.g. for visualization and evaluation purposes)

There are allocations between these diagrams for traceability like the possibility to link a requirement and a component for example or a logical component and a physical component.

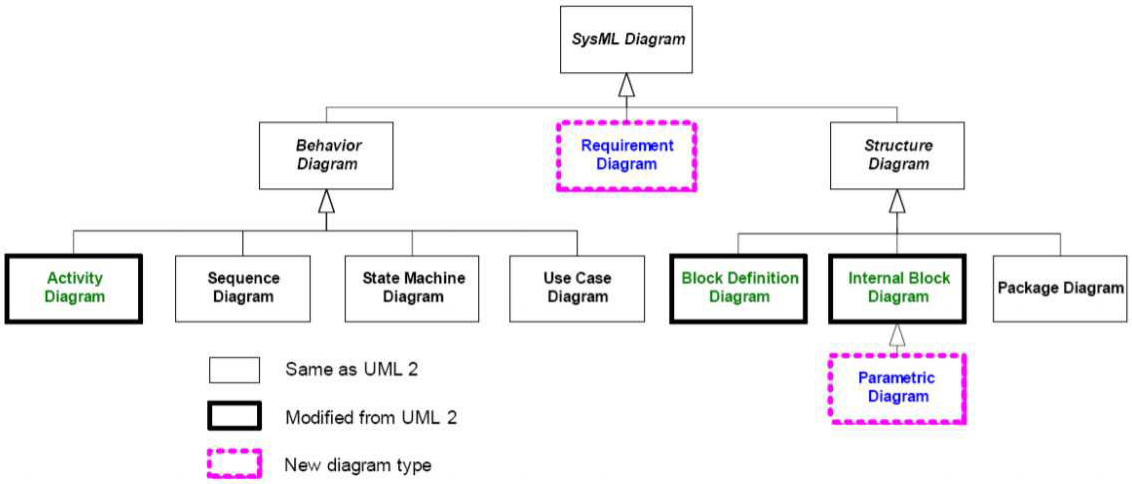



Figure 1-9. Diagrams of SysML language (Friedenthal, Moore et al. 2014)

The modelling of a system does not require the use of all the SysML diagrams, it depends on the type of study. To use this language efficiently in mechatronic design, we need methods to use the right diagram at the right moment.

### 1.3.3 Methods for SE in mechatronic design

As explained before, SysML language contains several diagrams that are linked to each other. A methodology is then needed to use them efficiently during the design cycle of mechatronic systems. "Black box" and "White box" approach was proposed for this purpose (Mhenni, Choley et al. 2014). In the first phase (Black box), the system is specified but not described by keeping an external point of view. The second phase (White box), the system is described in detail which corresponds to the conceptual phase. This methodology improves the traceability in the decomposition phase and promotes the diagrams reuse.

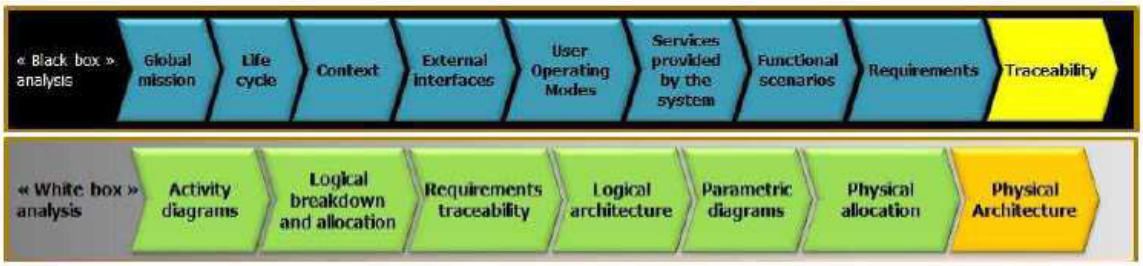



Figure 1-10. Black box-white box approach (Mhenni, Choley et al. 2014)

An integration modelling approach was also proposed in three steps (Abid, Pernelle et al. 2015). The first step is to define the system requirements with the external environment, main functions and link it to the PLM system. The second step is to study in detail the structure of the system with its subsystems and components. Finally, the behavioral study of the system allows defining the development process of the mechatronic system and its life cycle from PLM data (Figure 1-11)

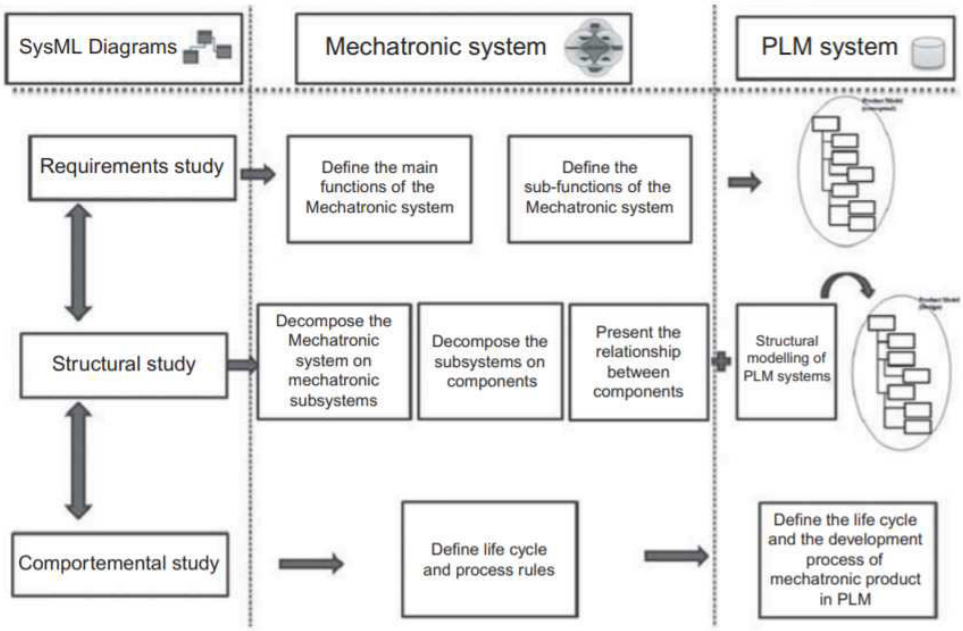



Figure 1-11. Integration modeling approach (Abid, Pernelle et al. 2015)

### 1.3.4 ETB definition example

To illustrate the SE approach. Two diagrams of the ETB system will be presented here. The functional diagram as explained before shows the different functions of the system and the links between them (Figure 1-12). This diagram shows also the input and output of the system and its environment.

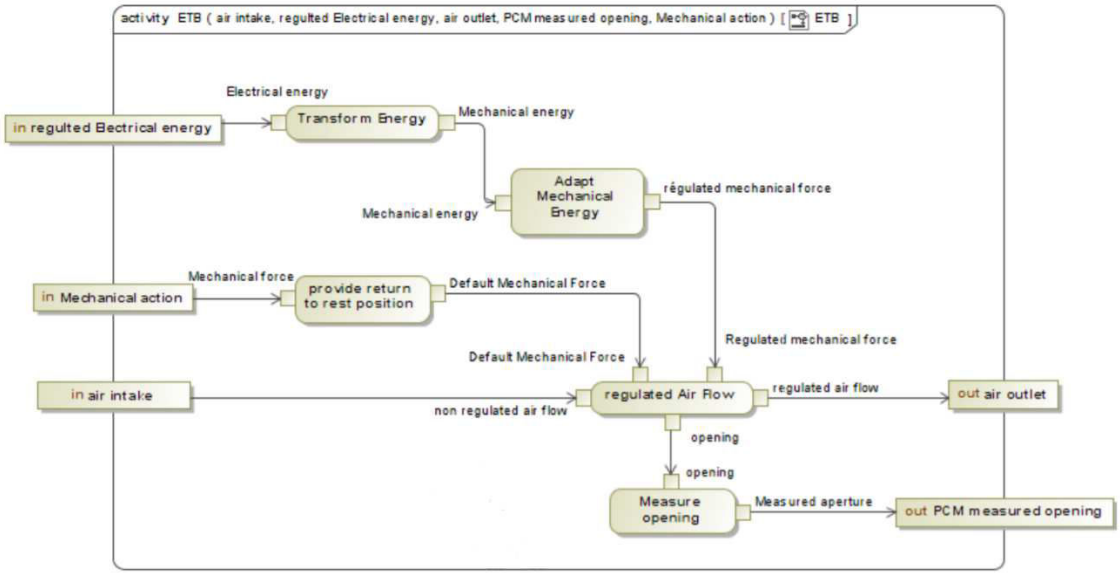



Figure 1-12. Functional architecture of the ETB (Ammar, Hammadi et al. 2017)

Different architectures can be considered for this functional architecture as illustrated in Figure 1-13. In fact, transmission and actuation function can be achieved by different technological solutions and this leads to a completely different product in terms of performance and cost.

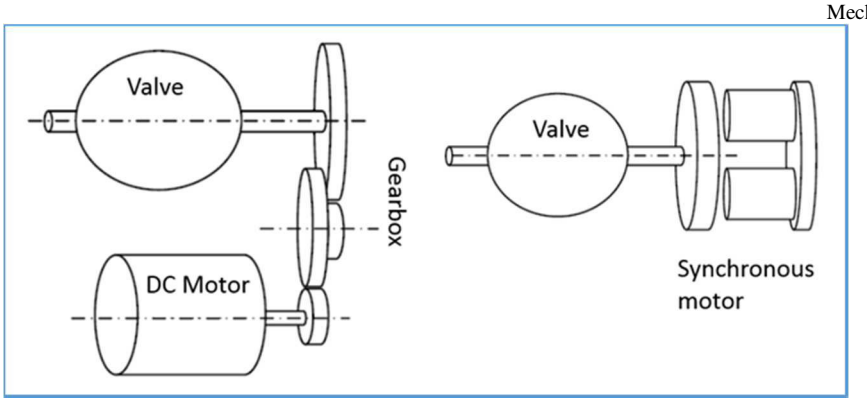



Figure 1-13. Two architectures of the ETB

In this manuscript, we will work with the standard architecture using a DC motor and a gearbox as shown in the BDD (Figure 1-14).

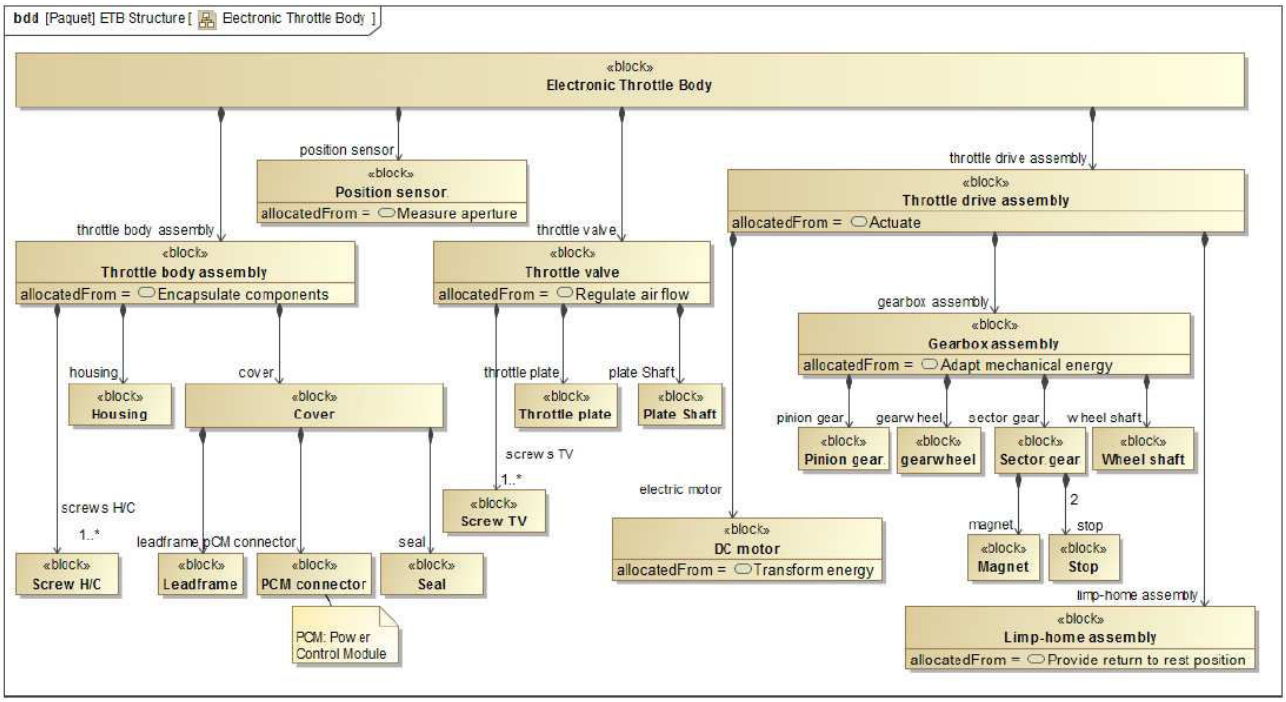



Figure 1-14. BDD of the standard ETB architecture

The decomposition of the system exposed in this paragraph is one of the main goals of the SE methodology. It gives a system level view of the product and organizes it in clear diagrams so that the disciplinary designers can refer to them. The next paragraph explains the current challenges of SE in industry.

### 1.3.5 SE challenges

These diagrams are only used to communicate between stakeholders and do not allow to make calculations or to be simulated. The integration of simulation and analysis capabilities within SE is one of the most challenging and promising research activities. Such interest is also highlighted by the recent creation of “Systems Modeling & Simulation Working Group” (SMSWG) between the “International

Council on Systems Engineering" (INCOSE) and the "International Association of the Engineering Modelling, Analysis and Simulation Community" (NAFEMS). This recent cooperation highlights the fact that the integration between system level and multidisciplinary level environments is currently one of the most investigated areas of research (Cencetti 2016). The current lack of integration between them generates several problems during concurrent engineering. Iterations between system level and detailed level are difficult and design changes (which are more and more present) cannot be solved easily by disciplinary designers. The next section explains more this issue from the DE point of view.

## 1.4 Disciplinary Engineering (DE)

### 1.4.1 Integration and verification phase

DE is a technical and detailed work to create models and simulate the mechatronic system using mathematical and physical laws. This rigorous approach aims to optimize the system in its entirety and to generate all the necessary data to produce the mechatronic system. Design teams are brought to work together and to develop new ways of working to deal with conflicts that emerge during the integration. In each discipline, designers are working with specific tools and methods but they have to merge their work to verify the global system functionalities. Therefore, the main disciplines (mechanic, electronic and control) are integrated depending on design maturity. This integration can be virtual by combining disciplinary tools or real by testing the control system with its structure (Figure 1-15). The main integration approaches in industry are:

- Model in the Loop (MiL): the plant model is connected to the controller model
- Software in the Loop (SiL): the plant model is connected to the controller translated C-code
- Hardware in the Loop (HiL): the plant model interacts with the controller hardware
- Rapid Control Prototype (RCP): where the real plant is operated together with the simulated controller

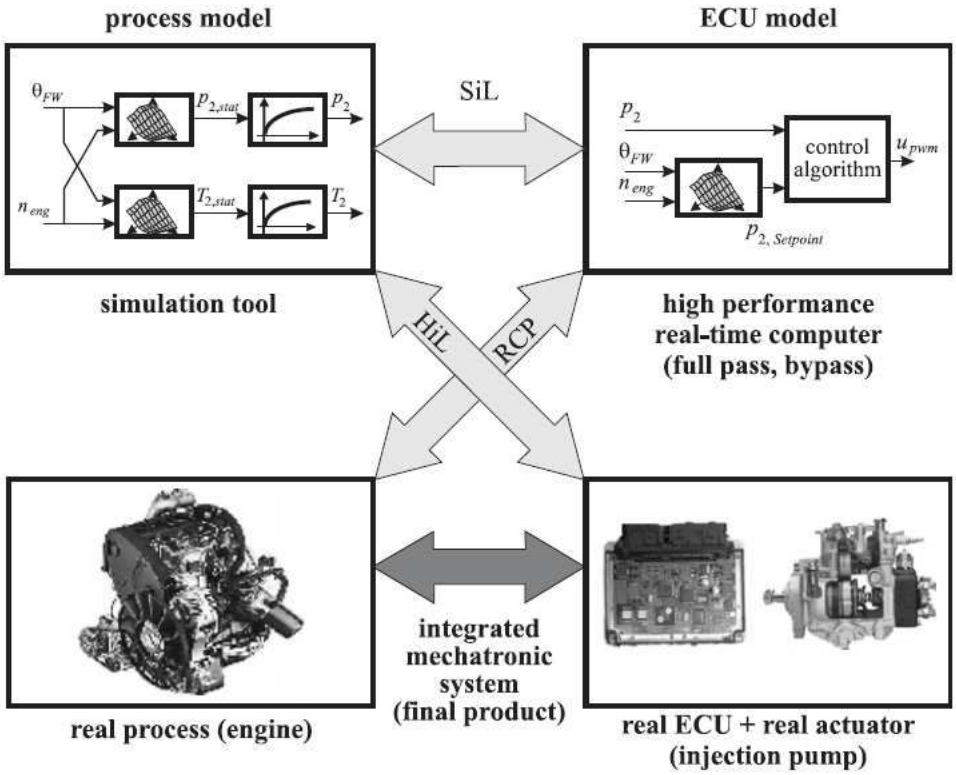



Figure 1-15. Combining real and simulated parts in the mechatronic design (Isermann and Müller 2003)

The system integration comprises the spatial integration of the hardware components (actuator, transmission, cables, ECU…) and functional integration of software algorithms (control system, fault diagnostic, HMI...). To sum up, integration in mechatronics means integration across traditional boundaries. In the following, we detail the tools used in this phase.

### 1.4.2 DE tools

The tool landscape for developing mechatronic systems is large and diverse, consisting of a number of multi-domain tools and mono-domain tools. In one hand, multidomain tools take into account the functional and physical coupling between components. This level of modelling is usually done using 0D/1D models, represented by algebraic equations, ordinary differential equations (ODE) or differential algebraic equations (DAE). These are some examples: Dymola, Open-modelica, Mobile VHDL-AMS, AMESIM, 20 SIM. On the other hand, Monodisciplinary tools are based on a 3D geometric representation. The monodomain phenomena are generally represented through partial differential equations (PDE) and by using numerical approximations such as finite element methods (FEM). Here are some Mono-domain DE tools:

• In mechanical design, dimensions, shapes, and materials that correspond to the physical objects are the main interest. These are some examples: Solid Edge, SolidWorks, Catia.ProE

• In control design, software based on transfer function are used like Matlab/Simulink.

• Electronics deals with the physical implementation of the controller. The software packages for electronic design support predictions of behavior and execution time through logical and physical simulations.

• Electrical engineering commonly designs components to link electronic and mechanical domains like Synopsys, OrCad

• For specific physical phenomena, like thermal, fluid and magnetic effects, FEM tools based on mesh calculations are used. We can cite: Ansys, Abaqus, SiemensNX

• Test benches: Because no single model can ever flawlessly reproduce reality, there will always be an error between the behavior of a product model and the actual products. A test bench offers a real environment to verify the product in various conditions.

In this context, Table 1-1 summarizes the DE tools used in an automotive project.

Table 1-1. Example of stakeholders and their roles in the automotive domain (Navet and Simonot-Lion 2008)

| **Stakeholder/role** | **Concerns** | **Analysis/synthesis** | **Model characteristics** |
|---|---|---|---|
| Electrical engineer/ hardware architect | ECU interfaces, EMC | Electrical load, tests | Logic, continuous, E-CAD, FEM |
| Software engineer (body area) | Logics of functionality | Simulation of behavior | Discrete-event |
| Control system engineer | Performance, robustness, disturbance, sensor noise | Behavior simulation, formal robustness analy-sis, controller synthesis | Continuous-time, discrete-time, discrete-event |
| Mechanical engineer | ECU packaging, geom., mtrl., fitting, sealing | Cable length, geometry, alignment, material sel. | 3D M-CAD, material DB, BOM, FEM |
| Thermal analyst | Temperature | Heat transfer, cooling | FEM, CFD |
| Purchaser | Comp. standardization | Life-cycle cost | PDM |
| Quality engineer | Reliability | Life-time prediction | Stochastic, logic |
| Service engineer | Serviceability, repair | Replacement, spare-parts | Lightweight CAD, PDM |
| Cost controller | Product cost | Profitability, sensitivity | Economical, uncertainty |
| Integration engineer | Verification, communi-cation, distr. functions | Testing, test automation & generation of test docu. | Discrete-event, test cases, logical structures |
| Safety engineer | System safety | FTA, FMEA | Logic, discrete-event, stochastics |

Mono-domain tools perform well within their domains but cannot consider all the aspects of the mechatronic system. Multi-domain tools can cover different aspects of the mechatronic system but they are not accurate enough to perform the various steps of the design cycle and need enrichment from more accurate tools. Therefore, interoperability is necessary between these tools to realize the integration and the validation of the mechatronic system.

### 1.4.3 DE and interoperability

A multidisciplinary system is a source of complexity because designers from different teams struggle to communicate and find agreement on design decisions, each with their business constraints and conflicting objectives. Moreover, if a significant number of tools is proposed for multi-domain design mechatronic systems, these do not integrate all the stages of the design cycle (Cabrera, Foeken et al. 2010).

One possible solution to support communication is through data exchange standards. For instance, within the CAD community, a number of international standards have been developed to make product data exchange possible. Among these standards, the STEP standard (Pratt 2001) is the most renowned. Some key issues, such as errors in exchanged data and loss of data when using a standardized neutral model, are still not resolved (Gielingh 2008). As a result, organizations tend to rely on tools from a single vendor (which can be costly), or stick to documents and meetings for data exchange (which can be time-consuming).

Another solution is the Functional Mock-up Interface (FMI) which is an open standardized interface for co-simulation and model exchange between simulation tools (Nouidui, Wetter et al. 2014). The purpose of the standard is to allow models to be interchanged between different vendors, departments or modelling disciplines. Each sub-part can then be modelled in the most appropriate tool.

Finally, XML is a well-known exchange standard. The goal of XML is to enable the automated exchange of content between heterogeneous information systems and thus to answer an interoperability problem. An XML document is a self-describing text file with tree-structured data. It is usable by any application but cannot support dynamic collaboration.

### 1.4.4 ETB open loop example

A simple example is presented in this paragraph to show some tools used in mechatronic multidisciplinary design. The task is to improve an existing ETB (Bosch). The first model is an embodiment model in Catia (Figure 1-16).

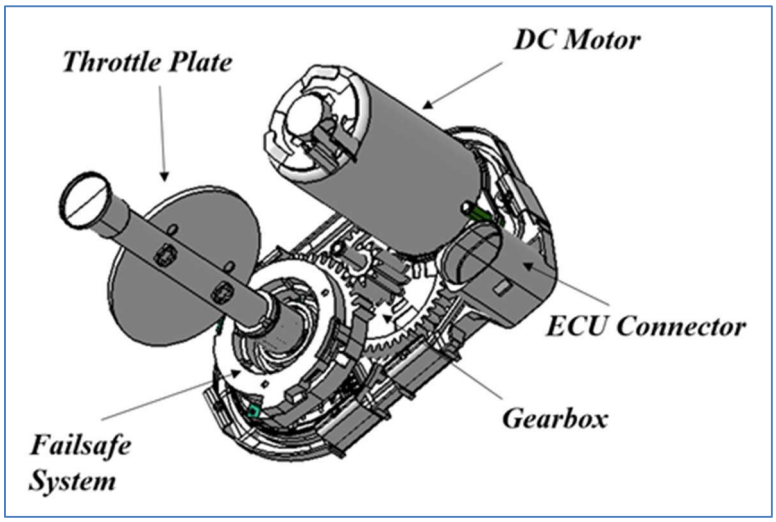



Figure 1-16. 3D model of the ETB

The architecture of the ETB and the involved parameters are detailed in Figure 1-17. The inertia of the valve, the gearbox as well as the gearbox ratio are calculated with the embodiment model to be used in the 0D model. The 0D model of the ETB is created using components from Modelica library (Figure 1-18). Initial values are given for the unknown parameters like the stiffness of the springs or the motor parameters. The list of the initial values is given in Table 1-1.Our goal is to understand the behavior of the system in open loop.

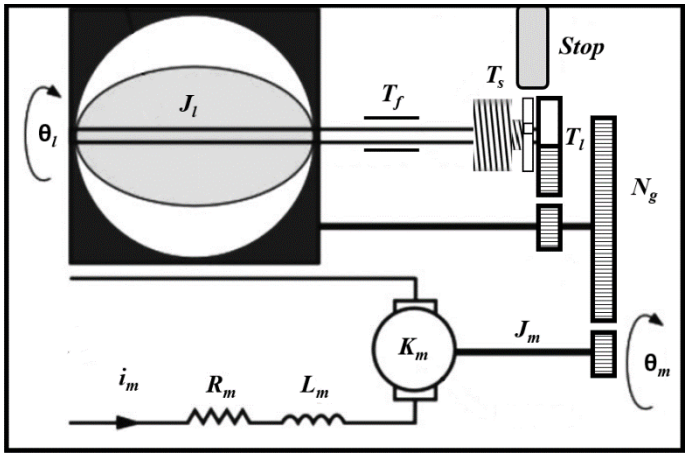



Figure 1-17. ETB open loop architecture and parameters

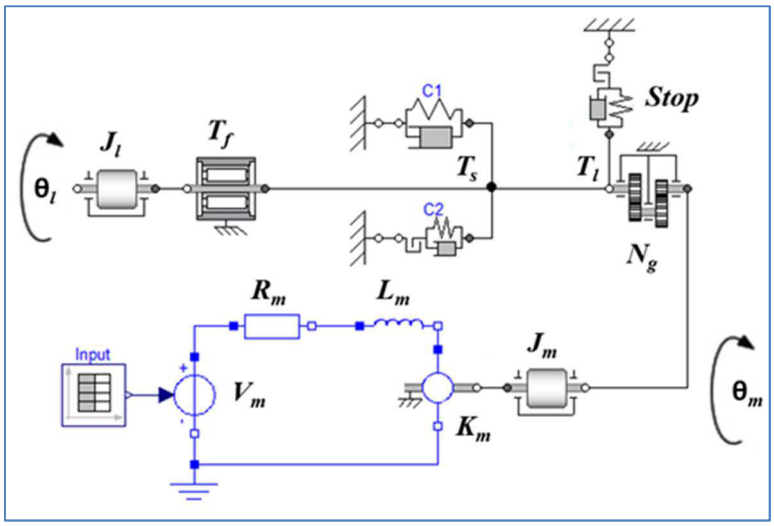



Figure 1-18. Modelica model of ETB in open loop

Table 1-2. ETB initial parameters

| Parameter | Unit | Description | Initial value |
|---|---|---|---|
| Motor | | | |
| $K_m$ | N.m/A | Motor constant | 0.02 |
| $K_e$ | N.m.s/rad | Back emf | 0.02 |
| $R_m$ | Ohm | Resistance | 1.5 |
| $L_m$ | H | Inductance | 0.0015 |
| $J_m$ | Kg.m² | Inertia | 4.8e-6 |
| $i_m$ | A | Current | variable |
| $V_m$ | V | Voltage | variable |
| $\theta_m$ | deg | Motor angle | variable |
| Gear | | | |
| $N_g$ | - | Ratio | 20 |
| Stop | deg | Angle interval | [0,90] |
| Load | | | |
| $J_l$ | Kg.m² | Inertia | 5.6e-5 |

| | | | |
|---|---|---|---|
| $T_s$ | N.m | Springs torque | variable |
| $T_f$ | N.m | Friction torque | variable |
| $T_l$ | N.m | Torque after reduction | variable |
| $C_1$ | N.m/rad | Main spring stiffness | 0.3 |
| $C_2$ | N.m/rad | Limp home spring stiffness | 0.6 |
| $\theta_l$ | deg | Throttle angle | variable |

A simulation is made to understand the failsafe functionality. The failsafe system works when the control system of the ETB fails. It brings the valve to a pre-defined angle (limp home position), in order to provide the necessary air flow to keep the engine running. This position is between 10 deg and 15 deg depending on models. The limp home spring is connected to the lever and the sector gear, it works only when the lever is in contact with the stop limit. The main spring is connected to the sector gear and the mass. The role of the main spring (stiffness C1) is to bring the valve from open positon to limp home position. And the role of the second spring (stiffness C2) is to bring the valve from the closed position to the limp home position. In the two cases the equilibrium position is reached at the limp home position. The Figure 1-19 shows an example done with a throttle initially positioned at 90° with 0V supply voltage. The two springs reach the equilibrium position at 10deg.

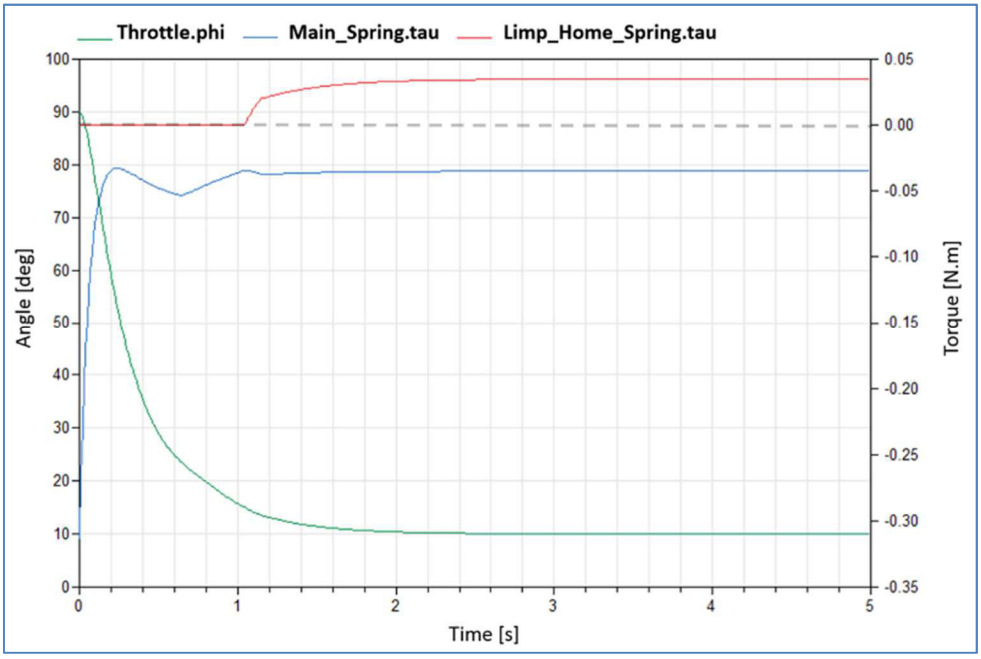



Figure 1-19. System response with 90° as an initial position to illustrate springs effects

This model gives us a first idea about the behavior of our system. We can understand how friction and springs work. However, this model needs optimization because the return time is too long. Optimizing such a model is difficult because we have conflicting parameters, if we reduce the main spring stiffness, the rising time will be shorter but the return time will be longer. Therefore, we need to use specific optimization methods. This challenge and others will be explained in the next paragraph.

### 1.4.5 DE challenges

Given the multidisciplinary nature of mechatronic design, engineers need diverse and heterogeneous DE tools. To integrate their work, engineers need communication means between these tools:

- The current interoperability solutions partially solve this need. They do not support the dynamic exchange in concurrent engineering (Van 2006).
- The network-based Workflow Management (WfM) have until now been more used for managing business processes, documents flow, and much less engineering process (van der Aalst 2012).
- The Computer Supported Collaborative Work (CSCW) tools focus on communication features (messaging) and co-ordination (approval forms, workflow tools, videoconference tools) but few of them are interested in collaboration among actors (Pawlak 2010).

Besides, current engineering problems are increasingly characterized by a wide set of conflicting objectives that must be properly approached. The MDO was proposed for this purpose.

## 1.5 Multidisciplinary Design Optimization (MDO)

Multidisciplinary Design Optimization is “a methodology for a design of complex engineering systems that are governed by mutually interacting physical phenomena and made up of distinct interacting subsystems (suitable for systems for which) in their design, everything influences everything else” (Sobieszczanski-Sobieski 1995). This methodology and its impact on mechatronic design are explained in this section.

### 1.5.1 Problem formulation

In mechatronic design, problems are non-linear, have many constraints and require minimization of one or more criteria. Optimization algorithms have been developed to help the design team in this quest. The optimization is an important tool in decision science and in the analysis of physical systems. To use this methodology, we must first identify the objectives to optimize and the constraints to respect. They depend on specific parameters, called Design Variables (DV). The aim is to find the

values for the DV that maximizes and minimizes the objective function with respecting the constraints.

In mechatronic design, multidisciplinary models are involved in the design cycle with a strong interaction between them. Therefore, optimizing a model without taking into account the others will generate integration problems. The optimal design may even tend to converge on an absurd solution for the system (Chapman and Pinfol 2001). Given this state, MDO has been recognized as a promising solution to optimize globally the mechatronic system (Alexandrov 2005).

To express the MDO problem we use the All-At-Once formulation (AAO). This formulation contains all the coupling variables, their copies, state variables, consistency constraints and residual equations. The formulation of the other problems derive from this definition (Figure 1-20). For more details, readers can refer to (Cramer, Dennis et al. 1994).

$$
\begin{aligned}
\text{minimize} \quad & f_0(\mathbf{x},\mathbf{y}) + \sum_{i=1}^{N} f_i(\mathbf{x}_0,\mathbf{x}_i,\mathbf{y}_i) \\
\text{with respect to} \quad & \mathbf{x},\hat{\mathbf{y}},\mathbf{y},\bar{\mathbf{y}} \\
\text{subject to} \quad & \mathbf{c}_0(\mathbf{x},\mathbf{y}) \geq 0 \\
& \mathbf{c}_i(\mathbf{x}_0,\mathbf{x}_i,\mathbf{y}_i) \geq 0 \qquad \text{for} \quad i = 1,\ldots,N \\
& \mathbf{c}_i^c = \hat{\mathbf{y}}_i - \mathbf{y}_i = 0 \qquad \text{for} \quad i = 1,\ldots,N \\
& \mathbf{R}_i(\mathbf{x}_0,\mathbf{x}_i,\hat{\mathbf{y}}_{j\neq i},\bar{\mathbf{y}}_i,\mathbf{y}_i) = 0 \qquad \text{for} \quad i = 1,\ldots,N
\end{aligned}
$$

| Symbol | Definition |
|---|---|
| $x$ | Vector of design variables |
| $y^t$ | Vector of coupling variable targets (inputs to a discipline analysis) |
| $y$ | Vector of coupling variable responses (outputs from a discipline analysis) |
| $f$ | Objective function |
| $c$ | Vector of design constraints |
| $c^c$ | Vector of consistency constraints |
| $\mathcal{R}$ | Governing equations of a discipline analysis in residual form |
| $()_0$ | Functions or variables that are shared by more than one discipline |
| $()_i$ | Functions or variables that apply only to discipline $i$ |
| $()^*$ | Functions or variables at their optimal value |

Figure 1-20. The All At Once problem formulation (Lambe and Martins 2012)

### 1.5.2 MDO architectures

The term MDO architecture identifies how the simulation blocks, analysis elements, algorithms and overall process flows are related between each other. The architecture refers to the algorithmic strategy to solve the problem. Different MDO architectures are available from the literature but they are

presented in different manners. An interesting survey is provided in (Martins and Lambe 2013) where clear algorithms of these architectures are proposed to basically describe MDO architectures in unified diagrams called XDSM (Figure 1-21). To read the diagram correctly, we have to follow the indicated numbers step by step. The directions of exchanging the parameters are illustrated in the Figure 1-21. The components consist of the discipline analyses represented by rectangles and a special component (driver) that controls the iteration which is represented by a rounded rectangle. The data flow is shown as thick gray lines. The components take data inputs from the vertical direction and output data in the horizontal direction. Thus, the connections above the diagonal flow from left to right and top to bottom, and the connections below the diagonal flow from right to left and bottom to top. The off-diagonal nodes in the shape of parallelograms are used to label the data. External inputs and outputs are placed on the outer edges of the diagram. The thin black lines show the driver process flow. Loops are represented by $j \rightarrow k$ with $j < k$. It means that if in the j step convergence condition is not reached we repeat again from the k step.

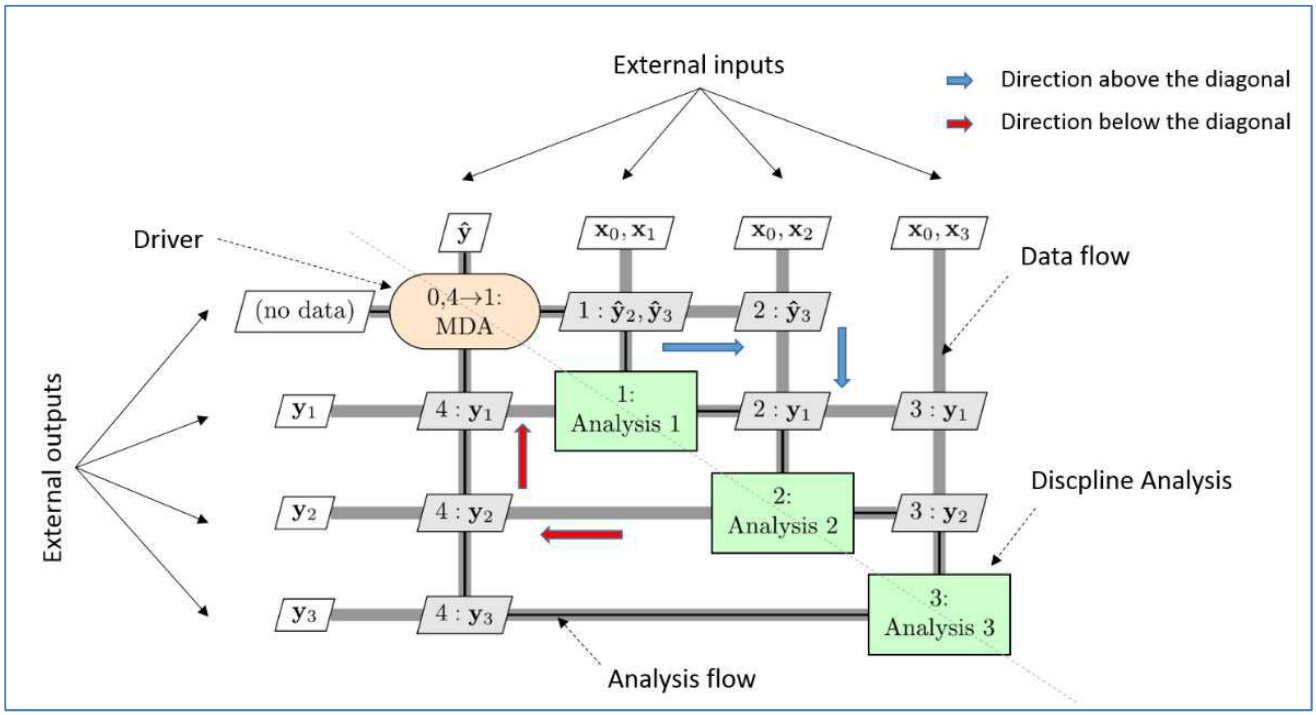



Figure 1-21. XDSM of a multidisciplinary analysis (MDA) process to solve a three-discipline coupled system

Two kinds of architecture can be found in the literature (Tedford and Martins 2010):

Monolevel approaches - The whole system is optimized using a single optimization process:

- Multidisciplinary Design Analysis (MDA)
- Multidisciplinary Feasible (MF)

- Individual Discipline Feasible (IDF)
- All-At-Once (AAO)

Multilevel approaches - Monodisciplinary optimizations are first conducted, before optimizing the whole system:

- Collaborative Optimization (CO)
- Concurrent SubSpace Optimization (CSSO)
- Bi-Level Integrated Systems Synthesis (BLISS)
- Analytical Target Cascading (ATC)

### 1.5.3 MDO tools

Unlike traditional optimization method, MDO is a multidisciplinary methodology (data analysis, visualization, sensitivity analysis, optimization architecture…). MDO is accomplished by several types of software tools. Some CAD tools (Ansys, Matlab, Dymola..) can perform MDO. For instance, Ansys workbench proposes an MDO tool to optimize models in its environment. These MDO tools are limited because they cannot integrate external DE models and do not provide enough algorithms and features (meta-modelling, sensitivity, architectures…).

MDO can also be done by Process Integration and Design Optimization (PIDO) tools. They are specialized in MDO and workflow management and offer various features to integrate DE tools in a common framework. Some examples are illustrated in Figure 1-22 (iSIGHT, modeFRONTIER, PAnO, ModelCenter, Optimus). They have relatively the same process: Optimization problem formulation, creating links between DE models involved in the loop and finally running the optimization algorithm and analyzing the results. Some associated techniques will be explained in the next paragraph.

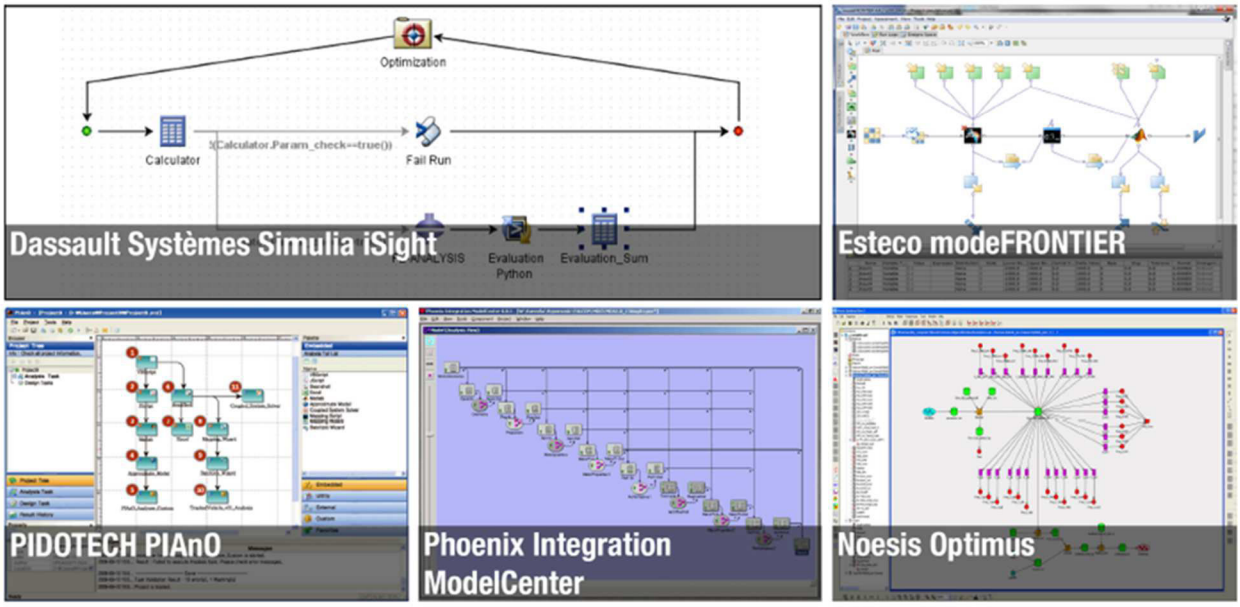



Figure 1-22. Examples of PIDO frameworks

### 1.5.4 Associated techniques

When the MDO problem involves conflicting objectives, the algorithm identifies several solutions that are optimal considering the objective functions, they are called Pareto solutions. Figure 1-23 shows a Pareto front defining the solutions for two objectives (F1 and F2). The multi-objective optimization becomes more difficult with increasing number of objectives (Hammadi, Choley et al. 2012).

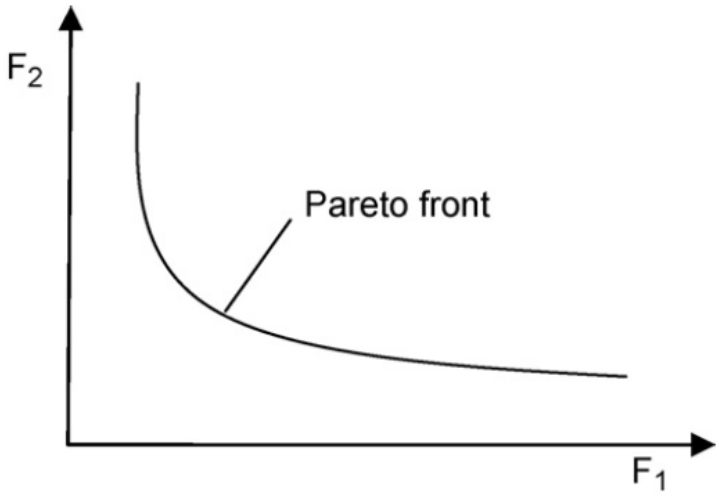



Figure 1-23. Pareto front in multi-objective optimization

Due to the presence of uncertainty, in real life optimization, it is often required to determine less sensitive solutions as robust designs (Figure 1-24). Robust solutions are areas in the search space

where significant changes in design variables produce only insignificant changes in the performance of the design. The challenge is to identify robust regions in the design space. Global and local optimums have to be taken into consideration in algorithm choice (gradient-based algorithms are sensitive to local optimums). The design space should be enlarged if the local optimum is not sufficient.

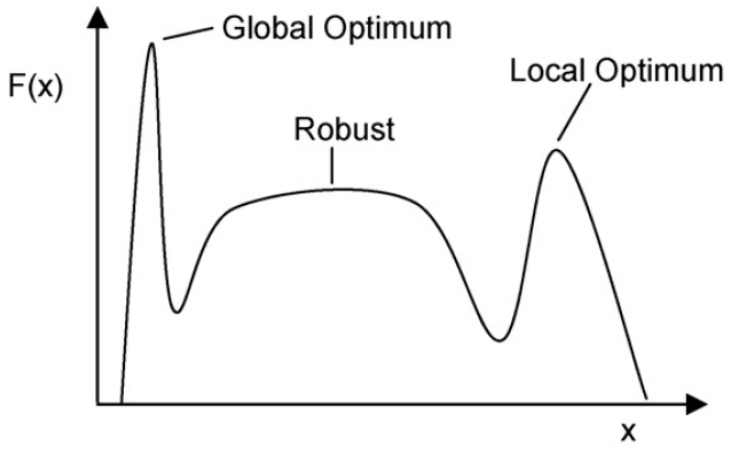



Figure 1-24. Robust vs local vs global optimum

In all the domains of engineering, designers use computationally expensive numerical models. The integration of such models in an optimization process can take a long time due to the significant number of iterations. Surrogate modeling is used in this case to make an approximation of the expensive model and then use it in the optimization process. In the field of surrogate modeling, three steps are required as we can see in Figure 1-25. An initial set of sample points is generated using a statistical method of Design of Experiment (DoE). These points are then used as inputs to run the simulation model and get function evaluations. Finally, a surrogate model type is selected to represent the response surface of the simulation model (Gaussian process (GP), polynomial regression, multi-variate adaptive regression splines (MARS)). If the precision of the model is not sufficient, new points are generated to adapt the model (Barton and Meckesheimer 2006).

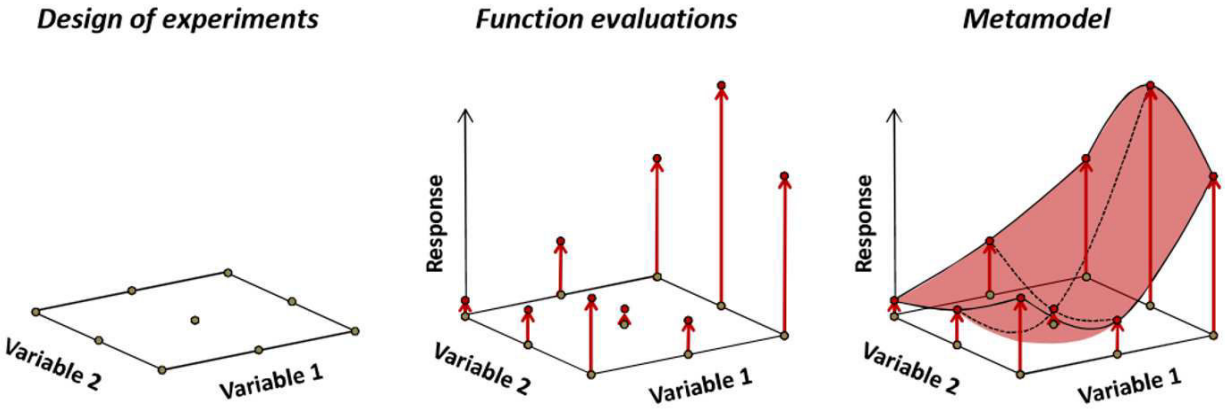



Figure 1-25. The three steps of the surrogate modelling process

Another Surrogate modelling method appeared recently, called Space Mapping. The difference with the first approach is that we assume the existence of two models for the same system: a fine model (precise and expensive) and coarse model (cheaper and less accurate). The idea is to use the coarse model in the optimization process and to update it with the fine model to improve the accuracy (Figure 1-26). The algorithm establishes a mapping between the two models using Broyden updates (Bandler, Biernacki et al. 1994). The critical part of the process is the mapping function. Many techniques exist depending on the mapping function (Intput Space Mapping, Manifold Space Mapping, Agressive Space Mapping, Output Space Mapping).

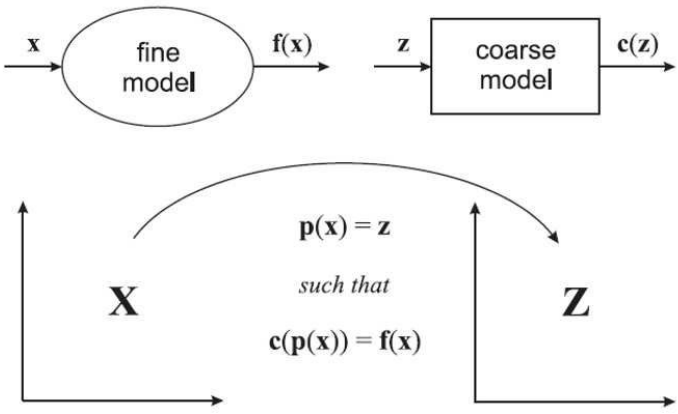



Figure 1-26. Space mapping principle

### 1.5.5 ETB optimization example

The ETB model presented previously in paragraph 2.4.4 was not optimal. In this paragraph we try to optimize it following these specifications:

- The rising time (0° to 90°) should be lower than 150 ms (12V supply voltage)
- The return time with the failsafe system (90° to 10°) should be lower than 300ms (0V supply voltage and open circuit)
- The maximum torque of the throttle should be between 2N.m and 3.5N.m

The optimization problem presented contains two contradictory objectives because if we want to decrease the return time we should use a spring with high stiffness and this will delay the rising time. Therefore, a multi-objective optimization is applied by combining Modelica with ModelCenter software. The optimization details are presented in Table 1-3.

Table 1-3. Optimization problem

| | Unit | Lower bound | Start value | Upper bound |
|---|---|---|---|---|
| Objectives to minimize | | | | |
| Rize_time | s | - | 0.24 | 0.15 |
| Return_time | s | - | 3.38 | 0.30 |
| Problem Constraints | | | | |
| $T_l$_max | N.m | 2 | 1.8 | 3.5 |
| θ_max | deg | 89 | 90 | 91 |
| Design Variables | | | | |
| $K_m$ | N.m/A | 0.01 | 0.02 | 0.08 |
| $R_m$ | Ohm | 1 | 1.5 | 4 |
| $N_g$ | - | 15 | 20 | 30 |
| $C_l$ | N.m/rad | 0.1 | 0.3 | 1 |

Note that the maximum angle was added to the constraints to force the optimizer to respect the requirement full closed to full open. Figure 1-27 shows the Modelica model of the ETB with the design variable (in blue) and the objectives (in orange).

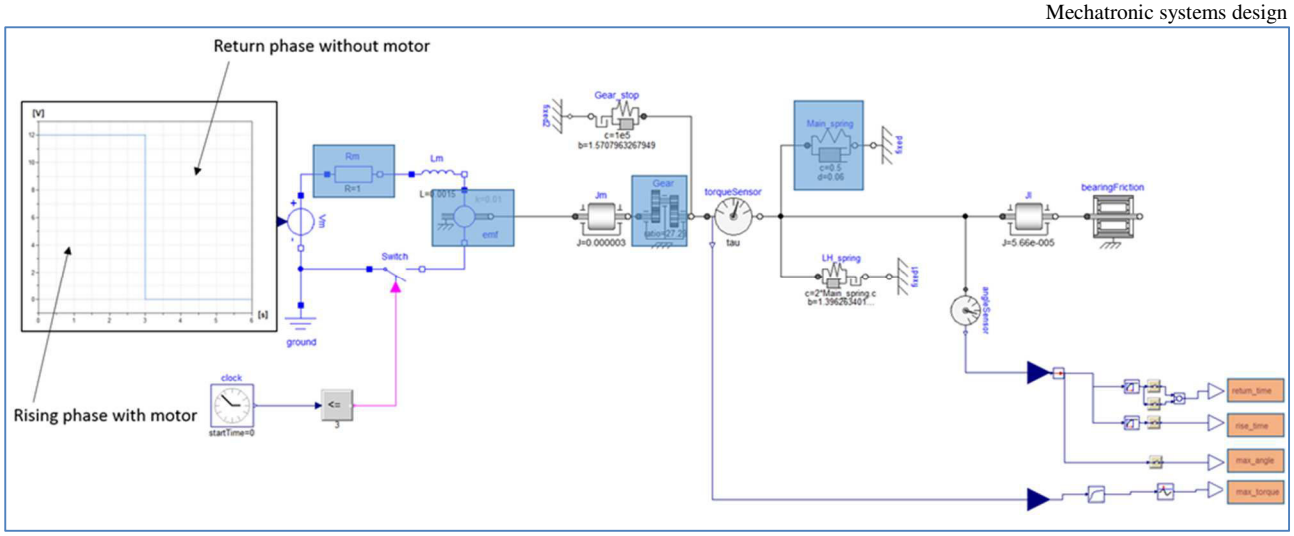
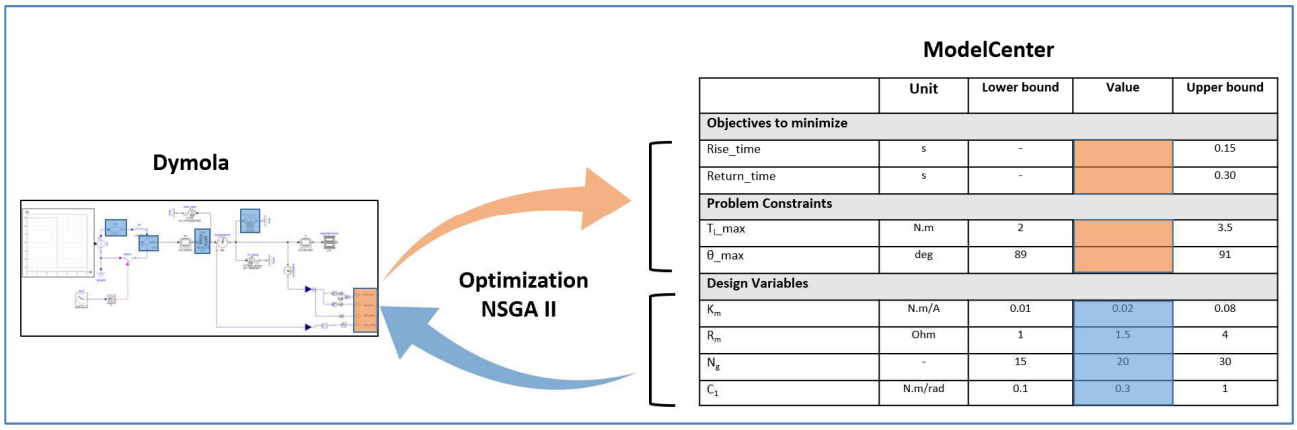



Figure 1-27. Modelica model for optimization

To carry out the optimization, Modelica was combined with ModelCenter software (Figure 1-28). From the algorithms available in ModelCenter, we have chosen the NSGA II algorithm (Nondominating Sorting Genetic Algorithm).




| | Unit | Lower bound | Value | Upper bound |
|---|---|---|---|---|
| **Objectives to minimize** | | | | |
| Rise_time | s | - | | 0.15 |
| Return_time | s | - | | 0.30 |
| **Problem Constraints** | | | | |
| $T_L$_max | N.m | 2 | | 3.5 |
| θ_max | deg | 89 | | 91 |
| **Design Variables** | | | | |
| $K_m$ | N.m/A | 0.01 | 0.02 | 0.08 |
| $R_m$ | Ohm | 1 | 1.5 | 4 |
| $N_g$ | - | 15 | 20 | 30 |
| $C_1$ | N.m/rad | 0.1 | 0.3 | 1 |

Figure 1-28. Optimization between modelica and ModelCenter software

Since the problem is multi-objective, the solution is not unique but we obtain a set of Pareto solutions. The Pareto Front is presented in the Figure 1-29. After 990 iterations, the solutions found present good performances and respect the torque constraint. The rising time can attend 118ms, the return time can attend 150ms and the designer should make the compromise between these two objectives. The torque value is in the range between 2N.m and 3.5N.m and this is sufficient to overcome the resistive torques (Mcharek, Hammadi et al. 2016).

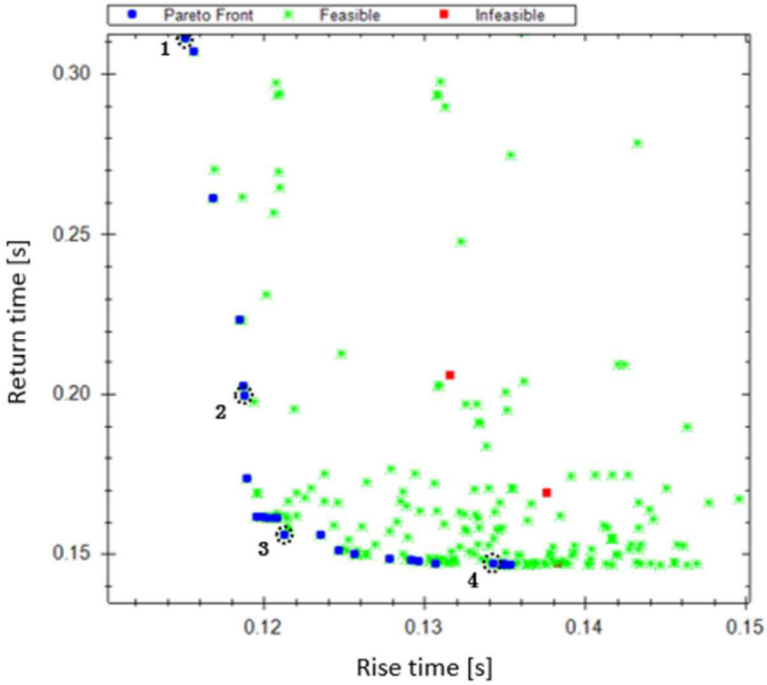



Figure 1-29. Pareto Front solution

Four points are selected with the associated design variables in the Table 1-4.

Table 1-4. Four selected solutions

| | Unit | 1 | 2 | 3 | 4 |
|---|---|---|---|---|---|
| Objectives values | | | | | |
| Rizing_time | ms | 128 | 118 | 120 | 131 |
| Return_time | ms | 206 | 302 | 190 | 250 |
| Constraints values | | | | | |
| $T_l$_max | N.m | 2.0 | 2.8 | 2.1 | 2.3 |
| θ_max | deg | 90 | 90 | 90 | 90 |
| Design Variables | | | | | |
| $K_m$ | N.m/A | 0.026 | 0.027 | 0.023 | 0.020 |
| $R_m$ | Ohm | 1.23 | 1.32 | 1.13 | 1.26 |
| $N_g$ | - | 20.71 | 27.29 | 23.68 | 25.73. |
| $C_1$ | N.m/rad | 0.77 | 0.56 | 0.80 | 0.61 |

The solution 2 is then compared with the initial set for illustration as shown in Figure 1-30.

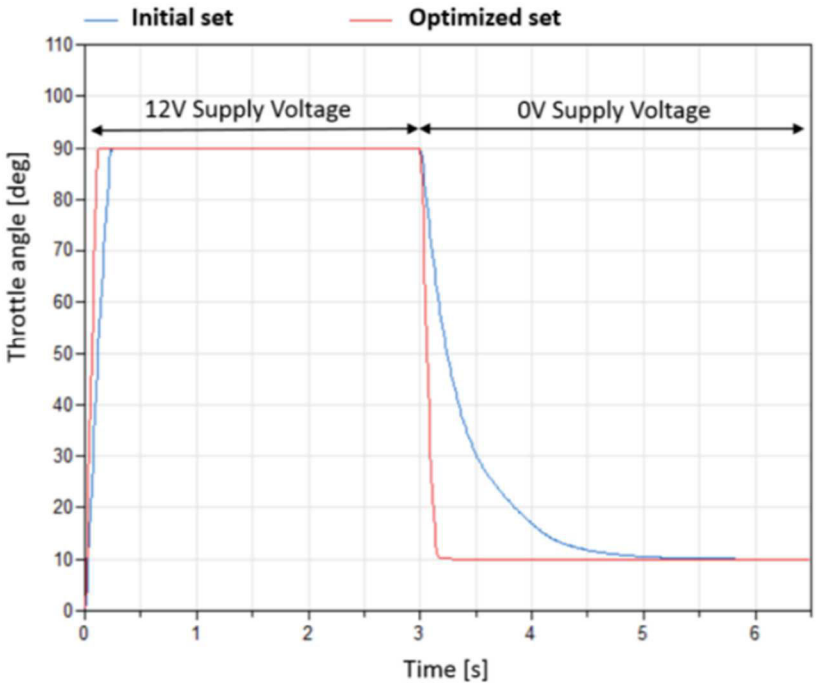



Figure 1-30. Comparison between the initial and the optimized set

### 1.5.6 MDO challenges

MDO is essential to the design and operation of a complex system. It has been successfully applied in the designs of many complex systems such as aircrafts, automobiles, shipbuilding, and civic infra-structures. However, industrial are still facing problems applying MDO due to:

- The difficulty of formalizing real problems into MDO problems. If the optimization problem is poorly defined, the goal will not be reached
- The difficulty to apply MDO on the integration phase due to the complexity of DE tools
- The lack of human decision-making support and collaboration means in MDO as reported in the NSF workshop
- The difficulty to integrate MDO to the design cycle

Regarding MDO challenges and those of SE and DE, we propose in the next section our research problematic.

## 1.6 Research problematic

### 1.6.1 Literature review

Design cycle is recognized as the primary contributor to the final product form, cost and reliability. The major opportunities for cost savings occur in this phase. There is a need to rapidly conduct design analyses involving multiple fields communicating together (Törngren, Qamar et al. 2014). Torry-Smith argues in his survey that the most reported challenges in mechatronic design are related to how information linked to the product concept can be shared across engineering disciplines (Torry-Smith, Qamar et al. 2013). We cannot share correctly the information across the design cycle or reuse efficiently previous works if SE, DE, and MDO are separated.

The fragmentation between SE and DE can be explained by:

- The nature of the results is different between the two fields (qualitative vs quantitative) (Simpson and Martins 2011).
- Different views and perspectives about the system (hierarchical decomposition vs disciplinary decomposition) (Zheng, Le Duigou et al. 2016).
- Incompatible tools used by system engineers (UML, SysML..) and disciplinary engineers (CAD, CAE, FEM..) (Borches and Bonnema 2010).
- Different capabilities between SE and DE (non-technical vs technical) (Price, Raghunathan et al. 2006).

The fragmentation between DE and MDO can be explained by:

- Different coupling methods between the analysis tools (semi-formal and manual vs formal and automatic) (Hiriyannaiah and Mocko 2008).
- Different strategies between DE and MDO (exploration vs optimization) (Simpson and Martins 2011).
- DE tools are tightly associated with a specific discipline but MDO are more generic and cut across disciplines (specific vs generic) (Simpson, Toropov et al. 2008).
- The definition of the MDO problem is time consuming and incompatible with concurrent disciplinary engineering (Rocca and L. Van Tooren 2009).

### 1.6.2 MIMe Project feedback

In order to have a realistic design approach, it is essential to meet the needs of companies. The MIMe project provides an excellent opportunity to access both industries in charge of new methodologies

and academic experts. It enables the industrial problems and requirements to be identified and discussed. My research activities have always been linked to the meetings of MIMe projects and the feedbacks of the involved industrials. Here is a summary of the reflections constructed with the different work packages:

WP1 – IT expertise

- Issues related to the database should be considered early in the conceptual phase of a collaborative framework.
- The traceability of the exchanges during the collaboration is important for reuse.

WP2 – KM expertise

- Traditional means of collaboration (emails, phone, visio…) are inadequate for concurrent engineering and reuse
- The semantic mismatch between stakeholders generates problems during collaboration phase.

WP3 – SE expertise

- Communication between SE and DE is crucial for trade-offs and cannot be performed with the current tools.
- The need for dynamic verification of requirements during concurrent engineering.

WP5 – Design process expertise

- When synchronizing MDO with the design cycle activities, how can we obtain the right results at the right moment?
- How can we ensure the collaboration between two companies having different PLM systems?
- How can we reduce the design efforts in reuse phases?

All these points raise serious problems. The next paragraph presents our research problematic in this context.

### 1.6.3 Research problematic and objectives

The main goal of the design cycle is to realize in a short time a high-quality design solution that satisfies the customer requirements. Disciplinary engineers are at the center of this cycle, they use various heterogeneous tools. As we explained before, the collaboration between disciplinary engineers is not efficient and it is disconnected from SE and MDO (Figure 1-31). The problematic of this work can be summarized in two questions:

- Q1: How can we ensure a dynamic collaboration at the disciplinary level while remaining coherent with the system level?

- Q2: How can we formalize the knowledge generated during the collaborative design to facilitate reuse and multidisciplinary design optimization?

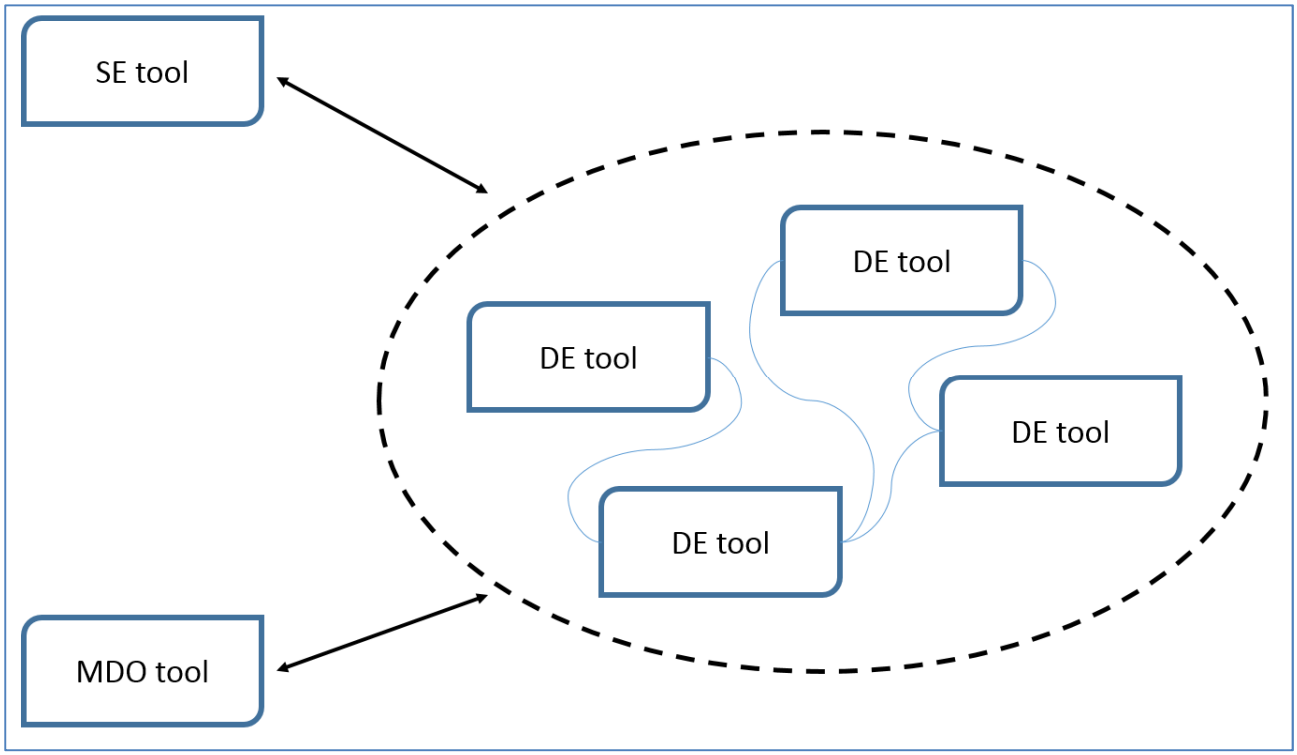
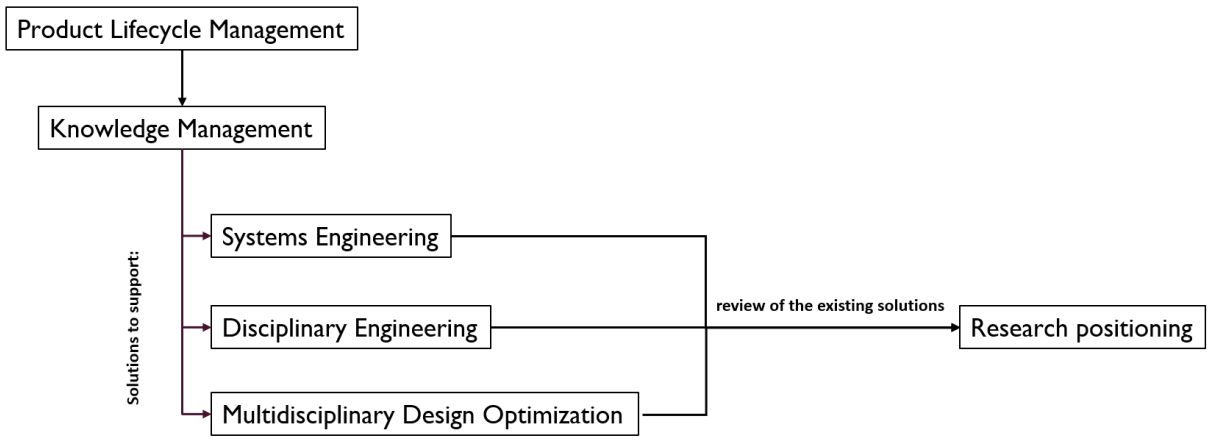



Figure 1-31. Research problematic

# Chapter 2 Product Lifecycle Management and Knowledge Management Solutions

*The topic of knowledge management and the technologies associated are nowadays receiving ample attention, especially in mechatronic design. This chapter is devoted to the description of knowledge based solutions to support SE, DE, and MDO. This state of the art will be followed by a work positioning.*



## 2.1 Product Lifecycle Management and knowledge

CIMData defines PLM as "a strategic business approach that applies a consistent set of business solutions in support of the collaborative creation, management, dissemination, and use of product definition information across the extended enterprise from concept to end of life – integrating people, processes, business systems, and information" (Amann 2002). This section explains the evolution of PLM and its relationship with knowledge management.

### 2.1.1 PLM and knowledge

PLM can be defined as the 'connective tissue' that allows the connection of design software to production and supply chain management software, taking into account the dispersed nature of the extended and collaborative enterprise (Figure 2-1).

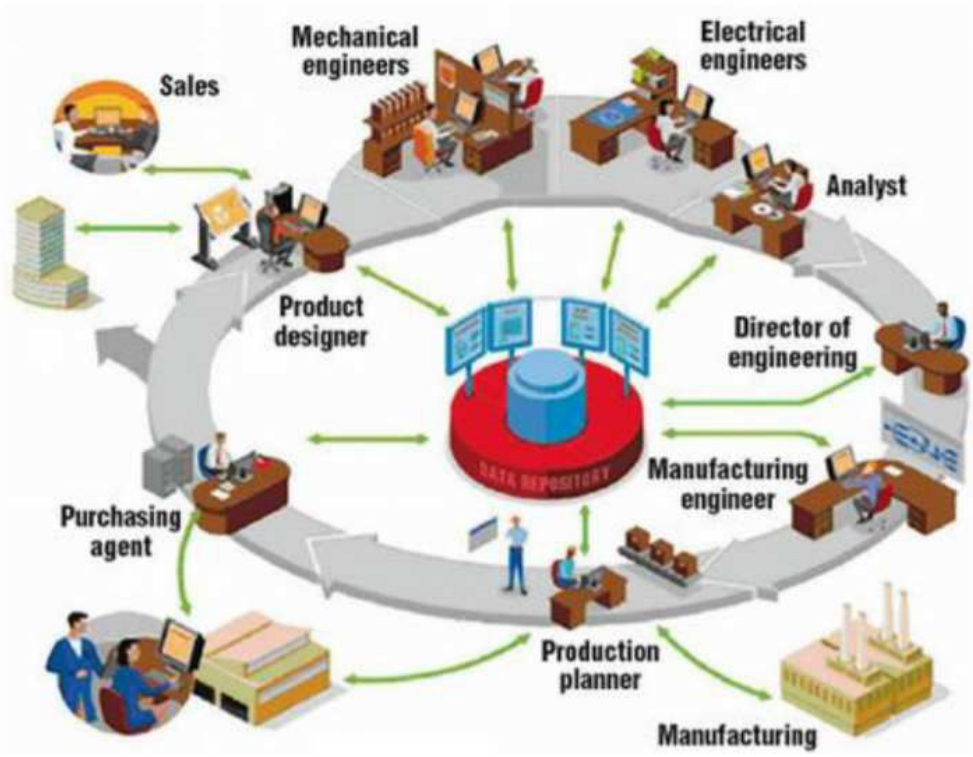



Figure 2-1. PLM functions in the entire product process (Matta, Ducellier et al. 2013)

A PLM system requires a high level of coordination and integration as stated in (Jun, Shin et al. 2007). However, the key enabler in PLM are stakeholders, as well as the knowledge they

generate during collaboration (Bruun and Mortensen 2012). Formalizing this knowledge contributes to efficient reuse and optimization of the design cycle as reported in (El Hani, Rivest et al. 2012).

The concept of knowledge refers to intellectual capital that is available in the company to carry out its design process. The pyramid of knowledge starts with data which are objective and independent. Then, in a specific context, the data are structured and transformed into information. Finally, the interpretation of the information to make decisions is considered as knowledge (Alavi and Leidner 2001).

It is important to incorporate knowledge in the early stages of PLM. Traditionally, uncertainties are embedded in the early phase of the design cycle due to the lack of knowledge. Therefore, decision making is postponed to late stages when more knowledge is accumulated. This is time-consuming and costly. An improvement of the traditional design processes is to increase the level of knowledge in early design phases, implying more design freedom as shown in Figure 2-2 (Ullman 2010).

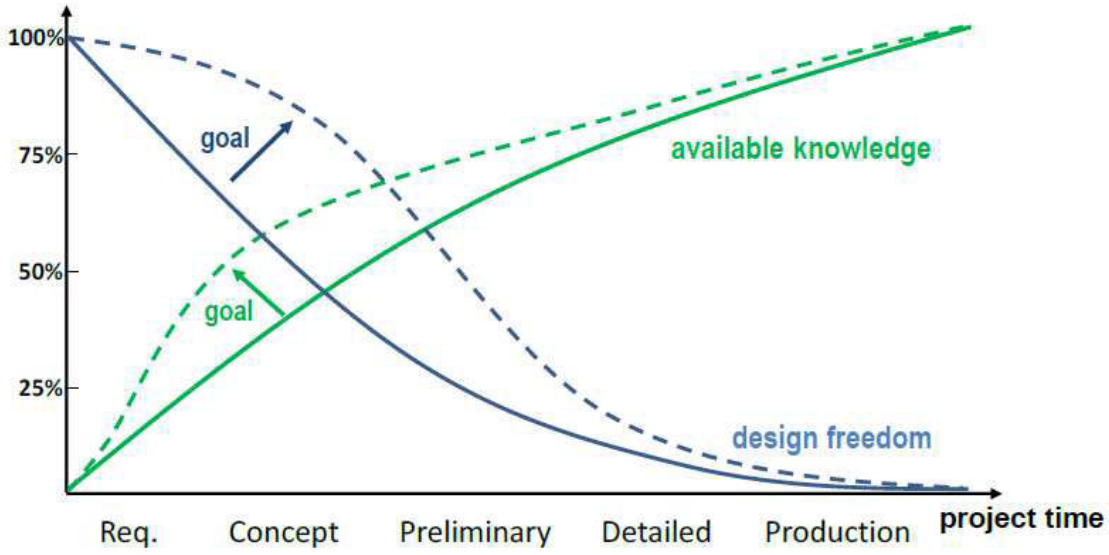



Figure 2-2. Design knowledge and freedom related to the design cycle (Verhagen, Bermell-Garcia et al. 2012)

### 2.1.2 Knowledge exchange in PLM

Nonaka and Takeuchi argued that to reach effective knowledge exchange, we need to convert internalized tacit knowledge into explicit codified knowledge to share it (Nonaka, Takeuchi et al. 1996). In PLM context, there is a spiral conversion of knowledge as illustrated in Figure 2-3.

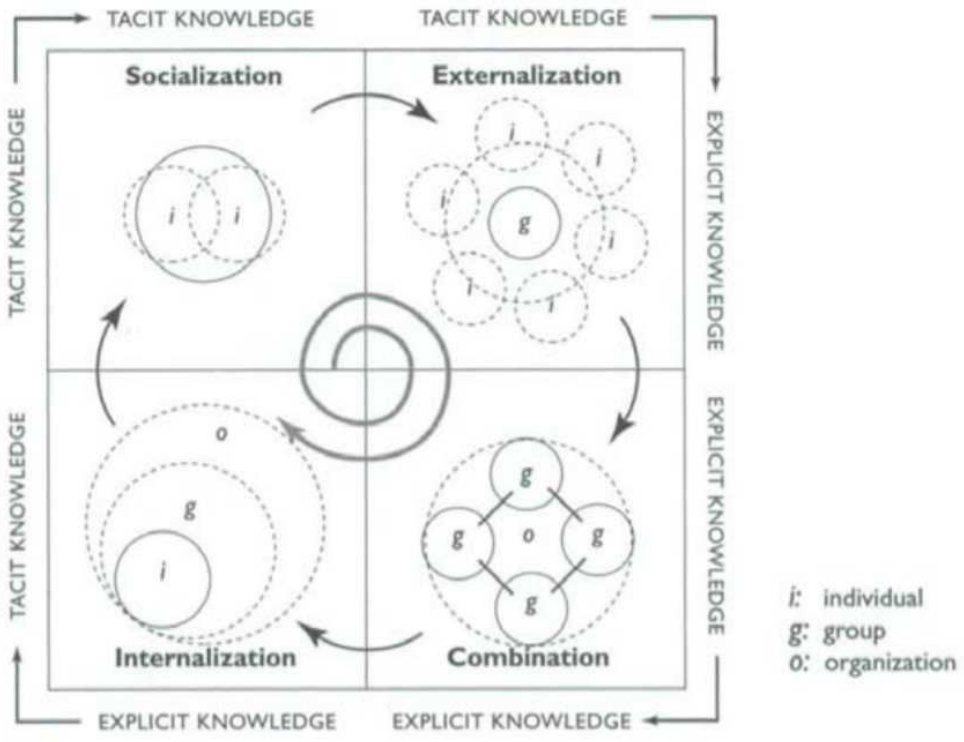



Figure 2-3. Spiral conversion of knowledge (Nonaka 1991)

• Socialization: The sharing of tacit knowledge between individuals. Meetings, video conferences, collaboration, and visualization tools are the major tools of knowledge socialization during the product life cycle.

• Externalization: Tacit knowledge is turned explicit. It requires an effort of formalization to express knowledge in an understandable form for others. CAD tools are among the most widely used tools for externalization within a PLM environment since they encapsulate multidisciplinary design knowledge.

• Internalization: Transforming explicit knowledge into tacit knowledge. It is a learning phase that requires the identification of knowledge relevant to an individual within a broader set of explicit knowledge. Direct tools in PLM are search engines which help actors in locating the required pieces of explicit knowledge. Indirectly, document and workflow management tools have a supportive role in the internalization process since they make explicit knowledge more organized.

• Combination: Combining two or more pieces of knowledge to generate new explicit knowledge. It represents the conversion of simple explicit pieces of knowledge to a more organized structure. Expert systems, for instance, perform combination through inference in PLM. This system can, for example, classify components based on their similarity in geometry.

### 2.1.3 PLM challenges

PLM offers collaborative solutions for product-centric environments. It applies a set of tools and technologies that provide a shared platform for collaboration among product stakeholders (Ameri and Dutta 2005). However, PLM is not enough to cover the complexity of mechatronic collaborative design and its heterogeneous tools (Fortineau, Paviot et al. 2013). This issue is amplified when a company works with partners and suppliers with different PLM-systems (Srinivasan 2011). To overcome this situation, engineers resort to informal collaboration channels (email, meeting, phone…). These means are no longer sufficient in concurrent engineering for they lead to unnecessary iterations and lack of traceability (Dave and Koskela 2009). To overcome these challenges, Knowledge Management (KM) was proposed (Garetti, Terzi et al. 2005).

## 2.2 KM to support PLM in mechatronics

Knowledge is important in the mechatronic design cycle. First, the simultaneous and collaborative design processes depend on effective transfer and interpretation of knowledge between teams. Second, the knowledge captured from previous projects facilitates decision making during the design cycle. Therefore, knowledge should be carefully managed as we will explain in this section.

Research indicates that, in a typical organization, only 4% of organizational knowledge is available in a structured and reusable format and the rest is either unstructured or resides in peoples' minds (Assouroko, Ducellier et al. 2014). The structured knowledge, although small in volume, has high value for companies because it can be accessed easily, mined and used for decision making. Generating structured knowledge, through a transformation from a tacit form into an explicit form, is one of the critical steps of knowledge management.

### 2.2.1 Knowledge classification and representations

Classifying knowledge is important to determine ways to represent it. Three axes are considered while classifying design knowledge (Figure 2-4):

Formal / Tacit: This axis is based on the works cited previously (Nonaka 1991). The formal knowledge is embedded in the product documents, description of the structure and product functions, etc. And tacit knowledge is knowledge related to experience, implicit rules, intuition, and others, which are anchored in the actor's memory.

Product / Process: Product knowledge takes into account information and knowledge about the evolution of the product throughout its life cycle. Process knowledge can be classified into design process knowledge, manufacturing process knowledge, and business process knowledge. In our context, we will only consider the design process knowledge

Compiled / Dynamic: The compiled knowledge is essentially obtained from experience that can be compiled into rules, plans, scripts, etc. Dynamic knowledge encodes information that can be utilized to create extra knowledge structures, not covered by compiled knowledge

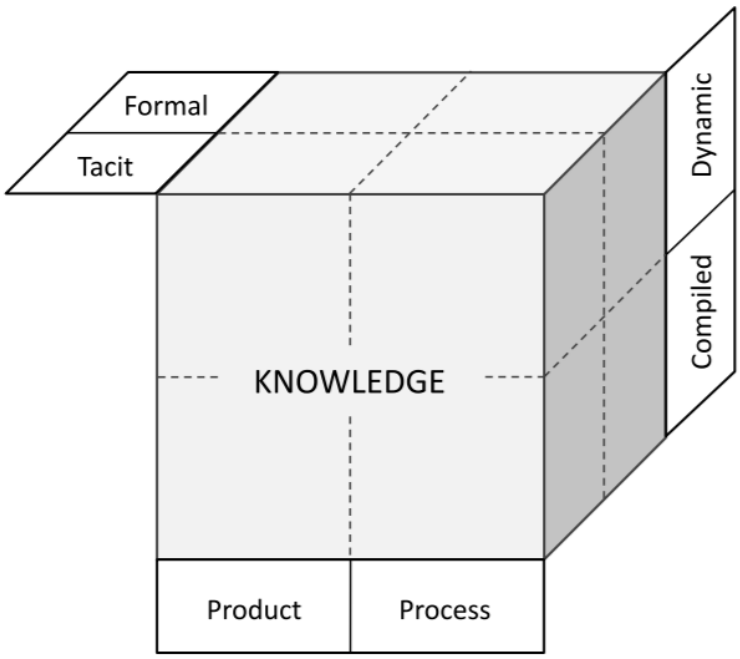



Figure 2-4. Classifying knowledge in three axes (Chandrasegaran, Ramani et al. 2013)

Knowledge representation lists various forms of knowledge that are used or produced at each stage. Recently, a design process view of knowledge was proposed (Chandrasegaran, Ramani et al. 2013). Figure 2-5 shows this interpretation of knowledge representation during the product lifecycle.

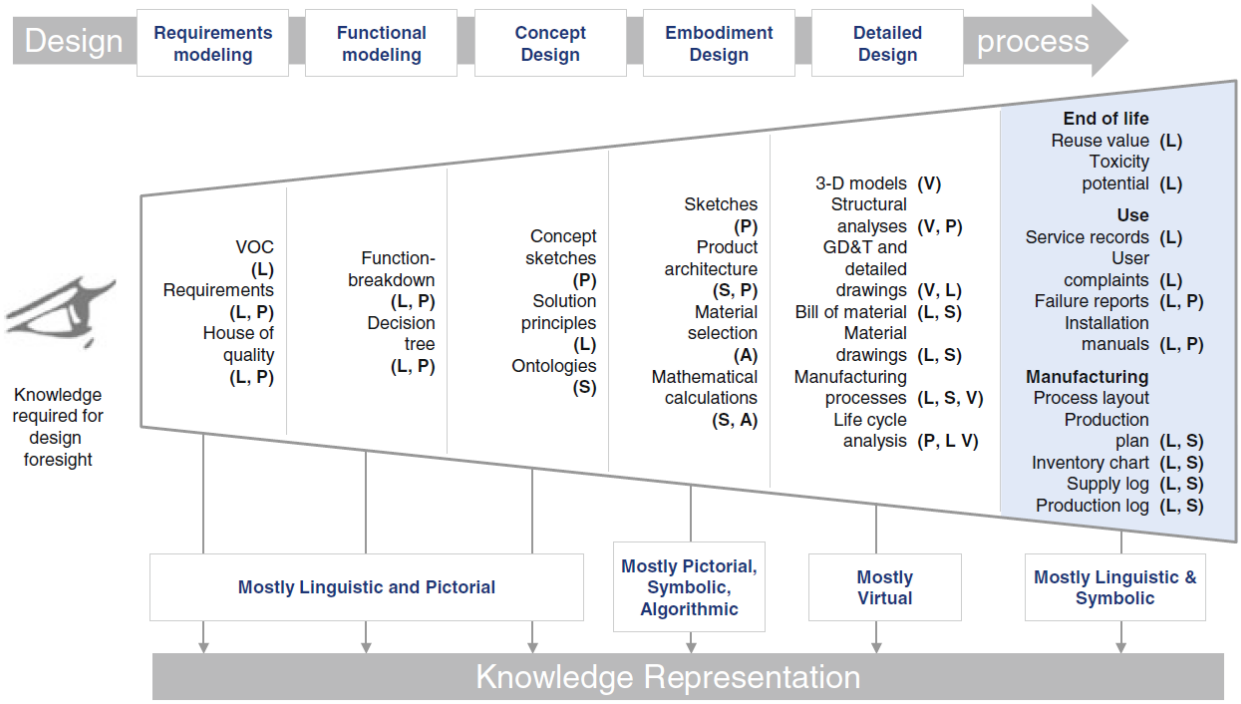



Figure 2-5. Knowledge representations throughout the product lifecycle (Ali 2016)

Five categories of knowledge representation are proposed (Owen and Horváth 2002):

- The pictorial representation, which relates to the knowledge that is communicated through photos, videos, drawings, etc.
- The symbolic representation, which represents knowledge about the logical aspect, such as diagrams, decision tables, charts, etc.
- Linguistic representation, which relates to different knowledge communicated through language, such as in verbal communications, texts, etc.
- The virtual representation, which allows representing the knowledge through virtual models such as CAD, virtual reality models, animations, etc.
- The algorithmic representation, which relates to different expressions mathematics, computer algorithms, calculation procedures, etc.

This helps to highlight the level of detail relating to each form of representation and the ease with which it communicates information: linguistic, then pictorial, then symbolic and / or algorithmic, then virtual (from the most abstract to the most concrete).

Research indicates that wasted time comprises about 60 percent of total operational time in most businesses. The major portion of this waste can be attributed to the absence of an efficient knowledge management system (Karayel, Özkan et al. 2004).

### 2.2.2 Knowledge Management cycle

KM is shown as the encompassing area, where the intention is to enable more efficient and effective use of knowledge assets in the organization. The basic processes involved in KM are creating, storing/retrieving, transferring and applying knowledge (Alavi and Leidner).

Knowledge creation refers to the creativity of an organization to develop novel and useful ideas and solutions, for instance, creating new products, new ideas, more efficient processes and new skills. Knowledge storage/retrieval refers to the retention/accessibility of knowledge assets in organization, namely organizational memory (Garetti, Terzi et al. 2005). That is the knowledge residing in various component forms, including written documentation, structured information stored in electronic databases, codified human knowledge in expert systems, documented organizational procedures and processes and implicit knowledge acquired by individuals and networks of individuals. Knowledge transfer refers to the availability of knowledge throughout an organization, which means distributing knowledge to locations where it is needed and used (Maier and Hadrich 2006). Knowledge application relates to the ability of utilizing knowledge to confront new situations in an organization. The source of competitive advantage for an organization resides in the application of knowledge rather than the knowledge itself (Levy, Loebbecke et al. 2003).
KM encompasses Knowledge Engineering (KE) and Knowledge Based Engineering (KBE) approaches as illustrated in Figure 2-6.

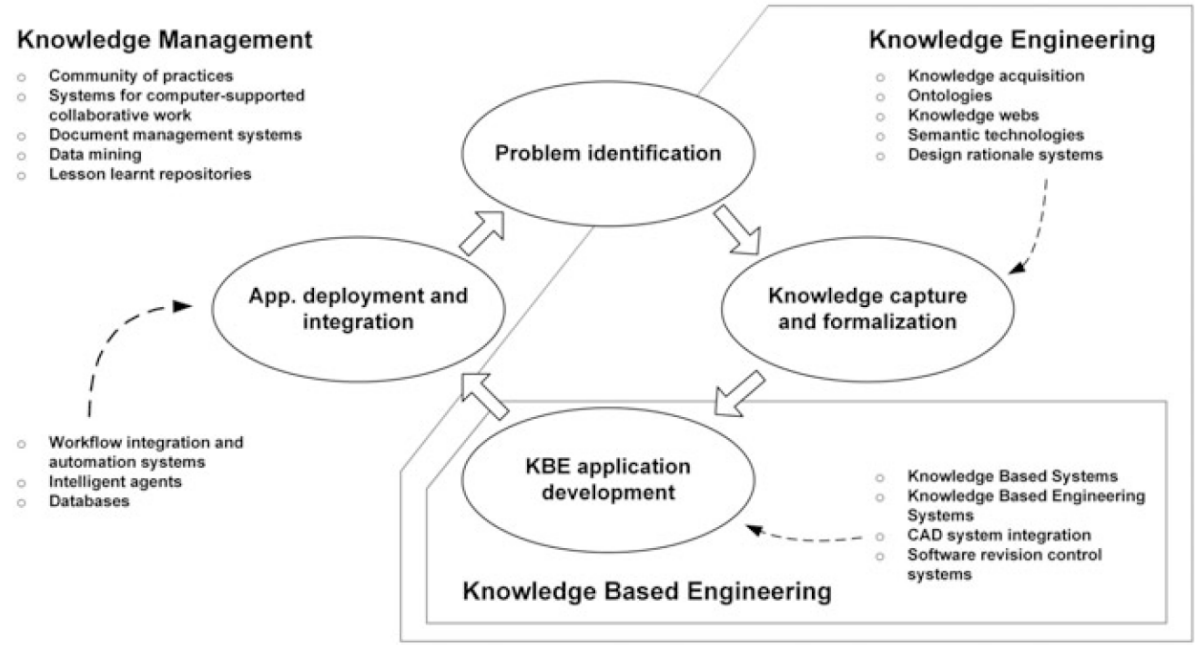



Figure 2-6. Overview of KM, KE and KBE (Chandrasegaran, Ramani et al. 2013)

- Knowledge Engineering (KE)

KE refers to the use of ontologies in the problem identification phase. An ontology can be defined as a definition of a common vocabulary for researchers who need to share information in a domain (Tudorache 2006). It includes machine-interpretable definitions of basic concepts in the domain and the relations among them. An ontology necessarily includes a common vocabulary of terms and a specification of their meaning (Huzar, Kuzniarz et al. 2004). Without specification, the set of ontology concepts would be variously interpretable by different sets of users. With specification, different users (e.g., experts in different lifecycle phases) with different views on a single reality can be accommodated.

- Knowledge Based Engineering

KBE approach is to automate repetitive, non-creative design tasks, which can lead to "significant cost savings" and "free up time for creativity" (Rocca and L. Van Tooren 2009). The meaning of KBE seems to be dynamic and once very tightly integrated with AI techniques. A contemporary description of KBE adopted in this thesis is: Automating non-creative design tasks by utilizing object oriented programming (Chapman and Pinfold 2001).

### 2.2.3 MOKA standard example

The MOKA (Methodology and tools Oriented to Knowledge-based engineering Applications) European project proposed a KM methodology adopted by various European companies. This

methodology aims to capture, organize and store data created in a design process in order to reuse this knowledge in future new product development projects (Klein 2000). It indicates the interaction between domain experts, knowledge engineers, system developers and end users, and their role in the key lifecycle phases (Figure 2-7). The "Capture" phase in MOKA terminology, occurs between the domain expert and knowledge engineer through the development of an informal knowledge model, represented by ICARE forms (Illustrations, Constraints, Activities, Rules and Entities). The informal knowledge can be read by the end user if needed. The next phase is to "Formalize" the informal collected knowledge into a formal knowledge model using MML Language (MOKA Modeling Language which is a UML extension). Knowledge engineers and software developers interact through the formal knowledge model. In "Package" phase, software developers, create appropriate KBE software applications to automate the identified knowledge contained in the formal knowledge model. In "Activate" phase, end user uses the KBE application and ICARE Form to accomplish his activities (Stokes 2001).

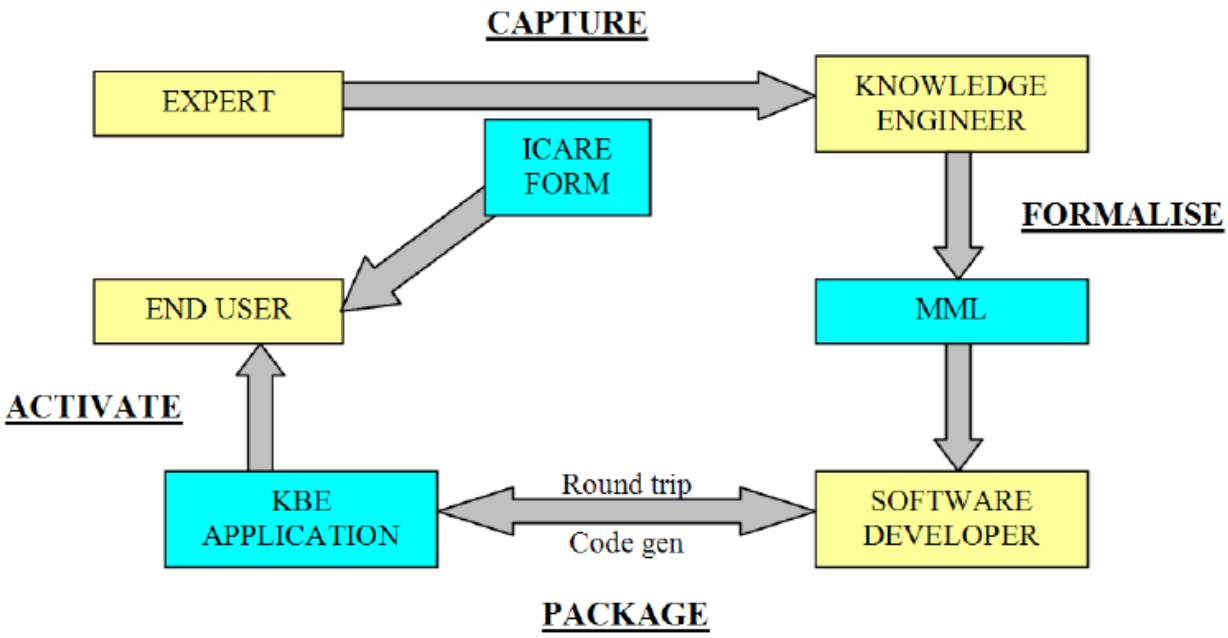



Figure 2-7. MOKA methodology process (MOKA Group, 2000)

### 2.2.4 Criteria for a mechatronic KM solution

Based on this state of the art, we suggest in this paragraph the criteria necessary to connect SE, DE, and MDO to answer the two research question:

- Q1: How can we ensure a dynamic collaboration at the disciplinary level while remaining coherent with the system level?

- Q2: How can we formalize the knowledge generated during the collaborative design to facilitate reuse and multidisciplinary design optimization?

We need a KM platform capable of managing the knowledge generated in disciplinary and system level as well as managing collaboration and conflicts between the different stakeholders. The knowledge generated during this collaboration should be formalized for reuse and MDO purposes. For this end, we need first to establish a connection between SE and DE. Then, a second connection between DE and MDO. We choose common criteria for the two connections as explained in Table 2-1.

Table 2-1. Criteria for KM mechatronic platform

| Criteria | SE – DE connection | DE – MDO connection |
|---|---|---|
| Product knowledge | Quantitative knowledge from SE tools and parameters involved in DE tools | Necessary knowledge for MDO problem resolution and collaboration results |
| Process knowledge | Design process between DE tools for reuse purpose | Process between DE tools to generate MDO architecture |
| Dynamic exchange | Interdisciplinary dynamic exchange and dynamic requirements check | Dynamic exchange between MDO engineers and disciplinary engineers |
| Tools interoperability | Interoperability between SE tool and DE tools | Interoperability between MDO tool and DE tools that are not directly related to the MDO problem |
| Conflicts resolution | System and interdisciplinary conflicts resolution | Conflicts resolution between MDO model results and related DE models |
| Consistency check | Check consistency between disciplinary level models and system model | Check the consistency between MDO model results and the other DE models |

To the best of our knowledge, no KM solution was proposed to deal with a global connection between SE, DE, and MDO. However, important works were made to support separately SE,

DE, and MDO. In the next 3 sections, we will explore these solutions and evaluate them according to our criteria.

## 2.3 SE support solutions

### 2.3.1 SysDICE framework

The lack of an easily accessible analytical capability makes it difficult for systems engineers to quickly understand consequences of inevitable changes in requirements. SysDICE methodology use SysML as a common language between system engineers and disciplinary engineers (Figure 2-8). They use simultaneously SysML diagrams for requirements, functions and conceptual solutions. First, system engineers and disciplinary engineers generate system models. The second step is the mathematical formulation related to these models. The final step is the evaluation by connecting this model with DE tools (Chami and Bruel 2015).

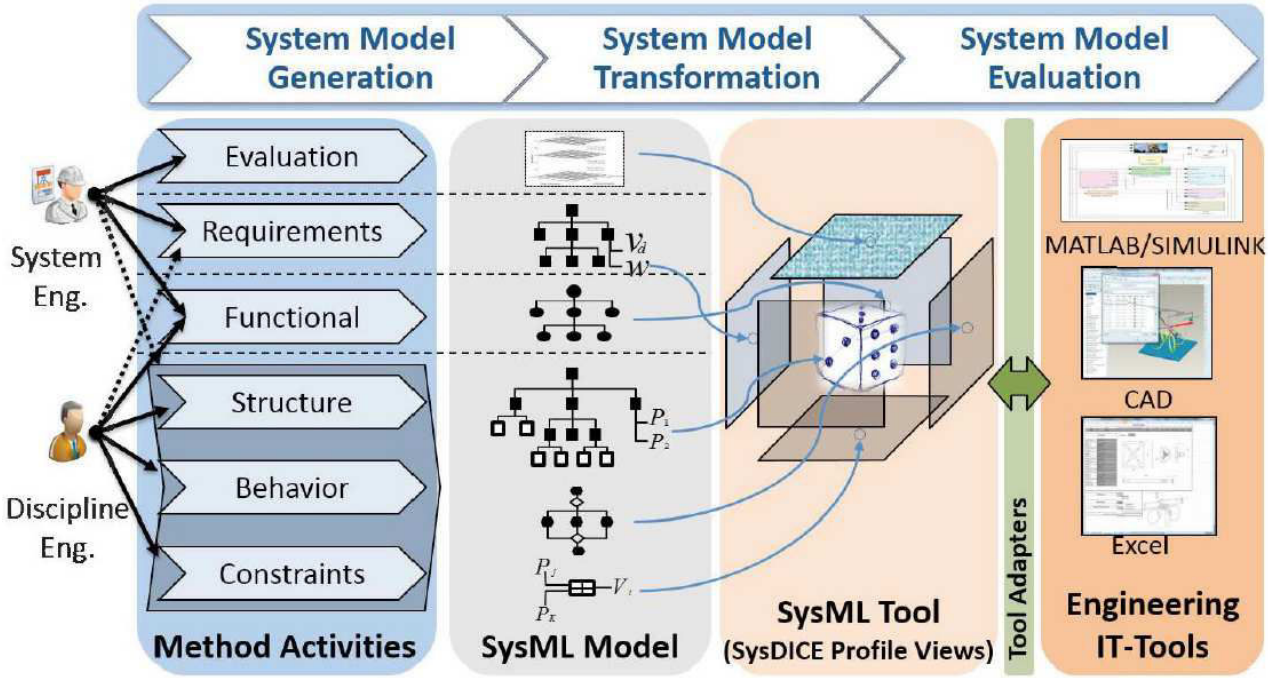



Figure 2-8. SysDICE framework (Chami and Bruel 2015)

This solution is interesting to fil the gap between SE and DE. It provides a common basis for system engineers and disciplinary engineers. However, it does not seem suitable for concurrent engineering, because the exchange of knowledge is not dynamic. Additionally, the design process is not captured for reuse (Table 2-2).

Table 2-2. SysDICE framework evaluation

| Criteria | **Product knowledge** | **Design process knowledge** | **Dynamic Exchange** | **Tools interoperability** | **Conflict resolution** | **Consistency check** |
|---|---|---|---|---|---|---|
| Evaluation | The product knowledge is centralized in SysML diagrams | No process knowledge is considered | The possibility to exchange via the SysDICE profile | Connectors are created to connect SysML with DE tools | Only system level conflict are considered | The consistency is checked only at the system level |

### 2.3.2 SLIM framework

SysML models are capable of describing a given system configuration with a high degree of detail, it is difficult to evaluate properly how well the design meets the requirements or to perform important trade-offs between performance, cost, and risk. The SLIM platform creates connectivity patterns between SysML models and DE models (Figure 2-9). Then it checks the changes and updates the two models of the connection. System engineers can drive automated requirements verification, system simulations, trade studies and optimization (Bajaj, Zwemer et al. 2011). It also provides plugins to integrate the system model to a variety of CAD/CAE tools.

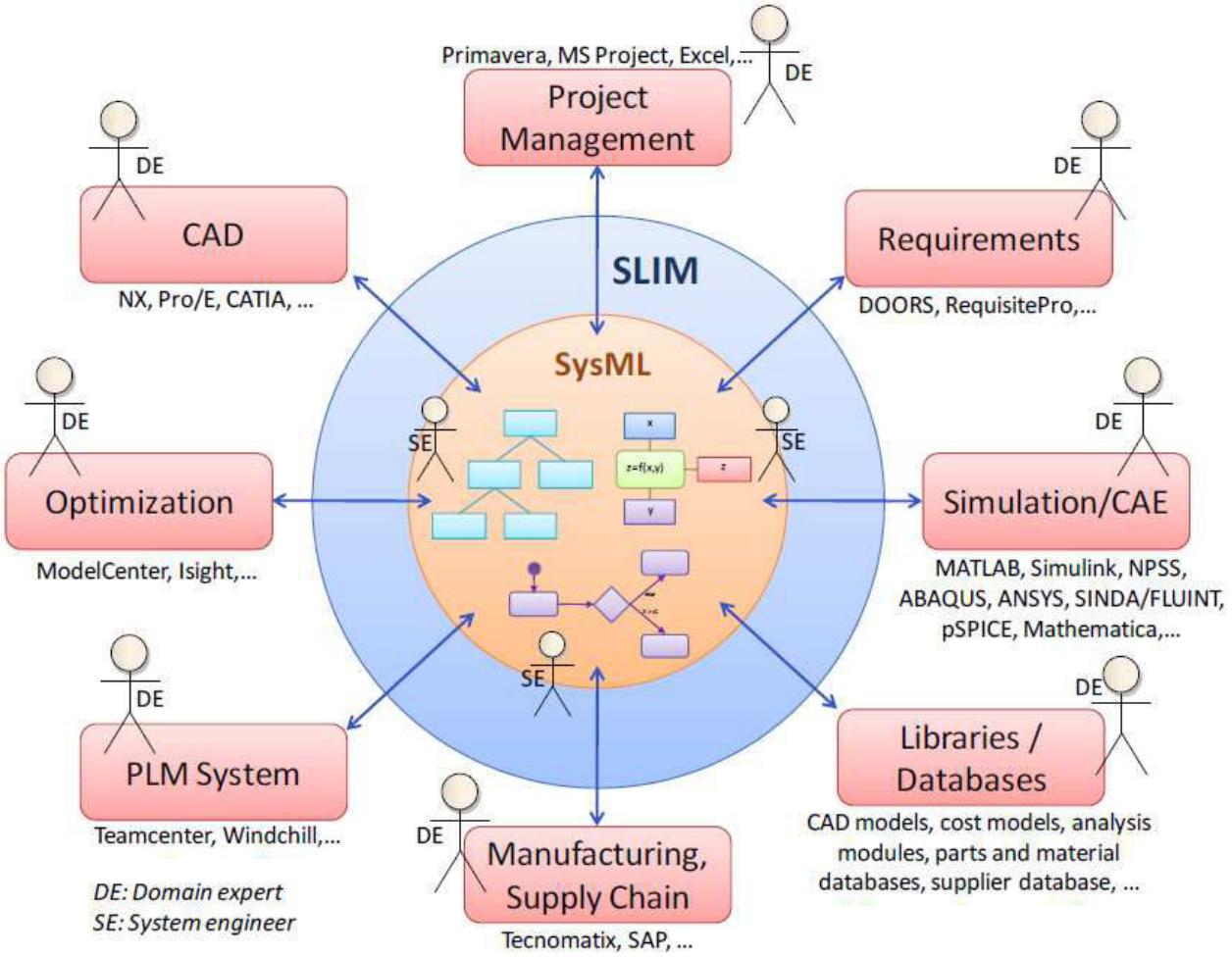



Figure 2-9. Conceptual architecture of SLIM (Bajaj, Zwemer et al. 2011)

SysML here captures the product knowledge, however, the traceability of the collaboration and conflict resolution are missing. This makes the collaboration and reuse phases difficult in mechatronics (Table 2-3).

Table 2-3. SLIM framework evaluation

| Criteria | **Product knowledge** | **Process knowledge** | **Dynamic Exchange** | **Tools interoperability** | **Conflict resolution** | **Consistency check** |
|---|---|---|---|---|---|---|
| Evaluation | The product knowledge is centralized in SysML diagrams | The process knowledge is not considered in this approach | Partially covered | Interoperability between SE, DE and MDO tools | Not considered | Consistency art system level |

### 2.3.3 SysML-PIDO

SysML-PIDO connection is based on the parametric diagram of SysML. This diagram (when created properly) has the necessary information to create an analysis model that can be executed through the PIDO framework (Kim, Fried et al. 2012). Each block of the parametric diagram is considered as an analysis tool. A connection has been made between MagicDraw tool and ModelCenter tool (Figure 2-10). This connection gives to the system engineer to execute tradeoffs and optimization.

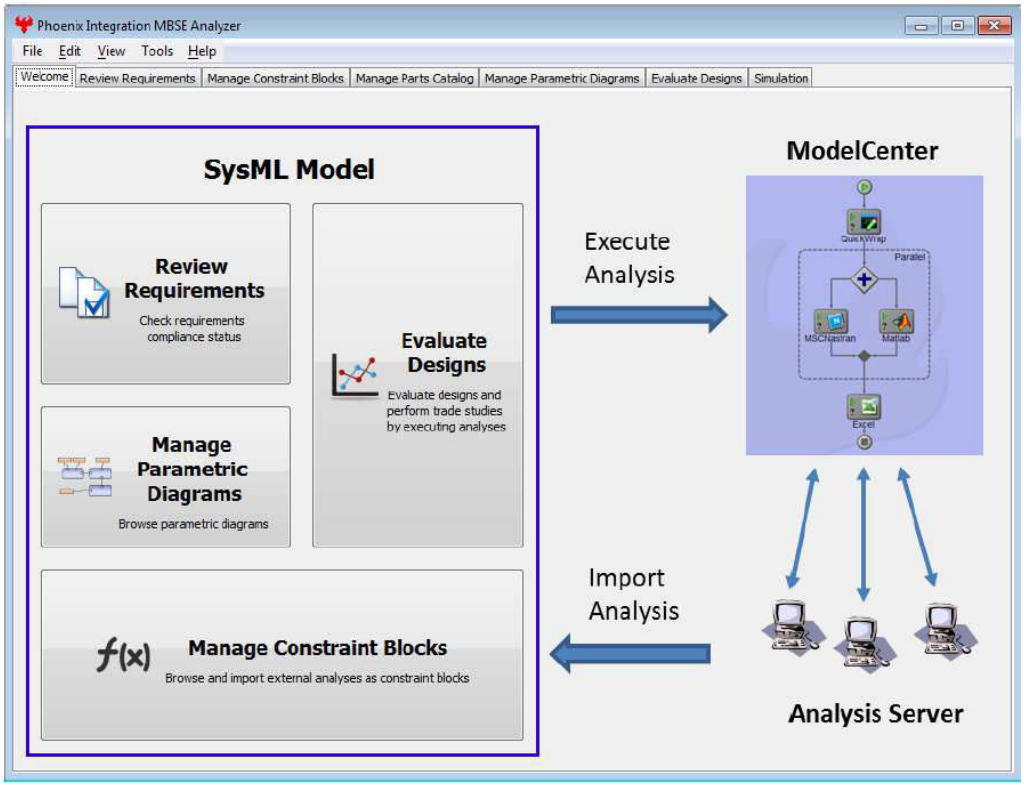



Figure 2-10. SysML-PIDO connection using MagicDraw and ModelCenter software (Kaslow, Soremekun et al. 2014)

This methodology requires a formalization of the DE tools to match the connections defined in the parametric model. The collaboration and the dynamic exchange are not covered also (Table 2-4).

Table 2-4. SysML-PIDO framework evaluation

| Criteria | **Product knowledge** | **Process knowledge** | **Dynamic Exchange** | **Tools interoperability** | **Conflict resolution** | **Consistency check** |
|---|---|---|---|---|---|---|
| Evaluation | The product knowledge is centralized in SysML | The paramet-ric diagram is | The ex-change is via MDO tool | Interoperability between SE tool and MDO tool | No conflict resolution | The con-sistency is di- |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | the only process knowledge | and it is not dynamic | | | rectly considered inside MDO tool |

## 2.4 DE support solutions

### 2.4.1 Multiview point methodology

This methodology focuses on the dependency between people, model and tool levels (Figure 2-11). The dependency is visualized using semantic web solution for inter and intra viewpoints. This tool permits consistency checking and supports effective collaborative design (Törngren, Qamar et al. 2014). The dependency graph proposed in this approach facilitates the reuse phase. However, it does not support the dynamic exchange between engineers (Table 2-5).

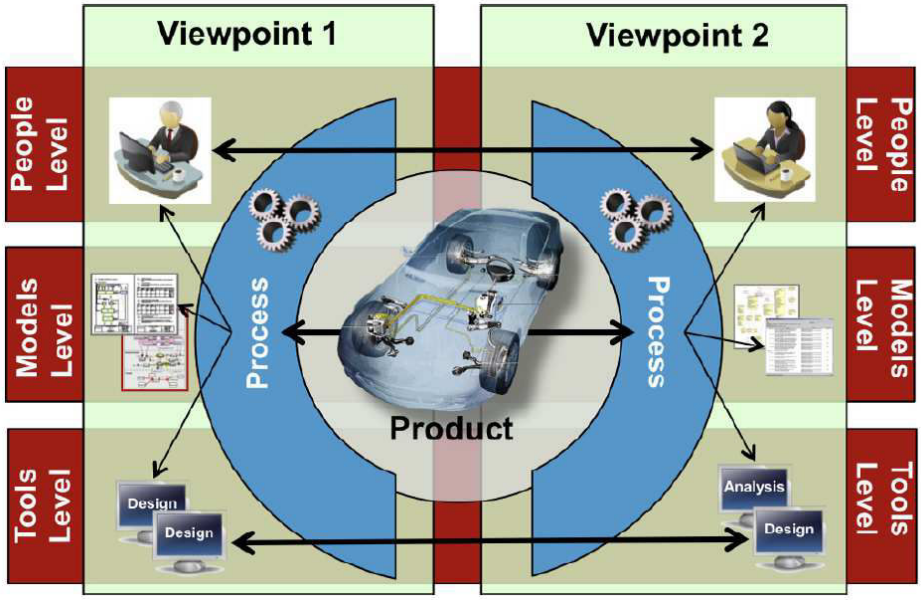



Figure 2-11. Multiviewpoint concept (Törngren, Qamar et al. 2014)

Table 2-5. Multiviewpoint framework evaluation

| Criteria | **Product knowledge** | **Process knowledge** | **Dynamic Exchange** | **Tools interoperability** | **Conflict resolution** | **Consistency check** |
|---|---|---|---|---|---|---|
| Evaluation | Parameters in DE tools | Dependency graph | Not considered | Not considered | No conflict resolution is proposed | Conflict check is managed by dependency graph |

### 2.4.2 PROXIMA framework

This approach manages the system parameters and constraints as well as the design process using DSM view (Figure 2-12). The design process permits reasoning over inconsistencies and their origins. Design space exploration is used to optimize the process and minimize the inconstancies (Dávid, Denil et al. 2016). PROXIMA is an efficient enabler of collaborative design and can support the project managers however it does not offer a collaborative exchange environment for disciplinary designers (Table 2-6).

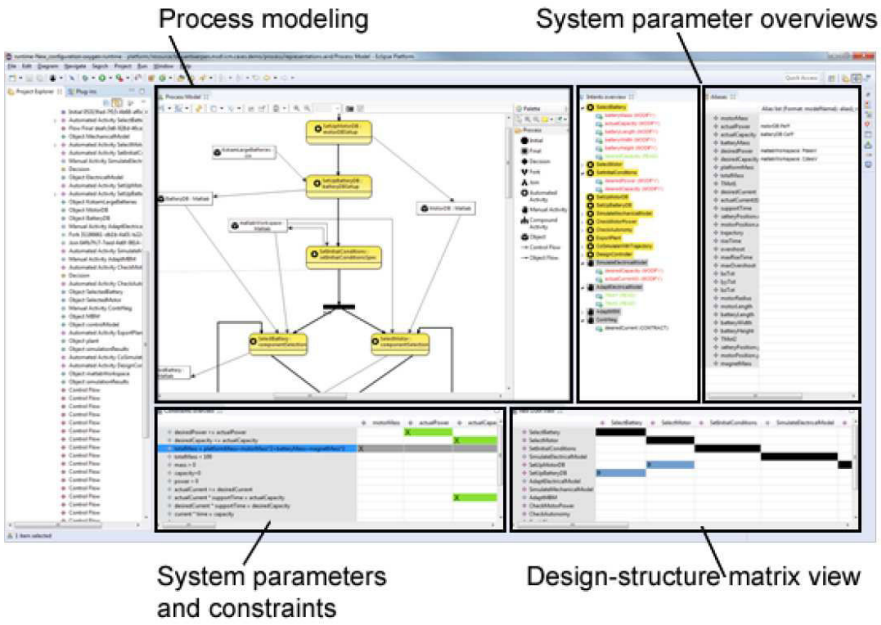



Figure 2-12. PROXIMA framework

Table 2-6. PROXIMA framework evaluation

| Criteria | **Product knowledge** | **Process knowledge** | **Dynamic Exchange** | **Tools interoperability** | **Conflict resolution** | **Consistency check** |
|---|---|---|---|---|---|---|
| Evaluation | The product knowledge consists of system parameters and constraints | Process knowledge is saved in the DSM view | No dynamic exchange is proposed between disciplinary engineers | The interoperability is not considered in this approach | A process optimization algorithm is used to avoid conflicts | The consistency check is realized using the process modeling |

### 2.4.3 Knowledge Configuration Model (KCM)

KCM is developed with the aim of managing knowledge by using configurations synchronized with expert models (Figure 2-13). KCM focuses on the couple product-simulation. The considered knowledge consists of parameters and expert rules organized in Information Core Entity (ICE). ICEs are used by disciplinary designers and conflicts are detected between them (Badin 2011). This methodology was tested in ADN project and can considerably assist in concurrent engineering (Alliance des Données Numériques in French, which means digital data alliance). However, in our case, the lack of process knowledge will affect reuse and the connection with MDO (Table 2-7).

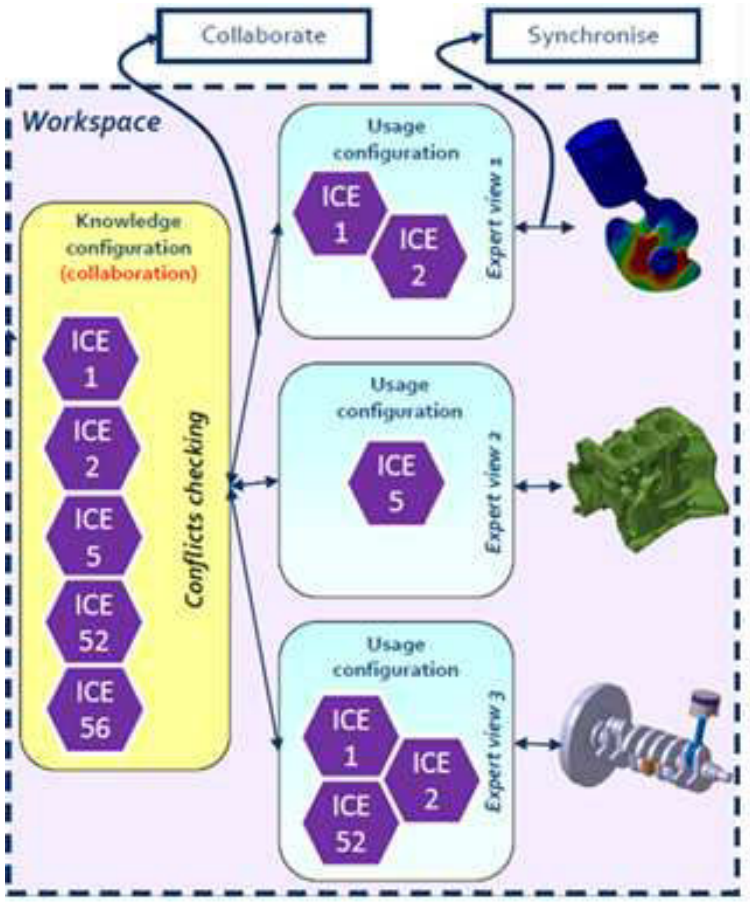



Figure 2-13. Knowledge Management Model

Table 2-7. KCM framework evaluation

| Criteria | **Product knowledge** | **Process knowledge** | **Dynamic Exchange** | **Tools interoperability** | **Conflict resolution** | **Consistency check** |
|---|---|---|---|---|---|---|

| | | | | | | |
|---|---|---|---|---|---|---|
| Evaluation | Parameters and rules organized in ICEs | No process knowledge | Only interdisciplinary dynamic exchange is possible | SOAP connection between engineering tools | Conflicts are detected when parameters does not respect constraints | The consistency is checked via ICE instances |

### 2.4.4 Constraint Linking Bridge (COLIBRI)

Constraints can be used to support cooperative work in consistency management. COLIBRI is completely dedicated to constraints linking parameters between heterogeneous DE models (Figure 2-14). This model allows the connection between parameters encapsulated in heterogeneous models through the integration of constraints, taking into account the structure of the product. In the constraint modelling method, the constraints are classified into mechanical and electrical domains respectively. Based on this classification, the disciplinary and the cross-disciplinary constraints of a mechatronic system can be managed (Kleiner, Anderl et al. 2003). This approaches are interesting to manage inconsistency but does manage exchange between engineers (Table 2-8).

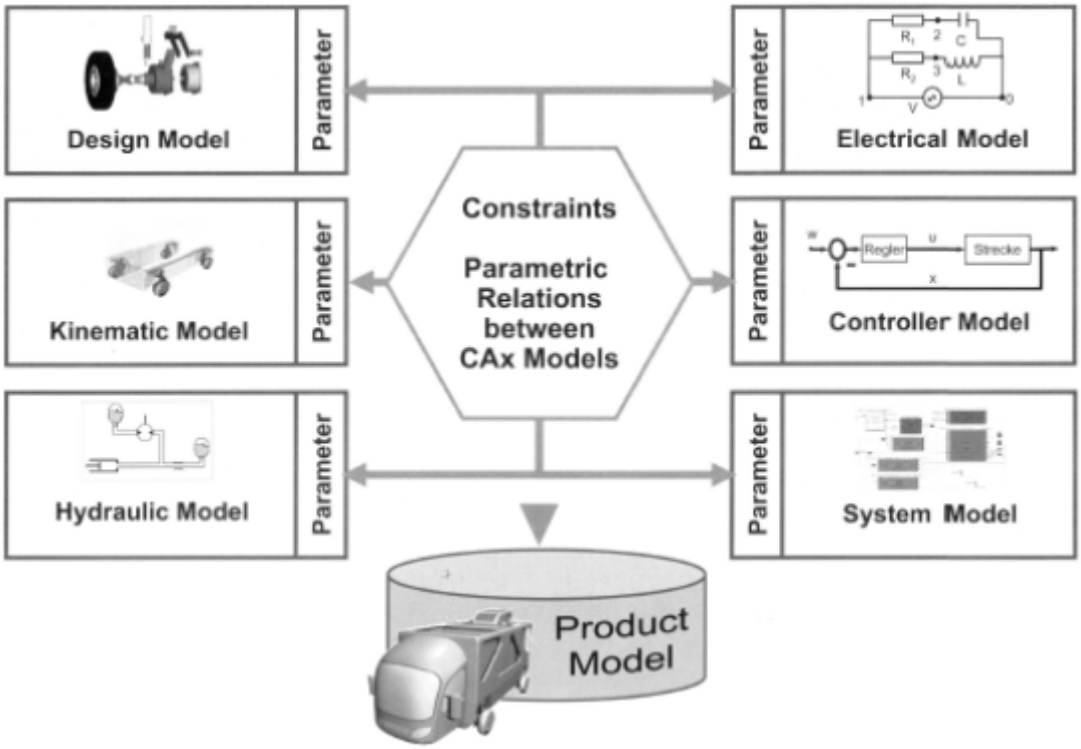



Figure 2-14. COLIBRI concept (Kleiner, Anderl et al. 2003)

Table 2-8. COLIBRI framework evaluation

| Criteria | **Product knowledge** | **Process knowledge** | **Dynamic Exchange** | **Tools interoperability** | **Conflict resolution** | **Consistency check** |
|---|---|---|---|---|---|---|
| Evaluation | Parameters and relations between DE models | No process knowledge is provided | No dynamic exchange is proposed | The interoperability is managed via constraints between tools | There is no conflict resolution approach | The consistency is managed using the constrints |

## 2.5 MDO support solutions

### 2.5.1 Design and Engineering Engine (DEE)

This system, called the design and engineering engine (DEE), is based on the KBE technique. The heart of the DEE is the multi-model generator (MMG). The MMG defines high level primitives (HLPs) to capture elements of similarity among very different configurations and using them as the parametrized modules for geometric modeling (Figure 2-15). The engineer therefore does not have to do any geometry modeling, only choose the HLPs, which are then assembled together automatically (Rocca and L. Van Tooren 2009). The presented tool is said to be effective and time-saving when performing MDO, partly because the geometry is produced much faster than with a CAD tool and partly because the pre-processing activities required to feed the various analysis systems in the MDO process can be largely or fully automated (Table 2-9).

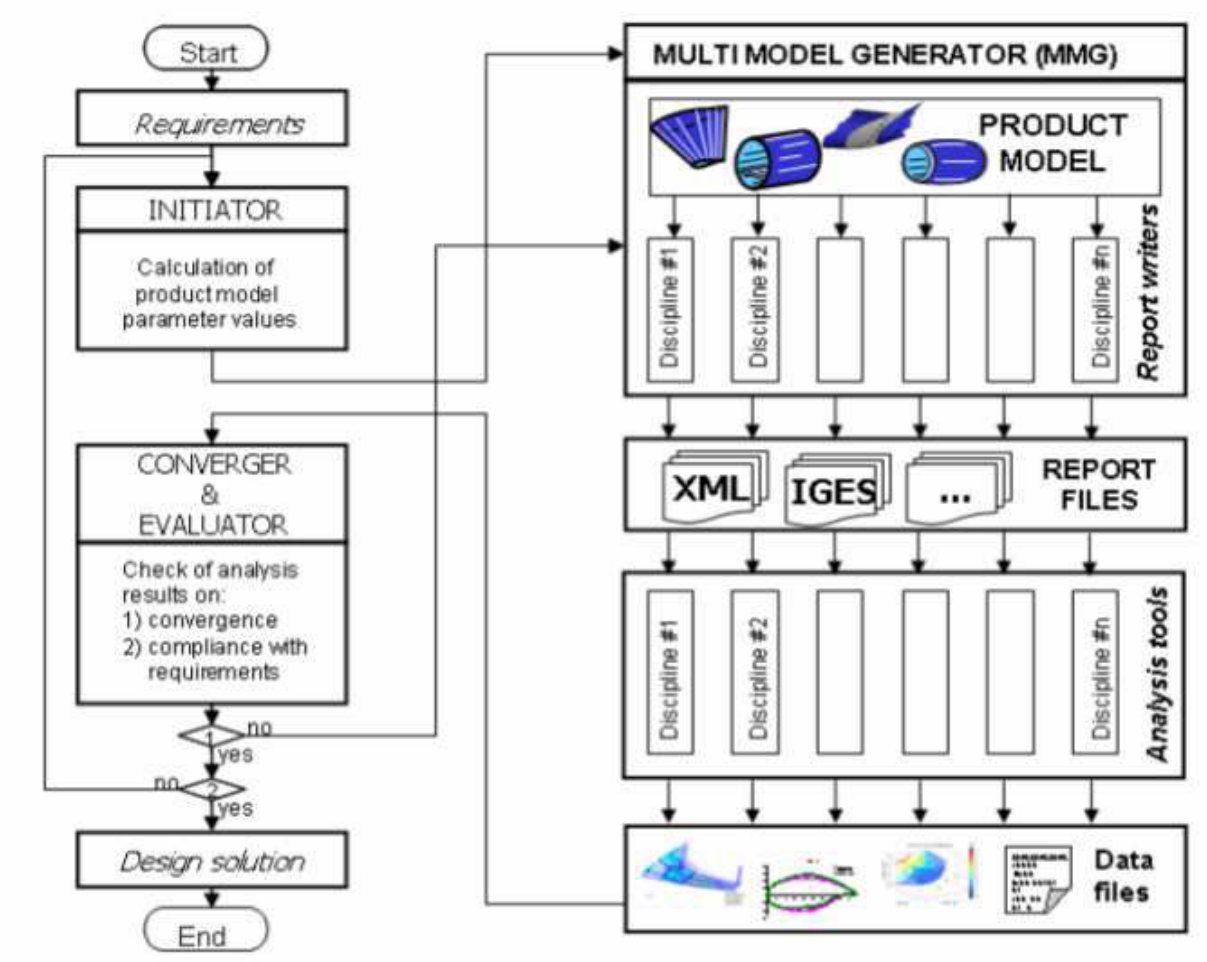



Figure 2-15. Design and Engineering Engine process (La Rocca 2012)

Table 2-9. DEE framework evaluation

| Criteria | **Product knowledge** | **Process knowledge** | **Dynamic Exchange** | **Tools interoperability** | **Conflict resolution** | **Consistency check** |
|---|---|---|---|---|---|---|
| Evaluation | Parameters used in MMG process | The process between the DE models | No dynamic exchange is proposed | The interoperability is managed by the MDO tool | Not considered | The consistency is considered only during the optimization |

### 2.5.2 FabK framework

FabK methodology is composed of 3 phases: Expert phase, Design phase processing phase, Validation and knowledge acquisition phase (Figure 2-16). The feedbacks from late phases are integrated into the design processing phase utilizing a KBE application (Toussaint, Demoly et

al. 2010). Product and process knowledge are considered in this methodology, nevertheless, there is no collaborative environment for dynamic exchange between engineers (Table 2-10).

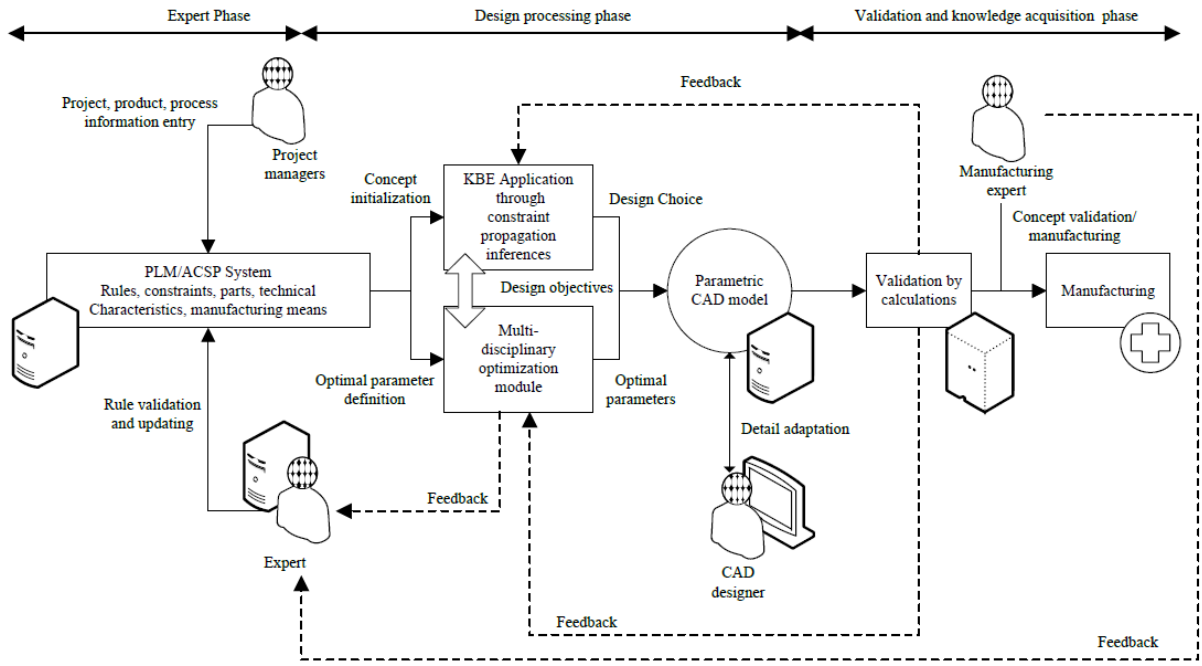



Figure 2-16. FabK Methodology (Toussaint, Demoly et al. 2010)

Table 2-10. FabK framework evaluation

| Criteria | **Product knowledge** | **Process knowledge** | **Dynamic Exchange** | **Tools interoperability** | **Conflict resolution** | **Consistency check** |
|---|---|---|---|---|---|---|
| Evaluation | Parameters and rules | The MDO process is considered in this approach | No dynamic exchange | The interoperability is not considered | There is no conflict resolution system | The consistency is checked using rules |

### 2.5.3 KADMOS framework

KADMOS (Knowledge and graph based Agile Design for Multidisciplinary Optimization Systems) platform propose a methodology to automatically generate MDO architecture. This approach was proposed in the context of AGILE EU project and an open access software was developed (Ciampa and Nagel 2017). This framework supports the formal specification the MDO problem using graph based system (Figure 2-17). It helps considerably disciplinary engineers to set up MDO architecture and enable system engineers to practice tradeoffs, however, it does not support the collaborative phase between engineers (Table 2-11).

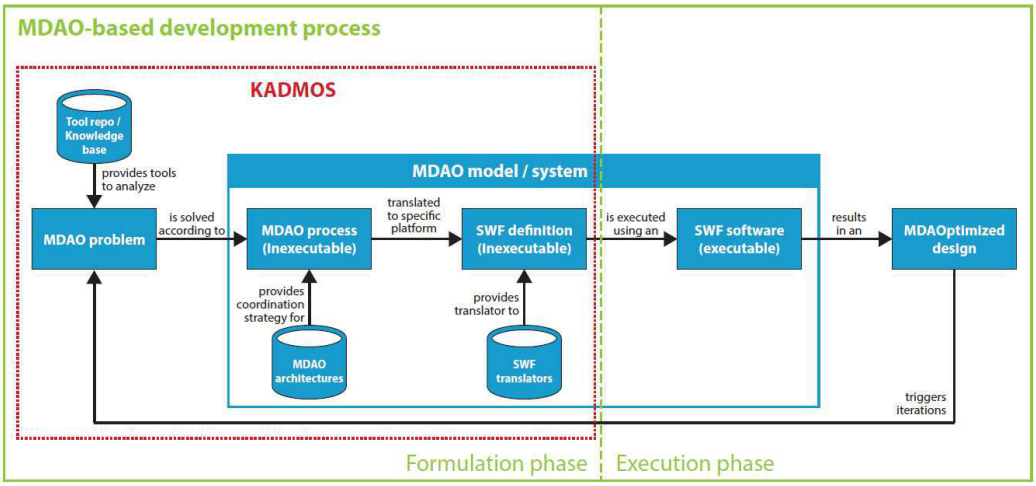



Figure 2-17. KADMOS methodology overview (van Gent, Ciampa et al. 2017)

Table 2-11. KADMOS framework evaluation

| Criteria | **Product knowledge** | **Process knowledge** | **Dynamic Exchange** | **Tools interoperability** | **Conflict resolution** | **Consistency check** |
|---|---|---|---|---|---|---|
| Evaluation | Parameters and MDO results | Process between DE tools and MDO architectures | Not considered | Managed by MDO tool | Not considered | Managed by the MDO tool |

## 2.6 Conclusion and work positioning

Many interesting models are proposed to support SE, DE, and MDO. Based on our criteria we will choose a methodology to apply it in the mechatronic design context. Our objective is to start with a methodology fulfilling a part of our criteria and extend it to reach the connection between SE, DE, and MDO. The Table 2-12 summarizes the state of the art.

Table 2-12. Summary of frameworks evaluation

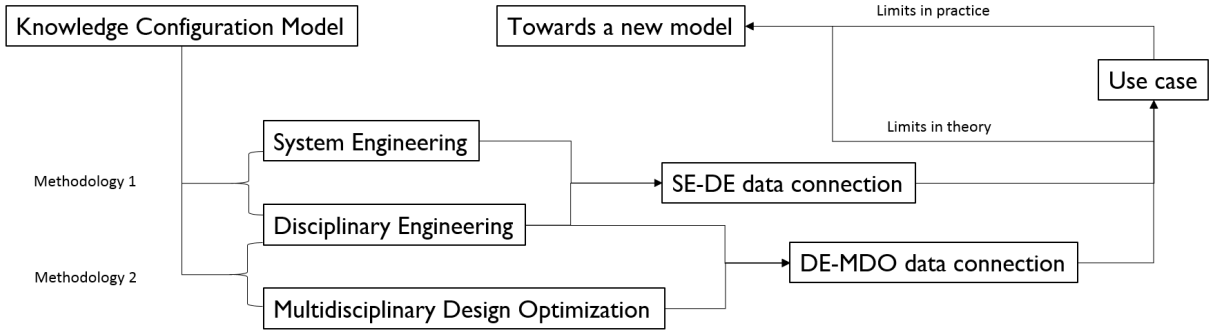
| Criteria | **Product knowledge** | **Process knowledge** | **Dynamic Exchange** | **Tools interoperability** | **Conflict resolution** | **Consistency check** |
|---|---|---|---|---|---|---|
| SysDICE | Yes | No | No | Partial | No | Yes |
| SLIM | Yes | No | Partial | Yes | No | Yes |
| PIDO | No | Yes | No | Yes | No | Partial |
| Multiview | Partial | Yes | No | Partial | No | Yes |
| PROXIMA | Yes | Yes | No | Partial | No | Yes |

| | | | | | | |
|---|---|---|---|---|---|---|
| KCM | Yes | No | Yes | Yes | Partial | Yes |
| COLIBRI | Partial | No | Partial | Partial | No | Yes |
| MMG | Yes | Yes | No | Yes | Partial | No |
| FabK | Yes | Yes | Partial | Partial | No | No |
| KADMOS | Yes | Yes | No | Yes | No | Partial |

KCM is the most eligible model with these criteria, offering more flexibility than the other solutions in terms of conflict resolution and consistency management. In the next chapter, we will define a methodology for KCM in order to enable the continuity between SE, DE, and MDO in mechatronic design.

# Chapter 3 Knowledge Configuration Model applied to mechatronic design

*In this chapter, we define a methodology for KCM in order to apply it in mechatronic design. This methodology consists of SE-DE and DE-MDO data connections. An application case is presented to evaluate KCM in practice. Finally, the practical and theoretical limits of KCM are listed.*



## 3.1 KCM principle

### 3.1.1 Configuration management

To improve the inter-operability between disciplinary engineering tools, numerous product models are created to support the PLM approach through data and information management (Function-Behavior-Structure, Core Product Model, Product Process Organization…) (Zheng, Bricogne et al. 2014). However, these models do not focus on the management of knowledge embedded in CAD and CAE models. In this context, KCM was proposed to manage the knowledge encapsulated in disciplinary models (Badin 2011). It ensures the coherence of parameters through several product design and simulation activities in a collaborative engineering context by using the concept of Configuration Management (CM).

CM is a managerial discipline that aims at providing consistency and accuracy of product knowledge throughout its lifecycle and for the same purpose it is being used in different extents in most of the organizations (Niknam and Ovtcharova 2013). CM is defined in ISO 10007:2003 releases:

- Configuration: interrelated functional and physical characteristics of a product defined in product configuration information.
- Product configuration information: requirements for the design, implementation, verification and support of a product.

Product configuration ensures the consistency through the design process by managing configuration information organized into “Configuration Items”. All the configuration items must be coherent over all the design process following the evolution into an iterative process and keep traceability of any changes. Configuration management can be particularly interesting to manage parameters and rules used in several CAD/CAE models.

### 3.1.2 KCM overview

The first objective of KCM is to enable designers to use parameters consistently in collaborative design. In fact, the lack of communication among different steps of the design cycle causes consistency problems in parameters. In CE, all participants need to access all relevant up-to-date product information. This methodology is based on the concept of crucial knowledge to keep the design models consistent with each other (Monticolo, Badin et al. 2015). Crucial

knowledge refers to the sufficient and necessary knowledge that need to be shared for the activity of a company (Saad and Chakhar 2009). Here, the knowledge consists of contextualized parameters and rules that are critical for the collaborative product development. Therefore, this crucial knowledge is structured and has its own lifecycle according to the guiding GAMETH framework (Grundstein and Rosenthal-Sabroux 2004). This framework defines a cycle with four facets which are: identify, formalize, value and update.

The principle is the capitalization of critical knowledge extracted from different expert models, into an abstract generic information entity called an "Information Core Entity" (ICE). The ICEs are grouped in a "Knowledge Configuration" (KC). Then each stakeholder instantiates the necessary ICEs in his "User Configuration" (UC). This principle is illustrated in Figure 3-1 and further explained in the next section.

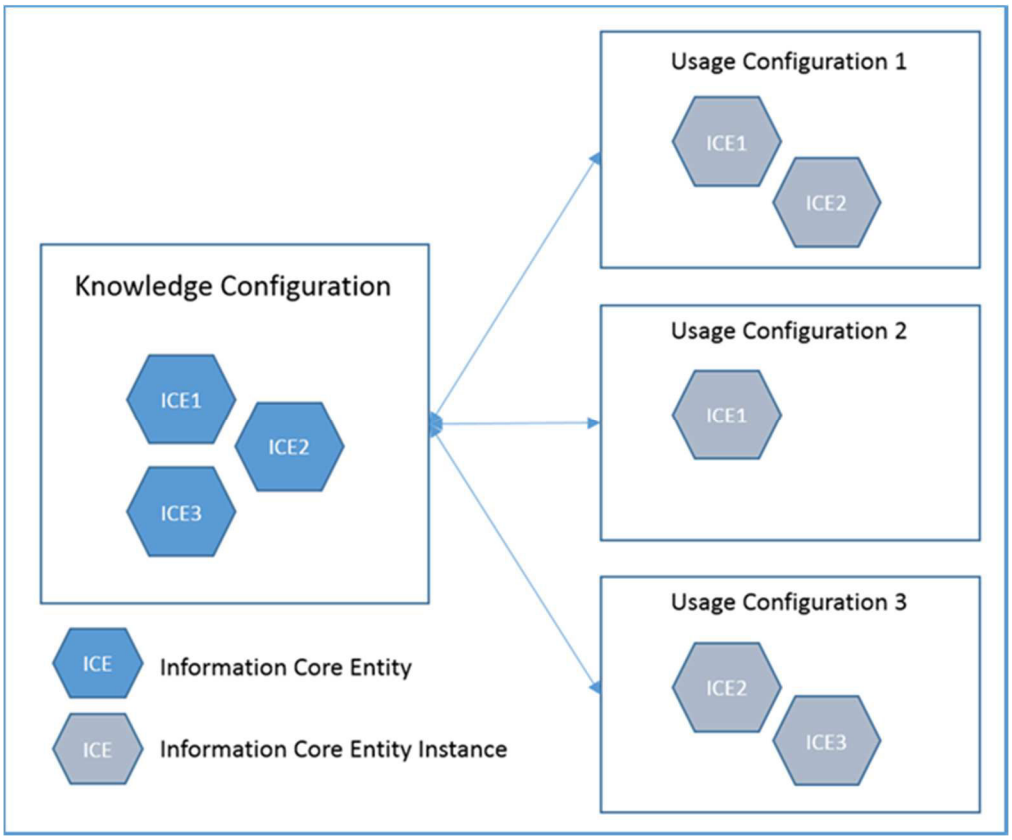



Figure 3-1 Knowledge Configuration Model principle

## 3.2 Main concepts

### 3.2.1 Information Core Entity (ICE)

ICE is an indecomposable generic entity that can capitalize crucial data extracted from disciplinary models in order to move from the data state to the information state (Badin, Chamoret et al. 2011). It is composed of parameters and generic constraints. These constraints include:

- Mathematic relations
- Expert rules (if ... then ...)
- Boundary conditions (min ,max ,default)
- Discrete values table

The creation of ICEs results from an analysis to identify data and information crucial to capitalize and share in the design process, especially between disciplinary models. The set of ICEs forms a dynamic knowledge base which is enriched as the project evolves.

### 3.2.2 Usage Configuration (UC)

Each user makes a selection in the ICE database to retain only those they need for their activity. Then, they instantiate this selection in a Usage Configuration (UC) and synchronize it with a disciplinary model. Thus, a UC is constituted of ICE instances and represents the knowledge encapsulated in a disciplinary model. The notion of configuration refers to all the services of version management, management changes and consistency management in the UC. Therefore, in a UC, a user can back up multiple versions based on changing parameter values or adding or deleting instances of ICEs. The interest of configurations is to have a mode of homogeneous representation of knowledge that can be used in different activities of the design process. It is through them that the coherence of shared knowledge is ensured because it is very difficult to directly compare several disciplinary models with each other that are using different heterogeneous tools.

### 3.2.3 Knowledge Configuration (KC)

KC contains the ICEs and UCs that will be used in the collaboration. It can be considered as the dashboard of the collaboration. UCs are compared in KC to give users information on the existence of possible conflicts. This global configuration ensures the management and collaboration of all the UCs and represents all the activities of a project milestone. ICEs in KC are either directly instantiated from the generic knowledge base or recovered from other previous

configurations. When the collaboration starts, the project manager can visualize the existing conflicts between the different instances ICEs of the published UCs.

### 3.2.4 KCM metamodel

The KCM UML metamodel is shown in Figure 3-2. The link between the various concepts. KCM manages:

- Technical data: the parameters and expert rules extracted from disciplinary models.
- Information: the data identified, structured and organized into a specific entity to construct a technical and generic product information baseline.
- Product Knowledge: a set of technical product information entities instantiated from the baseline in a configuration used in specific design or simulation activity. This configuration is synchronized with a specific CAD or CAE model.

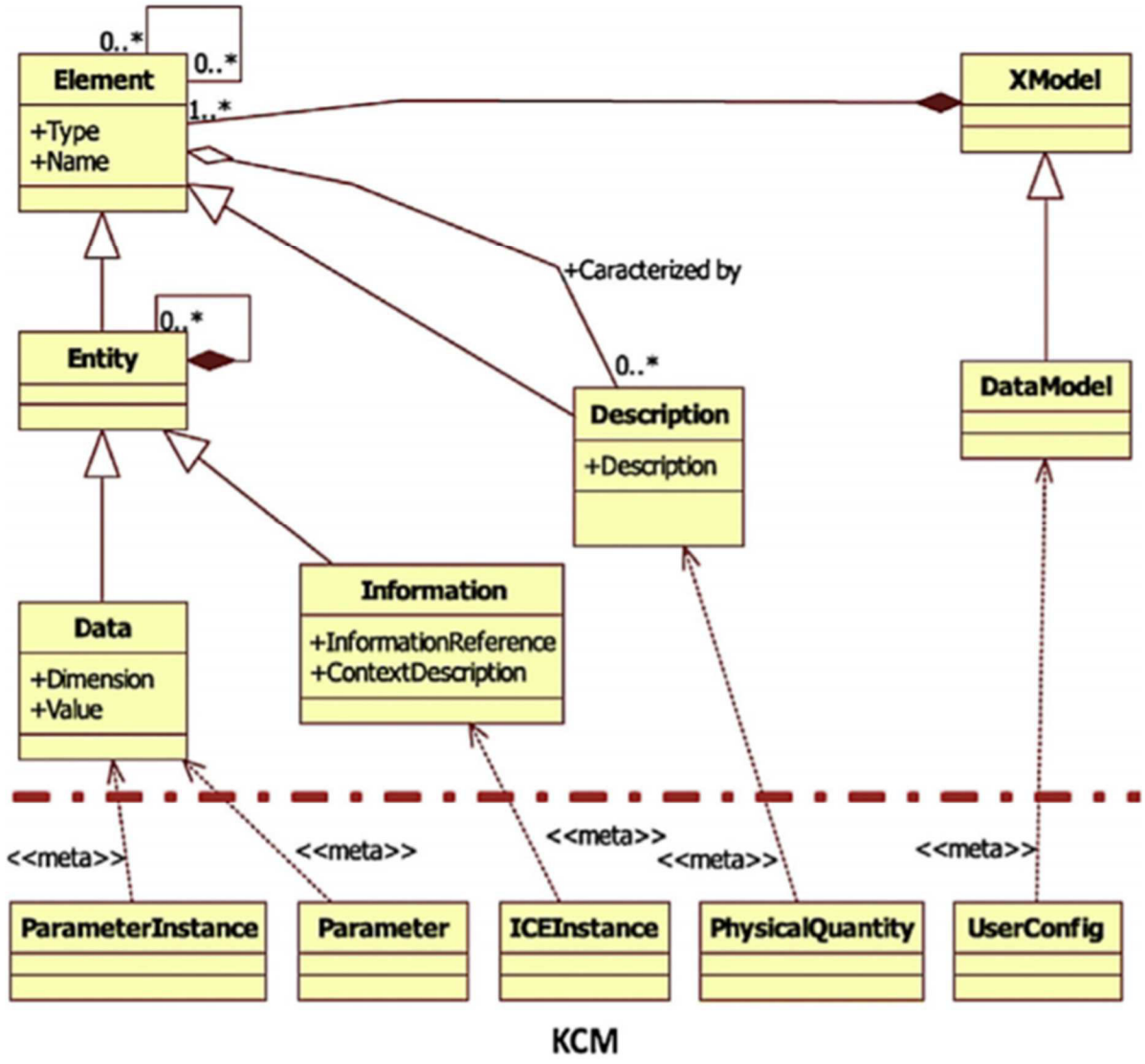



Figure 3-2 UML meta-model of KCM (Monticolo, Badin et al. 2015)

## 3.3 KCM use

### 3.3.1 Consistency checking

Each user working on very different expert models (design and simulation domain, different tools, different components, etc.), exports data from multiple product architectures into their configurations simultaneously. The consistency management between configurations can warn users if conflicts occur between data shared by several expert models. Two kinds of consistency are considered (Badin, Monticolo et al. 2011):

- Consistency in User Configuration level: parameters must respect all the rules defined in the UC

- Consistency in Knowledge Configuration level: Instances of the same ICE must contain the same values.

### 3.3.2 PLM connection

In the ADN project, a PLM connector was realized for KCM (Penciuc, Durupt et al. 2014). The connector was designed as an independent web application which is accessible independently of the information systems considered. The Windchill and ADN environments were chosen to illustrate the implementation of the PLM connector. The connector architecture is flexible enough to allow portability and interoperability with external applications. The communication is handled by the SOAPClient2.4 component, which is a JavaScript library implementing the SOAP1.1 protocol. The connection is illustrated in Figure 3-3.

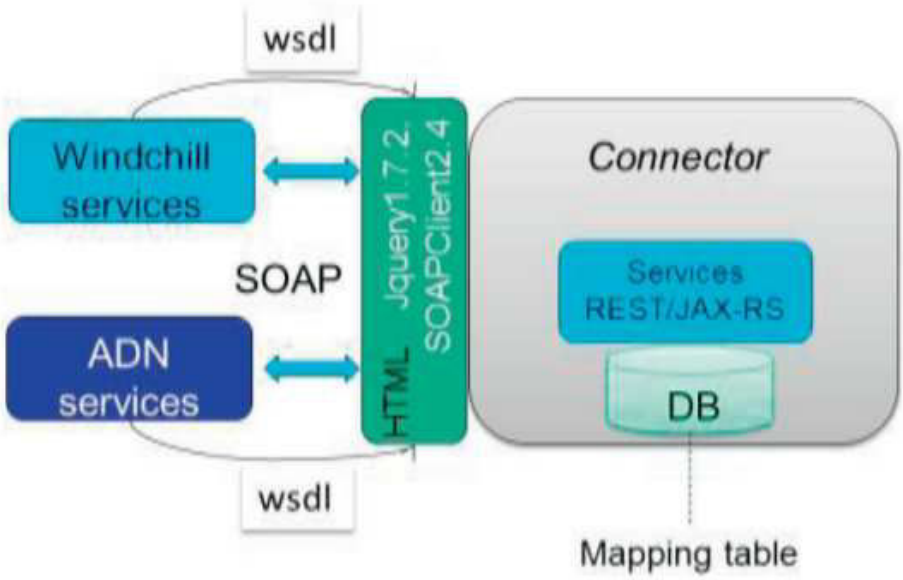



Figure 3-3. KCM-PLM connection (Penciuc, Durupt et al. 2014)

### 3.3.3 The need for a mechatronic methodology

This chapter aims to develop an approach to use KCM in mechatronic design. To provide an answer for both research questions, we focus first on the connection between SE and DE with the necessary steps to formalize the knowledge for the system and disciplinary engineers. Second, we use KCM to integrate MDO and DE by means of synchronization between MDO tools and KCM.

## 3.4 SE-DE connection

Our contribution here is to define a methodology in order to connect SE and DE using KCM. After analyzing the collaboration process of PSA and Valéo we defined actors and steps to transfer the critical knowledge from system engineers to disciplinary engineers.

### 3.4.1 Actors

The mechatronic system is a combination of mechanical, electronic/electrical and software technology, a modification in any discipline will influence the others. The product model of the mechatronic system should evolve dynamically according to the progress of the design process. Therefore, KCM can clearly help the designers in incorporating changes during the process. However, KBE is not limited to its technological approach, it includes a sociotechnical approach by analyzing the actors and their roles in knowledge management (Arduin et al. (2013)).

We identified three main roles in our approach:

- Knowledge Expert
- Locating and characterizing the critical knowledge needed to solve the problem
- Capitalizing the knowledge inside ICEs
- System Engineer
- Converting the descriptive requirements into bounds and constraints in ICEs
- Creating User Configurations and manage the collaboration
- Disciplinary Engineer
- Filling the parameters of ICE instances in his configuration.
- Transferring the parameters' values from other Users if needed

### 3.4.2 Methodology

The different steps are illustrated in **Erreur ! Source du renvoi introuvable.**:

1. The critical knowledge is captured and organized in ICEs by the Knowledge Expert. The Knowledge Expert should have a multidisciplinary background to collect the knowledge from different designers.
2. System Engineers add the requirements of the project to ICEs. This can be achieved by adding new parameters or defining constraints for the existing parameters. This step is important because it represents the link between SE and DE teams.

3. The manager creates Usage Configurations for the collaboration. Each disciplinary engineering instantiates needed ICEs then they can freely collaborate.
4. Each user can send results to other collaborators by publishing ICE instance (ICE_A)
5. Each user can receive results from other collaborators by updating the common ICE instance (ICE_B)
6. There is also the possibility to work on the same ICE and check the results of two different users for validation purposes (ICE_C)

This approach is flexible and offers a collaborative environment for users from different steps of the design cycle so they can collaborate efficiently. Generally, the multidisciplinary collaboration is error-prone with difficult decision making but the dynamicity offered by this approach can handle these problems. In addition, the work made by the different stakeholders can be adapted and reused for future collaborations.

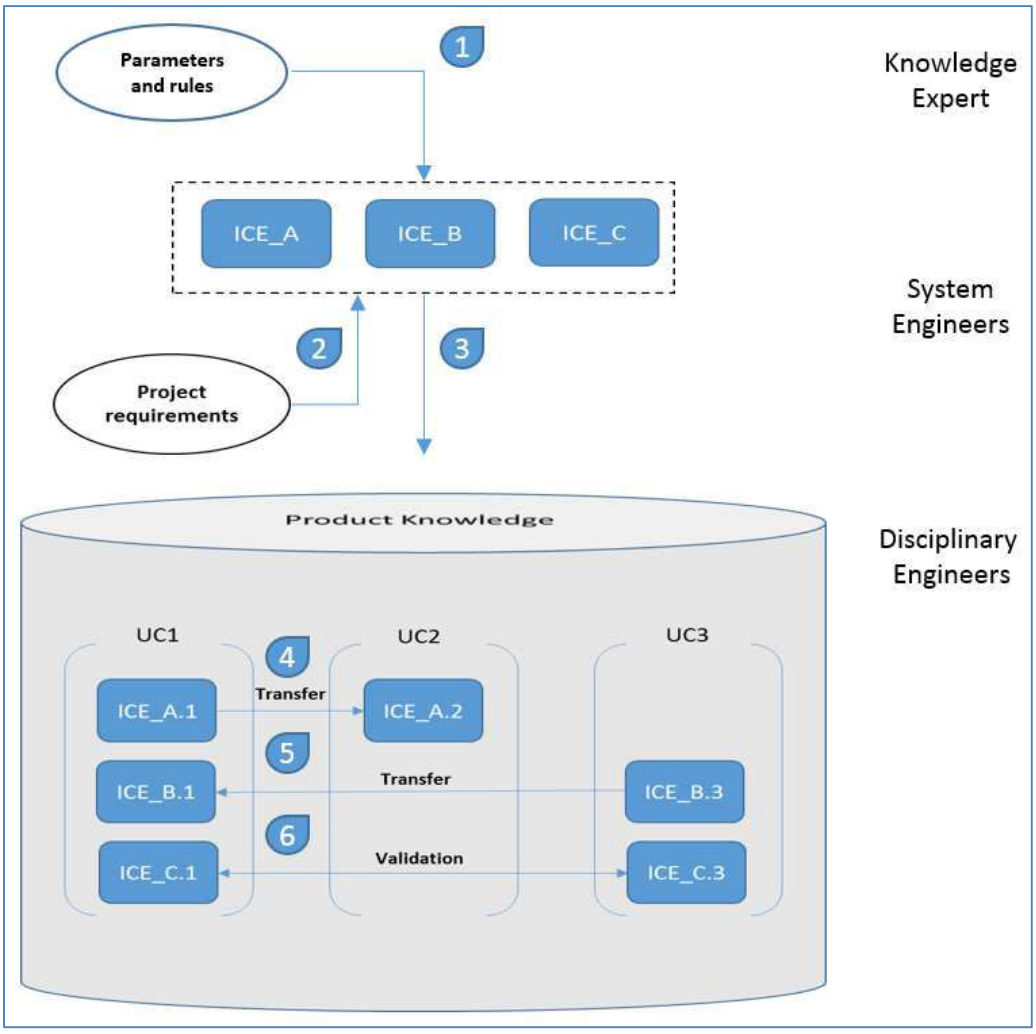



Figure 3-4. SE-DE connection using KCM

The key element to apply KCM in mechatronic design is the organization of roles during the process. The structure of KCM allows the exchange between users but there is a lack of decisions traceability to improve the reuse phase. In the next section, we will focus on the MDO part.

## 3.5 DE-MDO connection

Our contribution here lies in defining a connector between MDO tool and KCM to give the MDO user the possibility to define the MDO problem in KCM and synchronize the results with the other stakeholders.

### 3.5.1 MDO user

The MDO User is an expert in optimization, he knows the optimization techniques and how to study the system under analysis in order to select the best optimization strategy. His knowledge also covers the techniques for design of experiment and system analysis. The MDO User is responsible for the creation of the multidisciplinary workflow, of the optimization plan, and for making the plan available to the other users. The definition of the optimization plan includes: definition of goals such as objectives and constraints, definition of the design space, and definition of the optimization strategies (Hiriyannaiah and Mocko 2008). Engineers use the results of the optimization to validate their assumptions and to examine different alternatives. The MDO user needs, therefore, an environment to exchange directly with other engineers and update his results.

### 3.5.2 MDO-KCM connector

Our methodology consists in defining an environment to manage MDO and collaborative design. To establish an MDO problem in an industrial context, the intervention of several multidisciplinary designers is crucial. This will be possible by using KCM and by correctly defining the UCs for each designer. Then a connection is made between KCM parameters and MDO tool. Finally, the results of the MDO can be validated in the KCM environment by the experts. A standard example is presented in Figure 3-5 to illustrate the methodology.

1. A Knowledge Configuration is first created to manage the collaboration. The KC manager defines the different UCs and the needed ICEs. In this level, the parameters are defined with their units and constraints as explained in the previous section. The values of the parameters will be defined later in the UCs.

2. Each designer instantiates in his UC environment the needed ICEs. The UC1_MDO should contain all the ICEs necessary to define the MDO problem. Then MDO designer takes the values of unknown parameters from the other users. In this way, we collaboratively define the MDO problem and we provide interaction between different users.
3. The parameters of the UC1_MDO are sent to the MDO tool via a connector. A mapping is realized between KBE baseline and MDO data. The ICE parameters bounds are used in MDO to define optimization objectives and constraints. Then the models' parameters of MDA are mapped to ICE parameters values.
4. Optimization is launched with iterations between the optimization algorithm and the models involved in MDO.
5. The MDO results are then mapped to the KBE baseline with a second connector. The data in UC1_MDO is updated with the MDO results.
6. The results saved in UC1_MDO can be used then for validation and experimental verification with other users connected to the KC. Indeed, the MDO becomes a bridge between different disciplines in different levels of the design cycle.

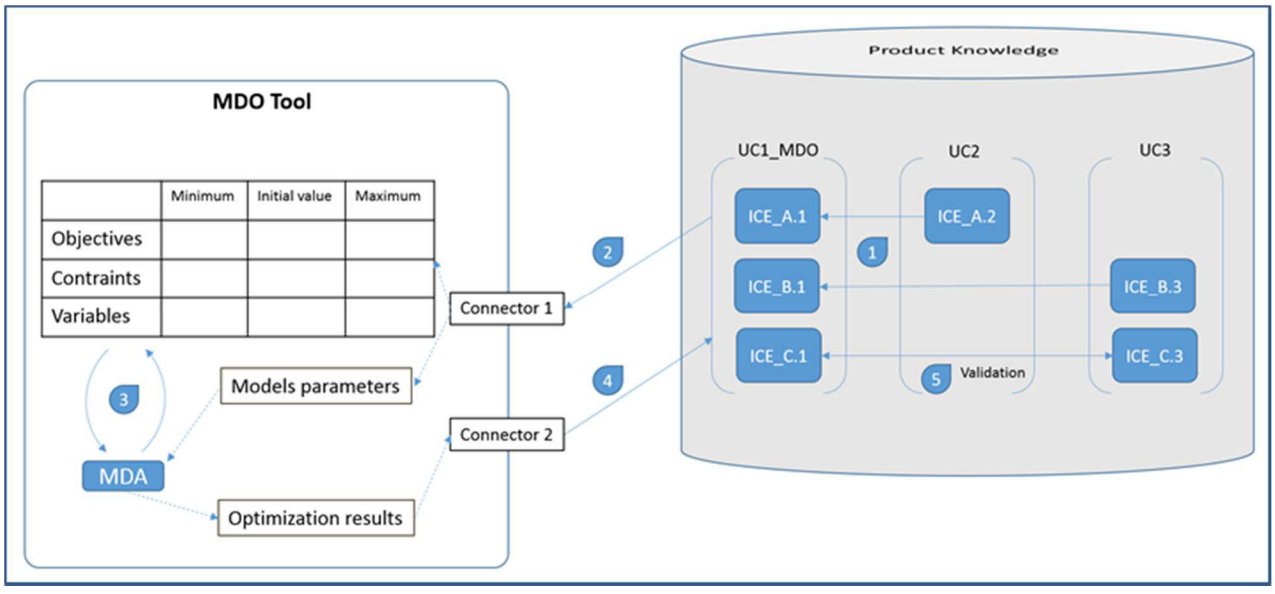



Figure 3-5. MDO - KCM connection

This connection will enable us to collaboratively define an optimization design and to collaboratively validate the results. However, KCM cannot support the automatic creation of an MDO architecture because it does not support process knowledge. Therefore, this methodology can be applied only when the MDO problem does not require a complex architecture.

## 3.6 Use case

To illustrate the methodology, we created a collaborative scenario to optimize the control system of a VDO Siemens ETB. This model is shown in Figure 3-6.


Figure 3-6. Siemens VDO model

### 3.6.1 ETB control system

The control system is a Proportional Integrator controller. It generates a PWM signal to actuate the HBrige. Figure 3-7 represents the model with the control loop. For security measure, a mechanical failsafe system with two springs (main spring and return spring) is used. So when the control system fails, the springs keep the valve at the Limp Home position (in our case 11.5 deg) in order to provide the necessary flow to keep the engine running. A closed-loop model is created using Modelica as shown in Figure 3-8. The optimization of the control system is not easy because the system is highly nonlinear and has a hysteresis behavior as illustrated in Figure 3-9.

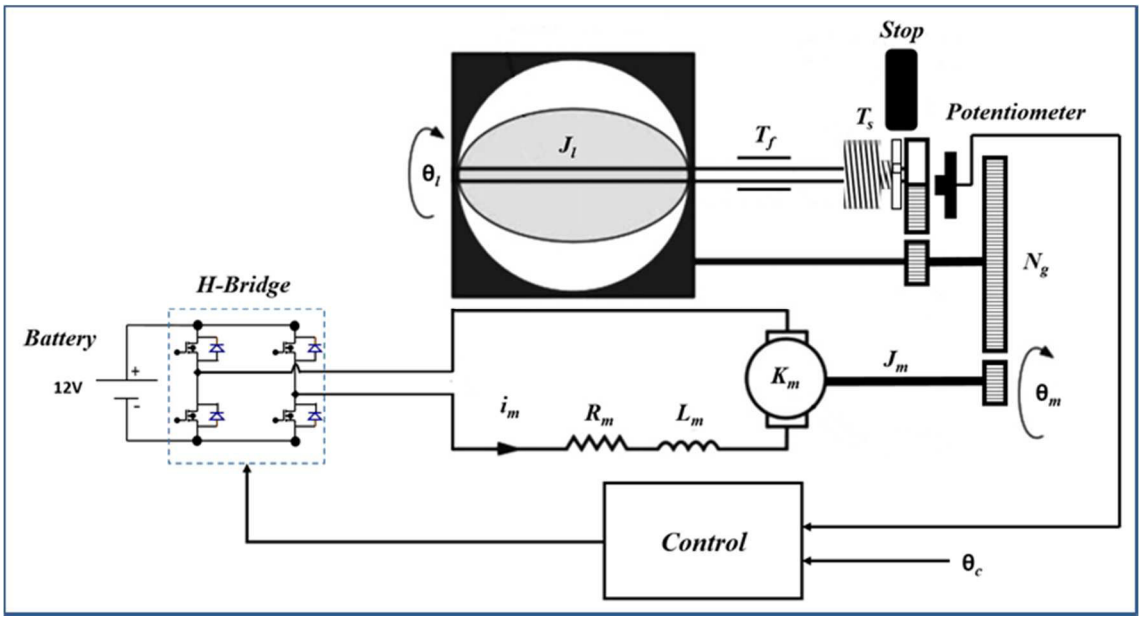



Figure 3-7. ETB architecture in closed loop

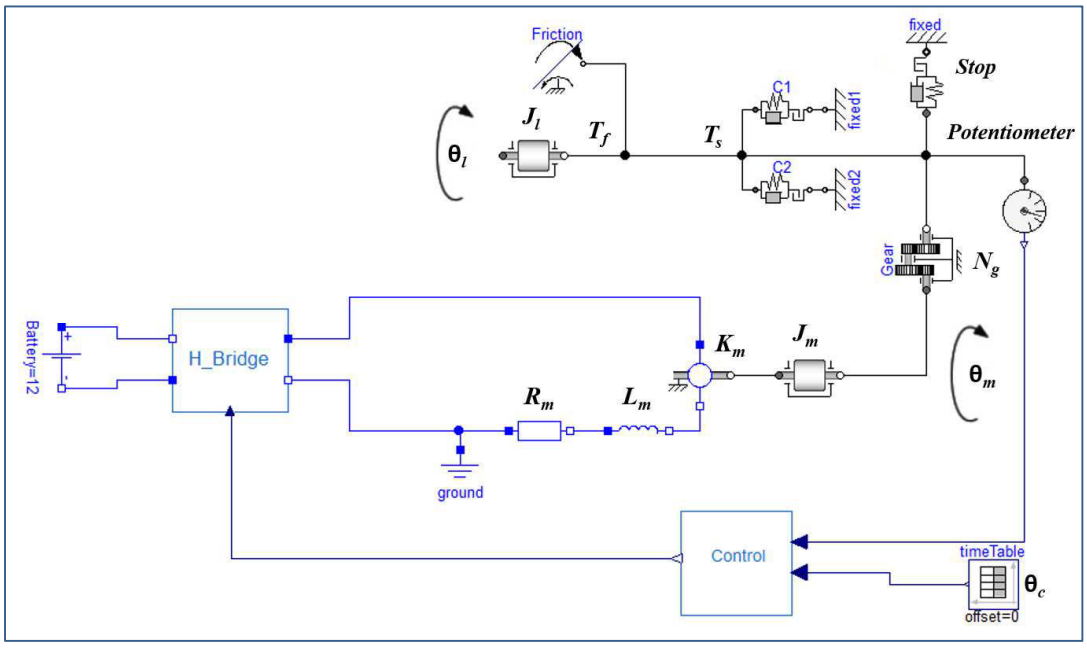



Figure 3-8. Modelica model of the ETB in closed loop

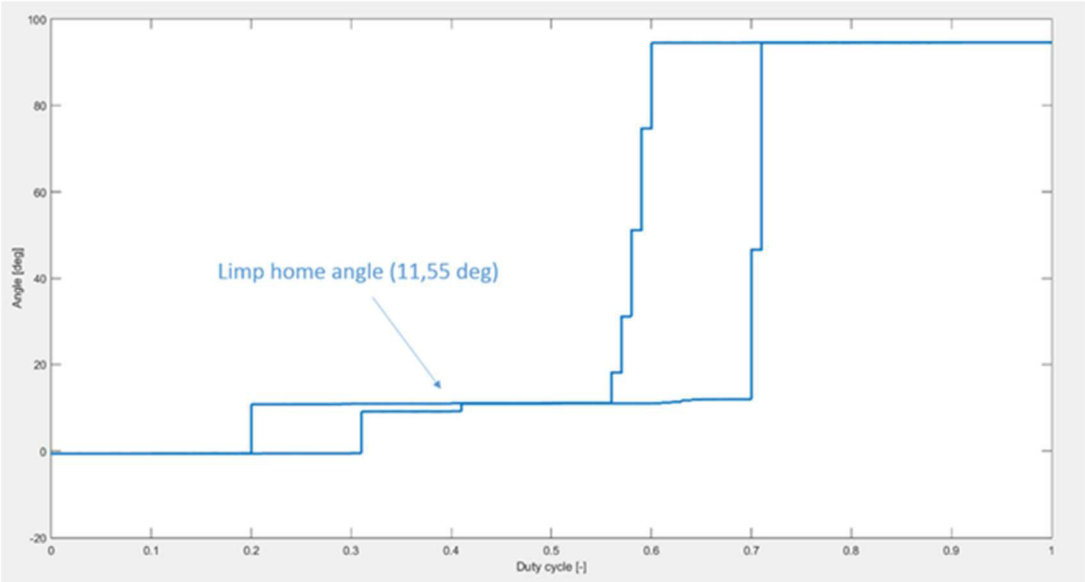



Figure 3-9. Hysteresis behavior of the ETB

### 3.6.2 Collaborative scenario

The throttle angle has to be precisely maintained based on the driver and other system requirements to provide an enhanced throttle response and drivability (Ashok et al. (2017)). The task is to design a PI control system for Siemens VDO ETB with the following requirements:

- Minimize the rise time (11.5 deg to 90 deg) : Lower than 200 ms.
- Minimize the return time (11.5 deg to 90 deg) : lower than 150 ms
- Overshoot lower than 5 deg.
- Static Error lower than 0.5 deg.

The first phase consists of applying SE-DE connection. The first issue that we faced is to organize correctly the parameters in ICEs. This step is important because we cannot change Ices when the collaboration begin. We choose to organize parameters All the parameters using system engineers decomposition (ICE_Control, ICE_Valve, ICE_Motor). Then parameters coming directly from the requirements are added to ICE_Requirements. For specific requirements, constraints are added to initial parameters. The list of all the parameters is given in Table 3-1.

Table 3-1. List of parameters and ICEs

| Parameters | Unit | Description | Min | Max |
|---|---|---|---|---|
| ICE_Requirements | | | | |
| Rise | ms | Rise time from 11.5deg to 90deg | 0 | 200 |

| | | | | |
|---|---|---|---|---|
| Return | ms | Return time from 90deg to 11.5deg | 0 | 150 |
| S_Error | deg | Signal static error | 0 | 0.2 |
| Overshoot | deg | Absolute signal overshoot | 0 | 5 |
| ICE_Control | | | | |
| P | - | Proportional gain | 0 | 0.30 |
| I | - | Integrator gain | 0 | 0.25 |
| ICE_Motor | | | | |
| $K_m$ | N.mm/A | Motor constant | | |
| $R_m$ | Ohm | Motor resistance | | |
| $J_m$ | $Kg.m^2$ | Motor inertia | | |
| $F_m$ | N.mm | Friction | | |
| ICE_Valve | | | | |
| $J_l$ | $Kg.m^2$ | Valve inertia | | |
| $N_g$ | - | Gearbox ratio | | |
| $C_1$ | N.m/rad | Main spring stiffness | | |
| $C_2$ | N.m/rad | Return spring stiffness | | |

KARREN software is platform based on KCM and we used this platform for this case study. For more details about the platform, readers can refer to the patent (Navarro, Badin et al. 2018). We first created the ICEs and the parameters listed in Table 3-1. Then, we created UCs for the users (UC1_MDO, UC2_Test, UC3_Identification) as illustrated in Figure 3-10.

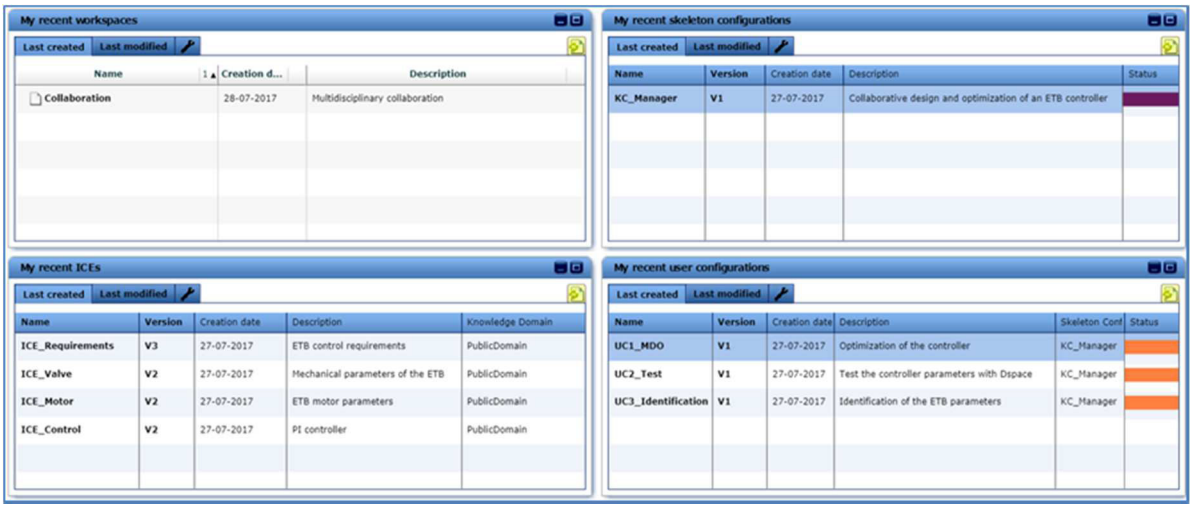



Figure 3-10. KARREN platform dashboard

For MDO, a connector between Karren tool and Isight tool is implemented. As illustrated in Figure 3-11, the Karren connector is placed as an input to map the necessary values for the optimization problem definition. After the optimization loop, a second connector collects the results and sends them to the baseline. This connection allows the integration of the MDO in the design process with a clear mapping between MDO parameters and KCM parameters

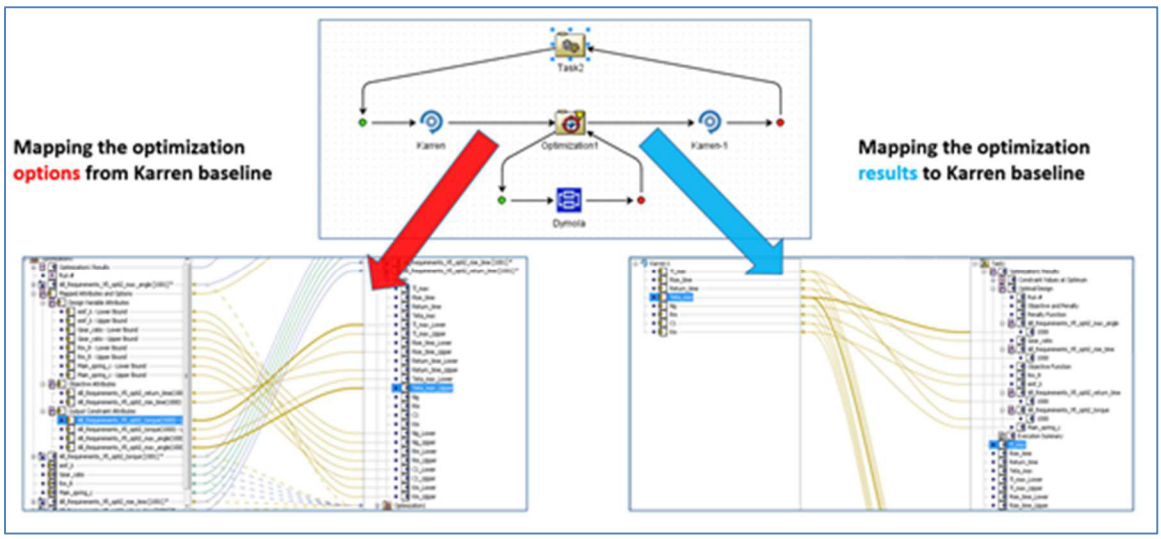



Figure 3-11. Karren - Isight connection

The sequential diagram in Figure 3-12 explains the collaboration between the different users. The collaboration starts with the KC Manager who defines the different users and fills the bounds of ICE_Requirements. These bounds will be used later as optimization objectives and constraints. Then each user instantiates the ICEs that he will need. First, the Identification user realizes the identification of the ETB and fills the values of the parameters found in ICE_Motor.3 and ICE_Valve.3. The MDO user takes this values and has now all the necessary data to

start the optimization. This data is sent to the MDO tool via the input connector and the optimization is launched. The results of the optimization are then sent back via the output connector and saved in ICE_Requirements.1 and ICE_Control.1. The manager has now access to the optimization results and waits for the validation. To validate the results of the controller parameters, ICE_Control is sent to the Tester that will use a test bench based on dSPACE card (control system) and the Siemens VDO ETB to check the requirements. Then, he fills the test values in ICE_Requirements.2 and finally the validation can be made by the Manager.

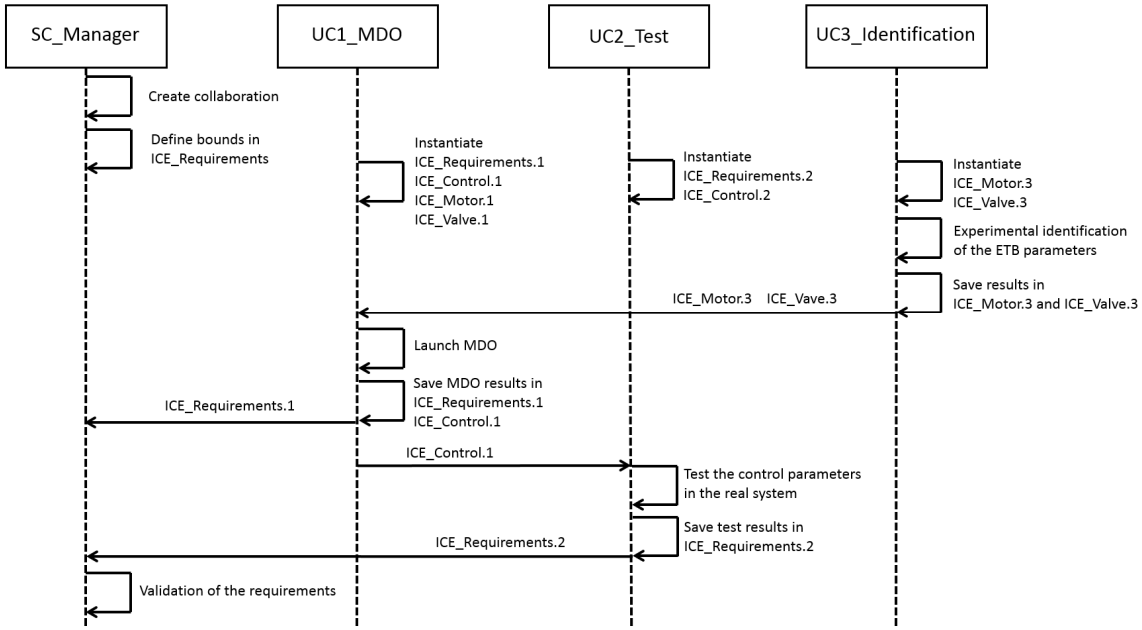



Figure 3-12. Sequential diagram of the collaboration

### 3.6.3 Results

The scenario was tested in Karren platform. Figure 3-13 summarizes the results of the collaboration. The consistent parameters represent the parameters sent between the users. Moreover, non-consistent parameters represent parameters that need validation. The results of the optimization respect the requirements and they are near to experiment test values.

A graphical comparison is also presented in Figure 3-14. A good agreement was obtained with experiment but there are low errors due to the assumptions made in the model. We notice that the return time is less than the rise time because in the return phase the main spring and the motor act in the same direction. The collaboration leads to a validation of the controller values.

In practice, this example which is relatively simple was difficult to setup because the decomposition of ICEs and the centralized collaboration process. The next section explains these limits.

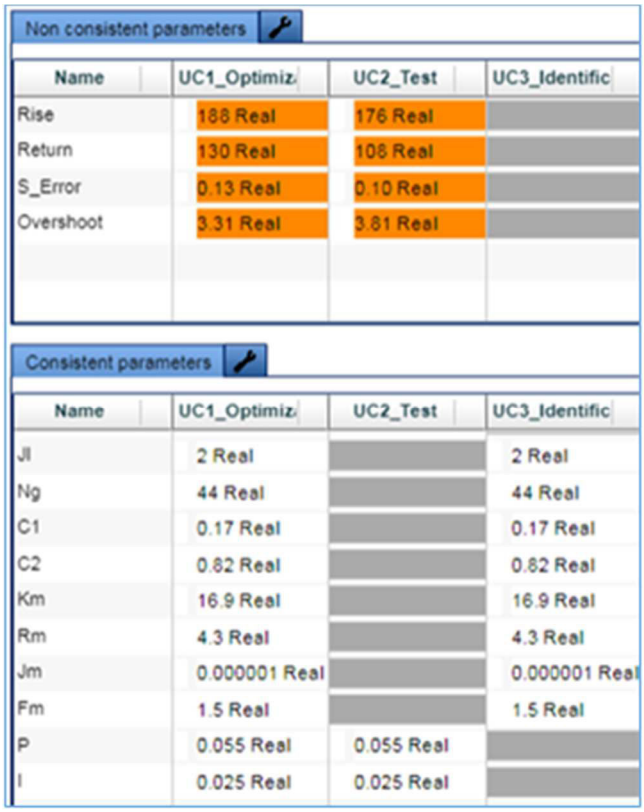



Figure 3-13. Results of the collaboration

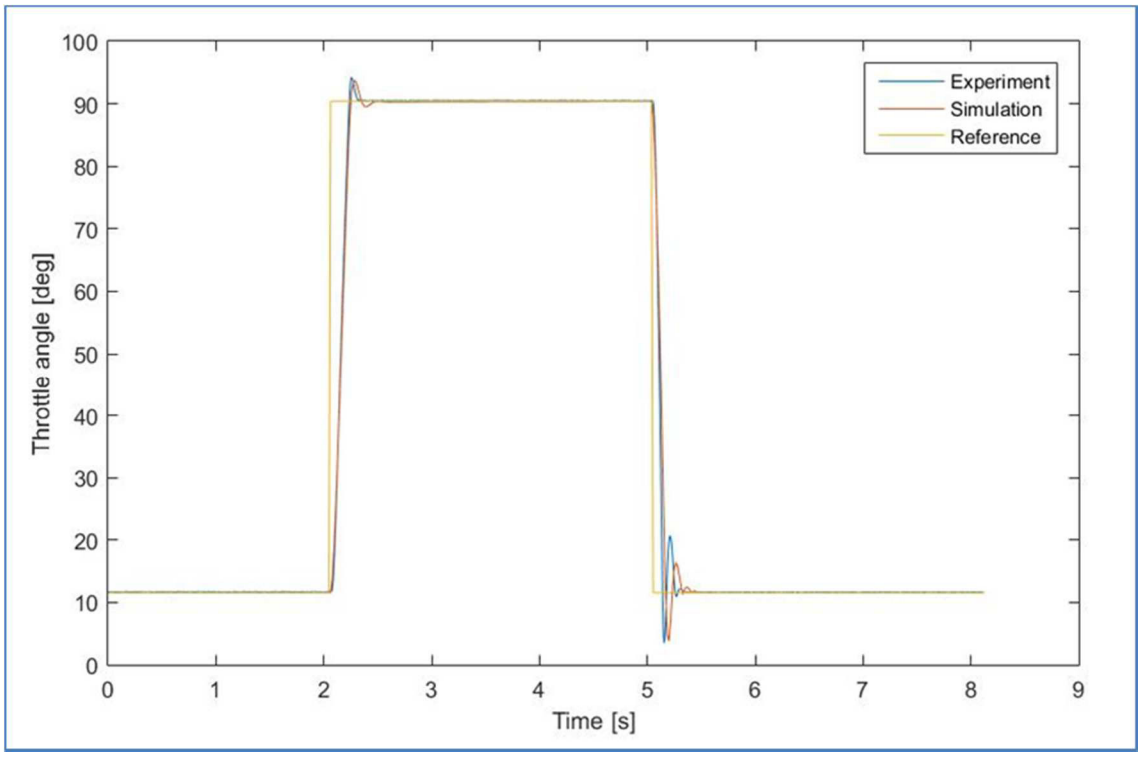



Figure 3-14. Comparison between simulation and experimental results

## 3.7 Towards a new KM model

As we can notice, ICE is at the heart of the KCM. These entities contain all the information and constitute a cross-functional baseline. The defined methodology partially solves our initial problematic. Indeed, KCM can ensure the consistency between the different models of the design process and tasks parallelization is then possible. Nevertheless, this consistency is not enough to ensure the continuity between SE, DE, and MDO. Even with the methodology that we propose, KCM still have limitations in the model and cannot be applied in complex cases. Here are listed the limits of this model:

- Difficult decomposition of the knowledge. This step is important in KCM because the ICE structure will be used in all the cycle. However, the decomposition of parameters in definition phase is different from the decomposition in the analysis phase and users can be confused.

- The ICE instance is indecomposable and may be unsuitable for some users. For example, an ICE instance may contain one important parameter and other useless parameters that can mislead the user. In fact, a user cannot select one parameter, but have to take the ICE instance with all its parameters.

- KCM detects conflicts between users but does not manage decision making (Drémont, Troussier et al. 2013).

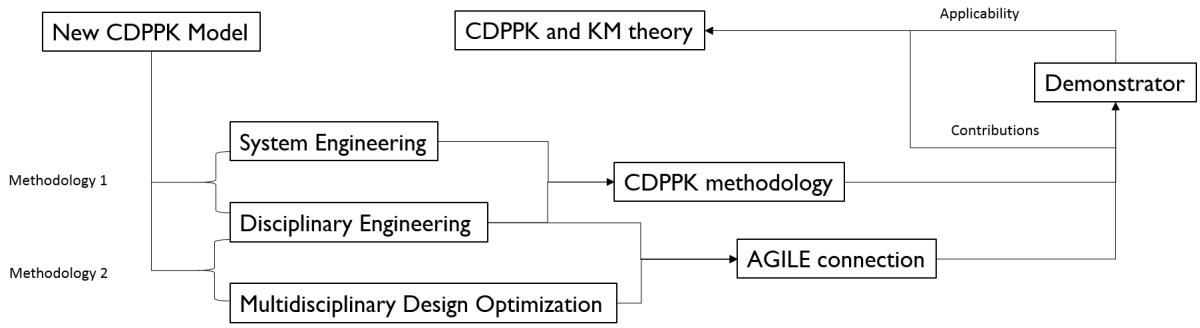
- The knowledge reuse is difficult because there is no traceability of the design process. It is important to save the final values of the project parameters but there is no information about how these parameters are calculated (Belkadi, Dremont et al. 2012).

- The collaboration is managed by the consistency between parameters. This centralized way is efficient for validation but not suitable for normal exchange.

As a result, stakeholders have problems accessing the information they need and also to understand previous projects. The limits listed indicate a clear demand for a new approach that suits the different views of the design cycle. Even with the contributions made in the methodology to use KCM in mechatronic design, modifications are needed in the structure of KCM itself. KCM can be used in simple projects but to be adapted for complex mechatronic projects. Based on these limits we propose in the next chapter a new original model and its associated methodology.

# Chapter 4 Collaborative Design Process and Product Knowledge Methodology

*This chapter presents our new model Collaborative Design Process and Product Knowledge (CDPPK). The model is first detailed then we present the methodology to connect SE and DE. The process knowledge generated during the first connection allows us to connect DE and MDO using KADMOS methodology. A Python demonstrator is presented to illustrate our purpose. Finally, we present the KM dimensions of CDPPK.*



## 4.1 CDPPK principle

The success of any collaboration framework is strongly linked to the need to share knowledge between actors to ensure a common representation of the problem to be solved (Assouroko, Ducellier et al. 2014). However, the shared knowledge is composed of fragments that are created by various actors according to their expertise domain. An important aspect to be considered is then the adaptation to these different viewpoints between SE, DE, and MDO.

One may wonder how we can adapt the already available KCM approaches in a way that covers the entire design cycle. Our goal is to manage and communicate knowledge between the decomposition phase and the integration phase. Our framework aims at overcoming collaboration and reuse issues. For collaboration, a new way of viewing the product is necessary to find solutions that lie across disciplinary boundaries. For the reuse, the current KCM focuses on declarative knowledge (know-what). This knowledge is important to understand the product but should be completed by process intents (know-how) to facilitate the project reuse.

Our new approach, illustrated in Figure 4-1, is inspired by KCM structure and our contribution lies firstly on reorganizing parameters in order to facilitate collaboration. In fact, all parameters involved must be reflected in a well formalized conceptual infrastructure to ensure an effective connection between SE and DE environments. On the one hand, we propose the use of ICEs as a point of reference for system engineers. On the other, disciplinary designers are only to instantiate the parameters that they need from specific ICEs. Accordingly, these disciplinary engineers become detached from the conceptualization of the ICEs.

Additionally, we added for each parameter an in/out flow to connect the disciplinary engineers and create a collaborative network that will be used later for DE – MDO connection. We will explain the main concepts of our model then the methodology to connect SE – DE and DE – MDO.

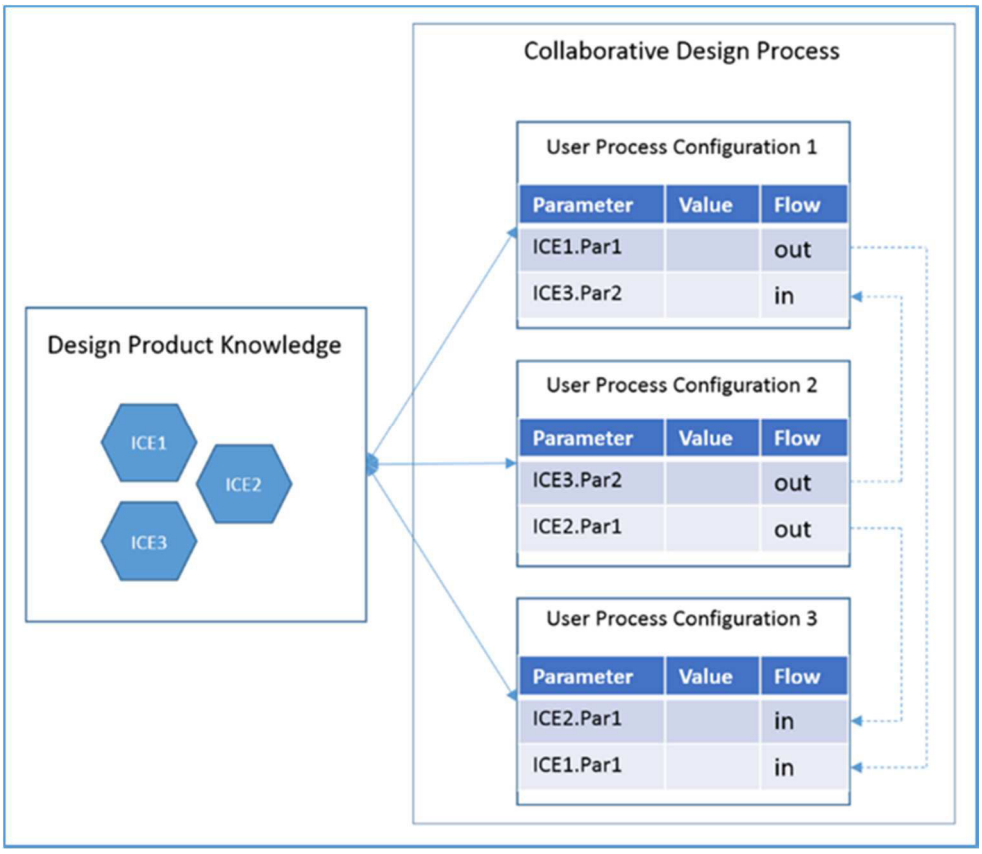



Figure 4-1. Collaborative Design Process and Product Knowledge principle

## 4.2 Main concepts

### 4.2.1 Information Core Entity (ICE)

We keep the concept of ICE in this methodology to capitalize knowledge. The difference with the previous methodology is that ICE is considered here as an organizational structure only for the decomposition phase. System engineers use ICE to decompose the system into functional subsystems and save the associated parameters and constraints. Therefore, ICE is used here to transform product data into product information. The parameters can be: reals, interval, vectors and matrix to give more flexibility to the stakeholders. For the constraints, we consider only interval constraints in this work.

### 4.2.2 Design Product Knowledge (DPK)

DPK contains all the ICEs used in the project. Stakeholders can add here the parameters and the constraints. DPK can be considered in this stage as a product information base. When the collaboration starts, stakeholders refer to this base which represents the system level. DPK also

plays an important role in conflict management, as we will explain later. After the collaboration, ICEs will contain the final product results. Thus, DPK generates product knowledge from information stored in ICEs.

### 4.2.3 User Process Configuration (UPC)

This new class differs from the UC defined in KCM. Instead of affecting a block of parameters in the form of ICE instances, we propose to break ICEs and give the flexibility to disciplinary engineers in the choice of parameters that they need. Our contribution is to define input/output flows for the instantiated parameters. This considerably facilitates the collaboration and data exchange between stakeholders and helps to construct a collaboration network.

### 4.2.4 Collaborative Design Process (CDP)

CDP contains all the UPCs present in the collaboration. It manages the disciplinary conflicts between UPCs and ensures the connection with DPK. All the disciplinary engineers are then connected to CDP which achieves the traceability of the collaboration. The collaborative design process is formalized in a directional graph which facilitates knowledge reuse. All the decisions made during the collaboration are also stored in CDP.

### 4.2.5 Project Domain (PD)

The concept of Project Domain makes it possible to define a macroscopic context for the definition of the basis of ICEs and the use of configurations. Indeed, we can qualify a PD as a global usage environment in which a system can be defined and decomposed into major subsystems. In this way, we capitalize the respective knowledge to each subsystem and we reduce the complexity during collaboration and reuse.

### 4.2.6 CDPPK metamodel

This new CDPPK model is adapted to the mechatronic design cycle. In the decomposition phase, each functional subsystem is represented by an ICE. Then, in the integration phase, each DE tool is represented by a UPC. We notice that the knowledge cycle (data-information-knowledge) for the design process and product is respected. The UML metal-model of CDPPK framework is shown in Figure 4-2. The different classes and connections are represented between the entities defined previously. This model is based on Configuration Management (CM) defined in ISO 10007:2015. CM carries the access to the adequate information relative to prod-

uct design, realization, verification and use. This information is organized in a “global configuration” which is updated and versioned to trace the evolution of the product. In CDPPK, this organizational concept is the Knowledge Configuration (KC). The standard defines also “configuration items” that are components of the global configuration to ensure the final use of information and allow users to converge to a single product. In our model, these items are the UPCs.

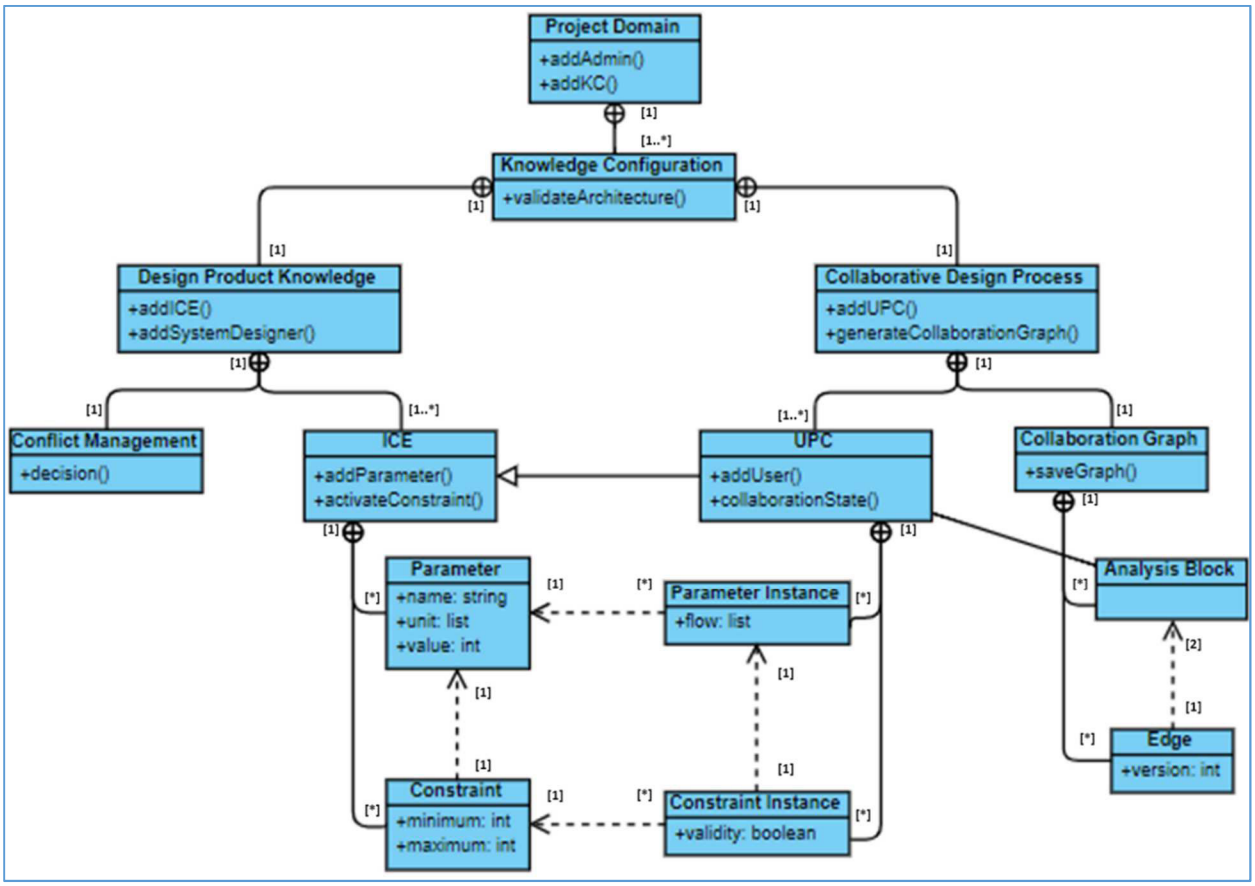



Figure 4-2. UML meta-model of CDPPK

## 4.3 SE-DE methodology

We define a methodology to connect SE and DE using CDPPK. The main difference with KCM is that we can have disciplinary and system viewpoints in the same framework. The methodology is detailed in the next paragraph followed by the conflicts management approach.

### 4.3.1 Methodology steps

Actors in the design cycle are submerged in a heterogeneous tools environment and information flow. They collaborate with traditional and informal ways (email, meeting...). These ways are

error-prone and impossible to trace for reuse perspectives. Whereas, if we have effective collaboration we can acquire not only short-term benefits, during the product development, but also long-term ones, during the reuse process. KM was proposed to improve collaboration performance in mechatronic design. Nevertheless, it is not limited to its technological approach for it, also, includes a sociotechnical approach which considers knowledge as a resource participating in companies' performance (Arduin, Grundstein et al. 2013). A very common problem of a failed collaborative framework is that it does not meet the users' real tasks. Therefore, the CDPPK's methodology is created with regards to the mechatronic design cycle (Figure 4-3). Each step is adapted to each stakeholder involved in the design cycle and their needs.

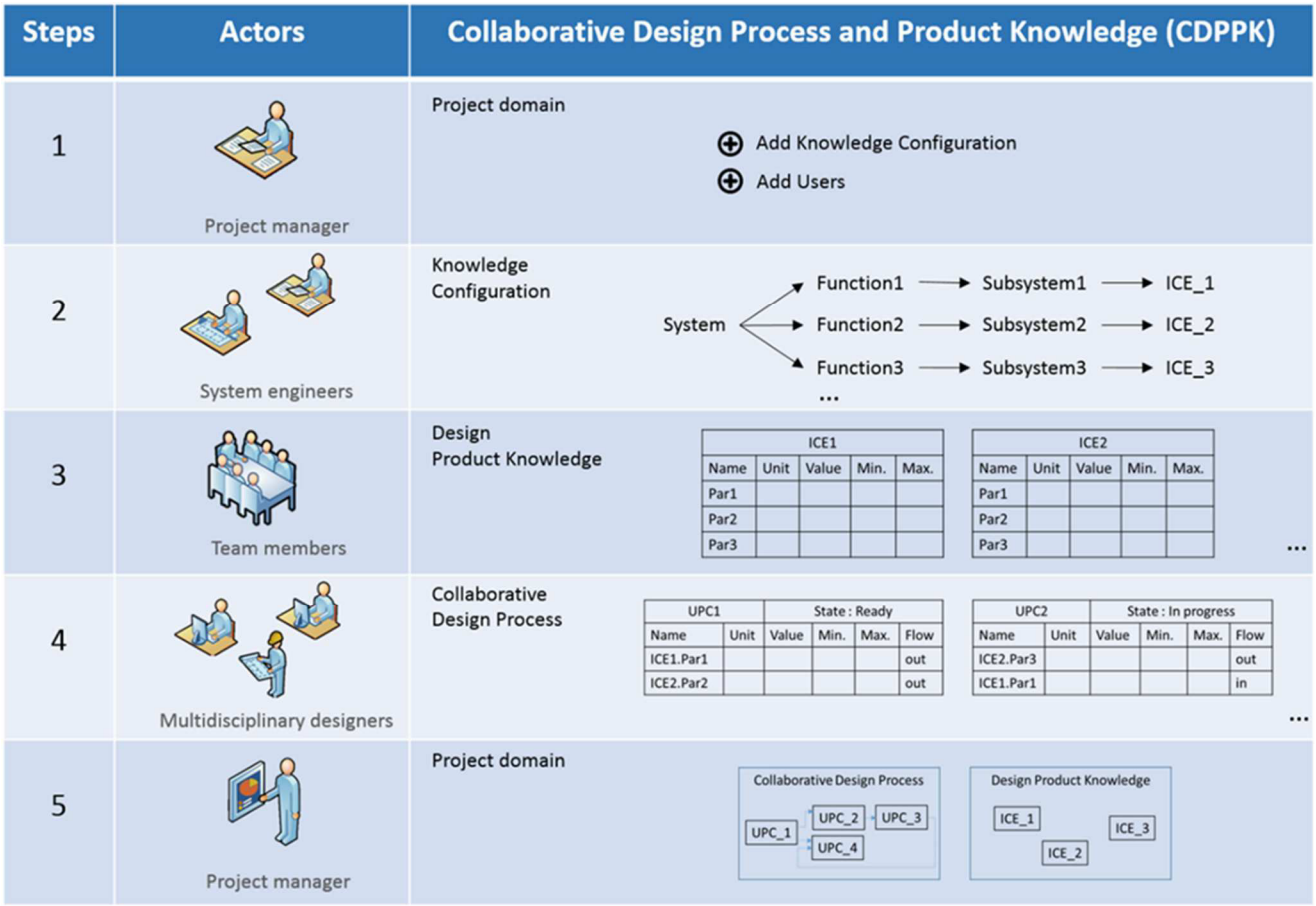



Figure 4-3. SE-DE methodology for CDPPK

Step1: Project Requirements

The project manager analyses the requirements of the customer. Accordingly, they create a Knowledge Configuration and identify the actors needed for each step. Their main objective is to lead the team towards accomplishing the project.

Step2: Conceptual Design

Based on customer requirements, system engineers define the functions of the system. They can explore several architectures according to the defined functions, in this work, we will focus on the development of one architecture. Our hypothesis is to decompose the system into functional subsystems. Each one of the latter is represented by an ICE. The available parameters and constraints (from requirements and previous works) are added in this phase to the ICEs.

Step3: Detailed Design

Here, the previously created ICEs are enriched with all the crucial parameters necessary for the integration phase. To do so, we recommend a meeting between the different stakeholders. A knowledge expert can also capitalize the crucial knowledge necessary for the collaboration. Semi-automatic extraction of parameters methods exists also. This step is a direct intermediate between the decomposition and the integration phases.

Step4: Verification & Integration

In this step, disciplinary engineers create their own User Configuration Process (UCP). Each engineer instantiates the parameters that they need in their own UPC. Then, they instantiate the parameters to link with their DE model. Then, the in/out flow is selected for each parameter. When all the engineers have filled their UPC, the collaboration starts and the exchange of the parameters follows the network generated by different parameters flows. Each designer gets to know which UPCs are currently related to their work. Each UPC has a collaboration state to manage the exchange (Start/Ready/In Progress/Publish). We will further explain this point in the following section.

Step5: System Validation

Thanks to the product knowledge in DPK and the process knowledge in CDP, the project manager has a holistic view of the design cycle. They need to check the collaboration state and the recorded conflicts between system engineers and disciplinary designers in order to assess the situation and make decisions.

This methodology enables each stakeholder to access and use the right knowledge. At this point, Knowledge becomes of great value for the company (Ilvonen, Jussila et al. 2015). In managerial terms, our methodology moves from the traditional predefined collaboration process to a more dynamic and flexible one. It provides a common basis of exchange that does not disrupt the tasks of the stakeholders. The main inconvenience of such methodology is the need to convene the different stakeholders to formalize knowledge for it might be time-consuming. Yet, research covered in this section proves the long-term benefits of KM (Assouroko, Ducellier et al. 2014). That is to say, that the time spent in formalization will be compensated in reuse.

### 4.3.2 Collaboration and conflicts management

Web technology communication can ensure the exchange of information between different actors but when it comes to complex multi-stakeholder problem resolution, it does not succeed to provide active supports to resolve conflicts. In collaborative design, a conflict is an incompatibility between two (or more) design decisions that a design party has a negative critique of another design party's actions. According to our methodology, conflicts are generated when parameters' values do not respect the constraints. The constraints associated with the parameters can be system constraints (defined at the beginning in the ICE) or interdisciplinary constraints (defined in UPC during the second phase). Therefore, system constraints help us to manage requirement changes and interdisciplinary constraints help us to manage conflicts between disciplinary engineers (Figure 4-4). We defined collaboration states for UPC and CDP in order to organize the conflict resolution process. Each UPC has four collaboration states in accordance with the four facets of knowledge management cited by Grundstein: identify, formalize, value and update (Grundstein 2000):

- Start: In this initial state, actor instantiates the needed parameters
- Ready: When all the parameters and the flows are assigned, the actor switches to "Ready" state to indicate their readiness to collaborate. They can receive in this state input parameters from collaborators who published their work.
- Publish: When the actor finishes their tasks, they switch to "Publish" state. All the output parameters will be sent to the other collaborators. Each publication upgrades the version of the UPC.

- Conflict: When a parameter value does not respect the constraint (system constraint or interdisciplinary constraint), the user switch to conflict state and the concerned collaborators are informed automatically.
- In parallel, the CDP has its own cycle:
- In progress: When one or many UPCs are in start state, it means that they didn't choose all the parameters they need.
- Process: When all UPCs are in ready status, it means that actors defined all the parameters and flows. The collaboration graph can be generated in this state.
- Collaboration: When all actors are publishing their work and/or solving their conflicts. In the case of an unresolved conflict, the project manager intervenes to provide a balanced solution based on compromises in global vision.
- Validation: When all UPCs are in Publish status. In this state, the final values of the collaboration are saved in DPK inside ICEs and the final collaboration graph is saved in CDP.

Developing products without a good collaborative environment can result in unnecessary iterations, higher development costs, and quality problems. This methodology assists designers finding optimal solutions and making decisions by structuring knowledge for different actors. The process knowledge plays an important role in decision making and collaboration as it will be explained in the next section.

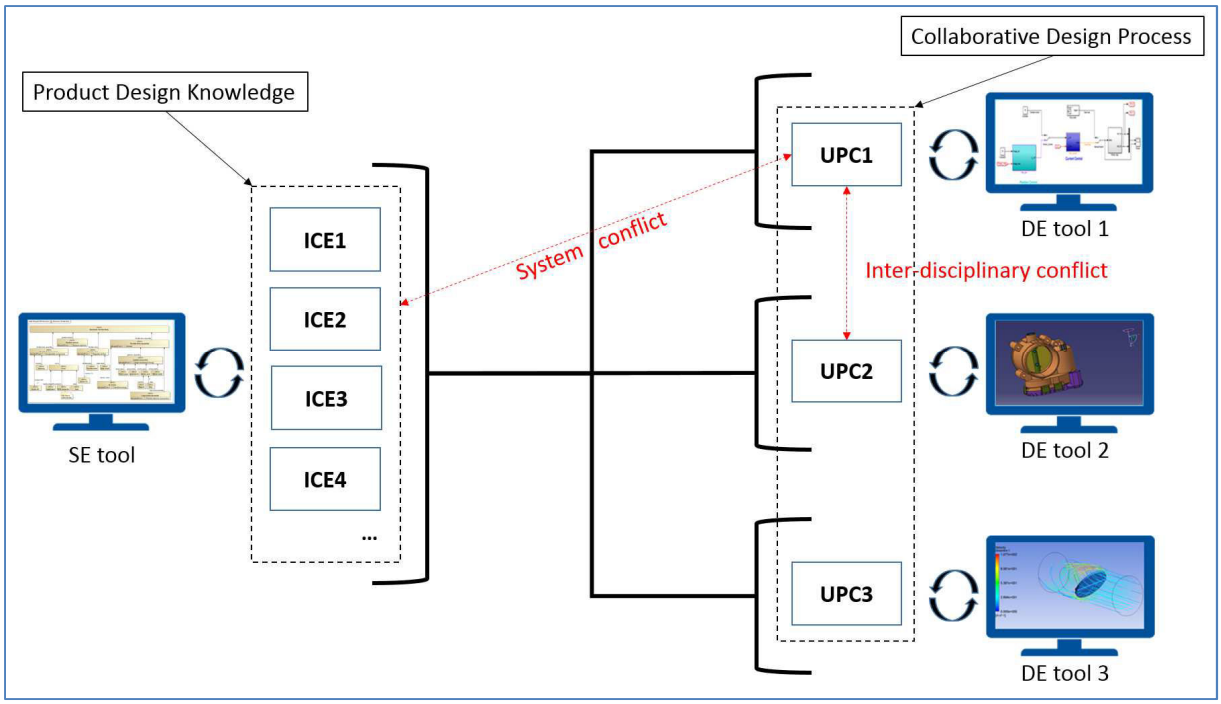



Figure 4-4. System and interdisciplinary conflicts management in CDPPK

Ensuring the consistency between different design models is crucial in a collaborative development environment. The purpose of conflict management is to detect inconsistencies and trace this information back to users to allow them to iterate, make decisions and finally converge towards a common solution. Once the conflicts are detected, it is a matter of putting the information back to the different users. In the case of interdisciplinary conflicts, the conflicts are listed in CDP and managed directly by disciplinary engineers. In the case of system conflicts, it is the PDK that is responsible for displaying this information for system engineers. In this way, the traceability and coherence of manipulated knowledge is guaranteed in the entire cycle. The next section explains how the generated knowledge can be used for MDO.

## 4.4 DE-MDO connection

The National Science Foundation Workshop argued that the future of MDO is not MDO; rather, the future lies in "Multidisciplinary collaborative human decision making" (Simpson and Martins 2011). Our methodology consists in using the process knowledge generated during the collaborative design process in order to create an MDO problem. In this way, we use human collective knowledge and decisions to generate the MDO problem as we explain in this section.

### 4.4.1 Graph generation

The collaboration management in CDPPK is based on graph theory (Bondy and Murty 1976). To generate collaborative graph process and manage the exchange of knowledge we propose to generate a unidirectional graph constituted of edges and analysis blocks. Each analysis block represents a UPC and the edges represent the exchange of parameters between the UPCs. Pate presents a graph approach to problem formulation for multidisciplinary design analysis and optimization (Pate, Gray et al. 2014). He presented two kinds of graphs:

- Maximal Connectivity Graph (MCG): Represents all the possible interconnections between DE tools.
- Fundamental Connectivity Graph (FCG): Simplification of the MCG to represent only the tools and the connections necessary to solve an engineering problem.

The approach is based first on Fundamental Problem Formulation (FPF). It is a general way to present an engineering or optimization problem by specifying the objectives to minimize, the constraints to respect and the design variables (Cramer, Dennis et al. 1994). Based on this formulation, there are two limits to this approach. The first limit is the difficulty to create the MCG when the models are dynamic. In fact, when only one analysis block changes this will affect the whole MCG graph. Second, one cannot have all the information needed to transform MCG into FCG. In our approach, we create directly an FCG. We propose to implicate actors in the creation of the global graph by recovering their work in UPCs that represent the nodes that constitute the FCG. Given the nature of UPCs, the FPG can be generated dynamically at any moment.

As illustrated in Figure 4-5, analyses are considered as blocks with inputs and outputs:

- Parameter connecting two analyses blocks is a coupling variable (out 1 of analysis 1 for example).
- Parameter without flow is a shared variable.
- Output parameter without a connection and without a constraint is an objective variable.
- Output parameter without a connection and with a constraint is a constraint variable.

- Input parameter with two entries is a decision node (when two multi-fidelity analyses output the same parameter as analysis block 1 and analysis block 2 for example).

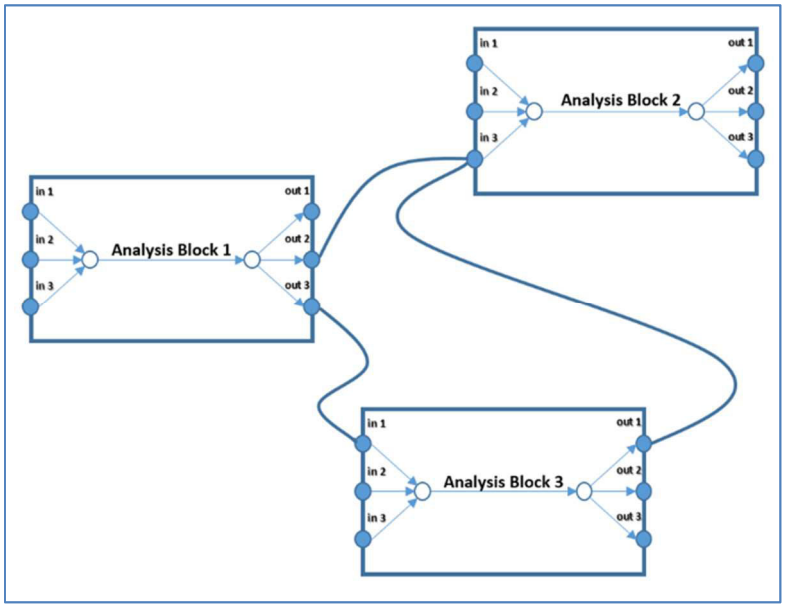



Figure 4-5. Exchange graph between DE models

This can be used to facilitate collaboration and reuse. For the collaboration, this graph offers an overview of dynamic parameters workflow. Therefore, it helps to detect analyze conflicts and gives a global view of the collaborative design process. For reuse, this graph represents a collective knowledge that stakeholders can learn from it to understand previous decisions and to improve the design process. In this work, we will also use this graph to facilitate automation by means of MDO.

### 4.4.2 MDO problem formulation

To connect DE and MDO, we propose to generate an MDO architecture from the collaborative design graph. MDO is an important technique in mechatronic design. However, the cost-effectiveness and scalability of an MDO process are highly dependent on designers' ability to rapidly formulate the optimization problem for a specific situation. The problem of determining a feasible data flow between DE tools to produce an MDO architecture is combinatorically challenging (Simpson and Martins 2011). To get an MDO process accepted by its planned users, the problem must be customized to their individual requirements. Therefore, we propose to start from the process knowledge generated by the collaboration between designers. The feasibility of the process was validated previously and this graph presents therefore a precious knowledge process for MDO. Our goal is to use the generated graph in KADMOS open source platform.

This platform was developed in the European AGILE project and allows automatic generation of MDO problems (Ciampa and Nagel 2017).

### 4.4.3 CDPPK-KADMOS connection

The KADMOS platform proposes an interesting methodology to automate the MDO problem generation (Lefebvre, Bartoli et al. 2017). This methodology contains 6 steps as illustrated in Figure 4-6:

1. Import tools and connections: A maximal connectivity graph is created based on XML baseline containing knowledge about the tools and all the parameters.
2. Formulate MDO problem: Transformation of MCG graph to FPG graph depending on the MDO problem.
3. Import MDO architecture on problem: Use the best MDO architecture to solve the problem.
4. Export CMDOWS file: The generated graph is exported in standard XML format called CMDOWS to be executed in a PIDO software (the software compatible with this format are currently: RCG, Optimus and OpenMDAO).
5. Create executable workflow: An MDO workflow is automatically generated in the PIDO software.
6. Run optimization and post-process results: This final phase is an analysis of the MDO results.

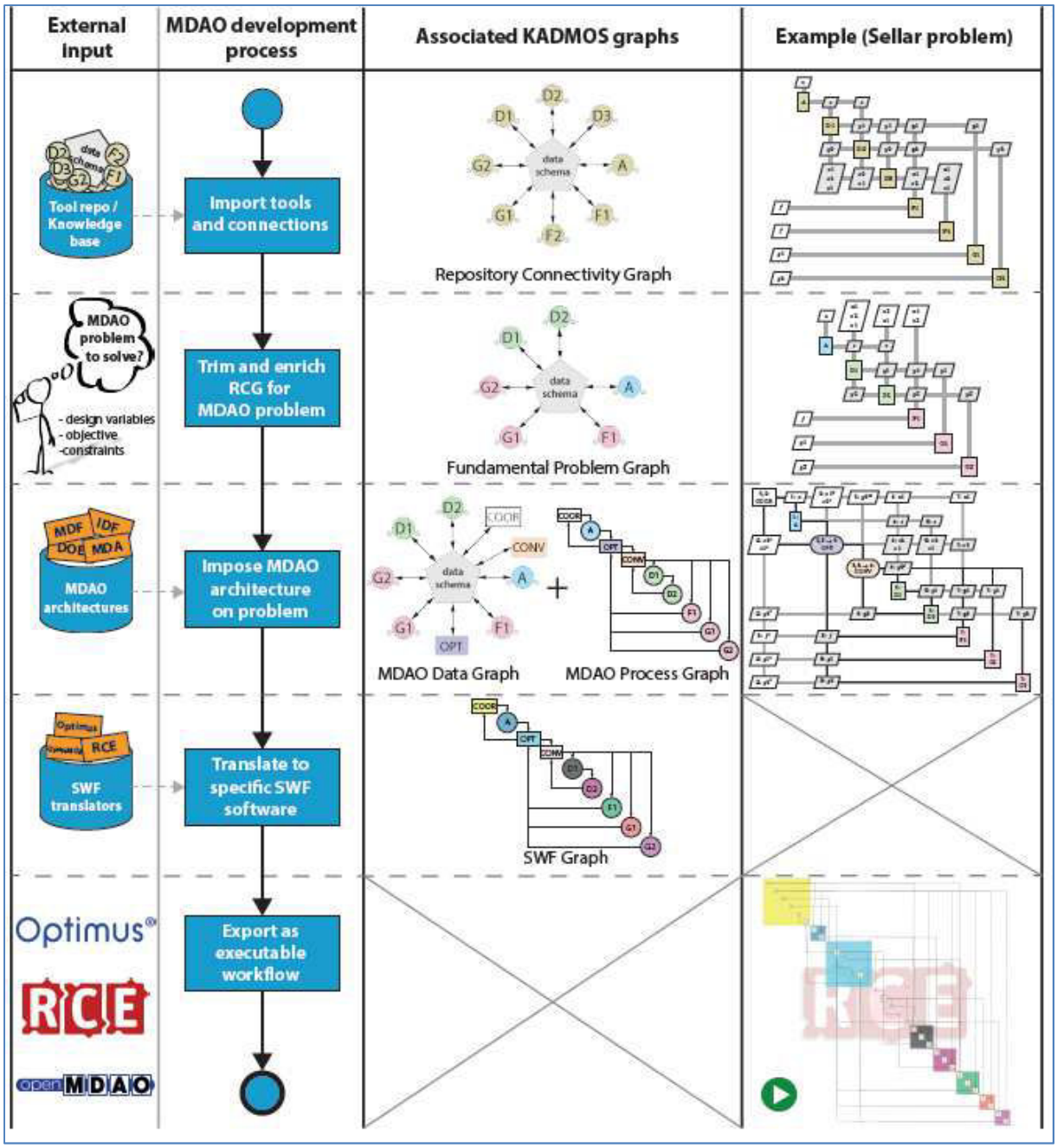



Figure 4-6. KADMOS platform overview (van Gent, Ciampa et al. 2017)

Our methodology is to use the knowledge generated during the collaboration and connect our FPG graph directly to the second step of KADMOS process. In this way, we reduce the setup time and we import a graph that was tested successfully in manual collaboration. Disciplinary designers participate in the creation of the MDO problem. This will reduce considerably the errors while launching the optimization in the PIDO tool.

This methodology was implemented in a Python demonstrator as it will be explained in the next section.

## 4.5 Python demonstrator implementation

A Python demonstrator was implemented to show the ability of CDPPK to assist designers in the whole collaboration process. Due to the time allocated, this demonstrator is not related to a web server that permits distributed collaboration but we choose to work on a local application. The goal of this demonstrator is to illustrated CPPPK steps and to validate graph theory aspects. This demonstrator contains a Product Design Knowledge window, a Collaborative Design Process window and a graph generator.

### 4.5.1 Product Design Knowledge part

In this window, we can add the necessary ICEs for the collaboration and fill the parameters in each ICE. As reported before, only interval constraints are considered for this demonstrator. When the collaboration starts, the system conflicts are listed also in this window (Figure 4-7).

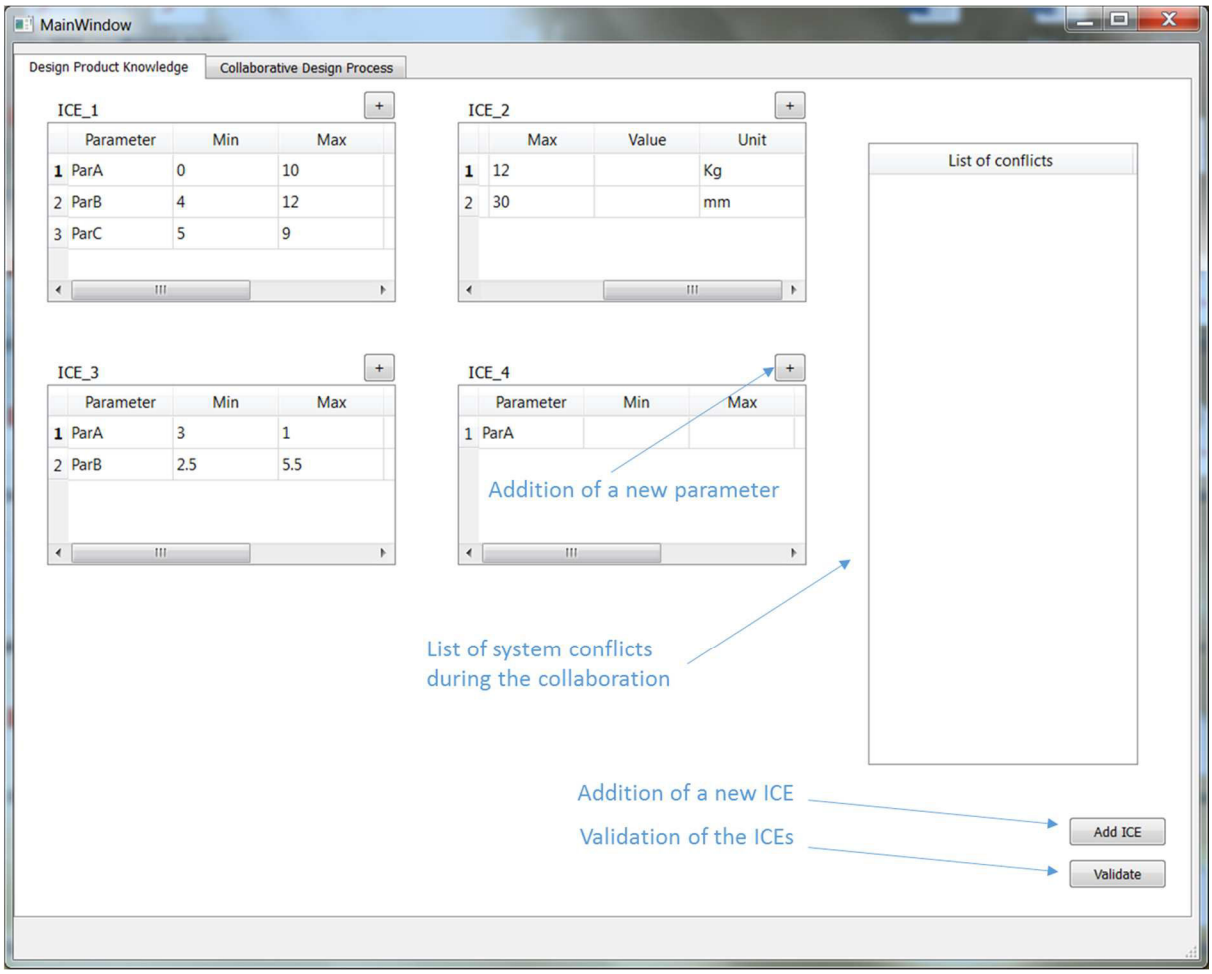



Figure 4-7. Design Product Knowledge in Python demonstrator

### 4.5.2 Collaborative Design Process part

In this window, we can add UPCs. Normally each UPC represents a user, but for this local application we manage all the UPCs in one window. For each UPC, we can add a parameter from the list of all the parameters defined before. Then the collaboration starts by defining the parameter flows and the collaboration state (Figure 4-8).

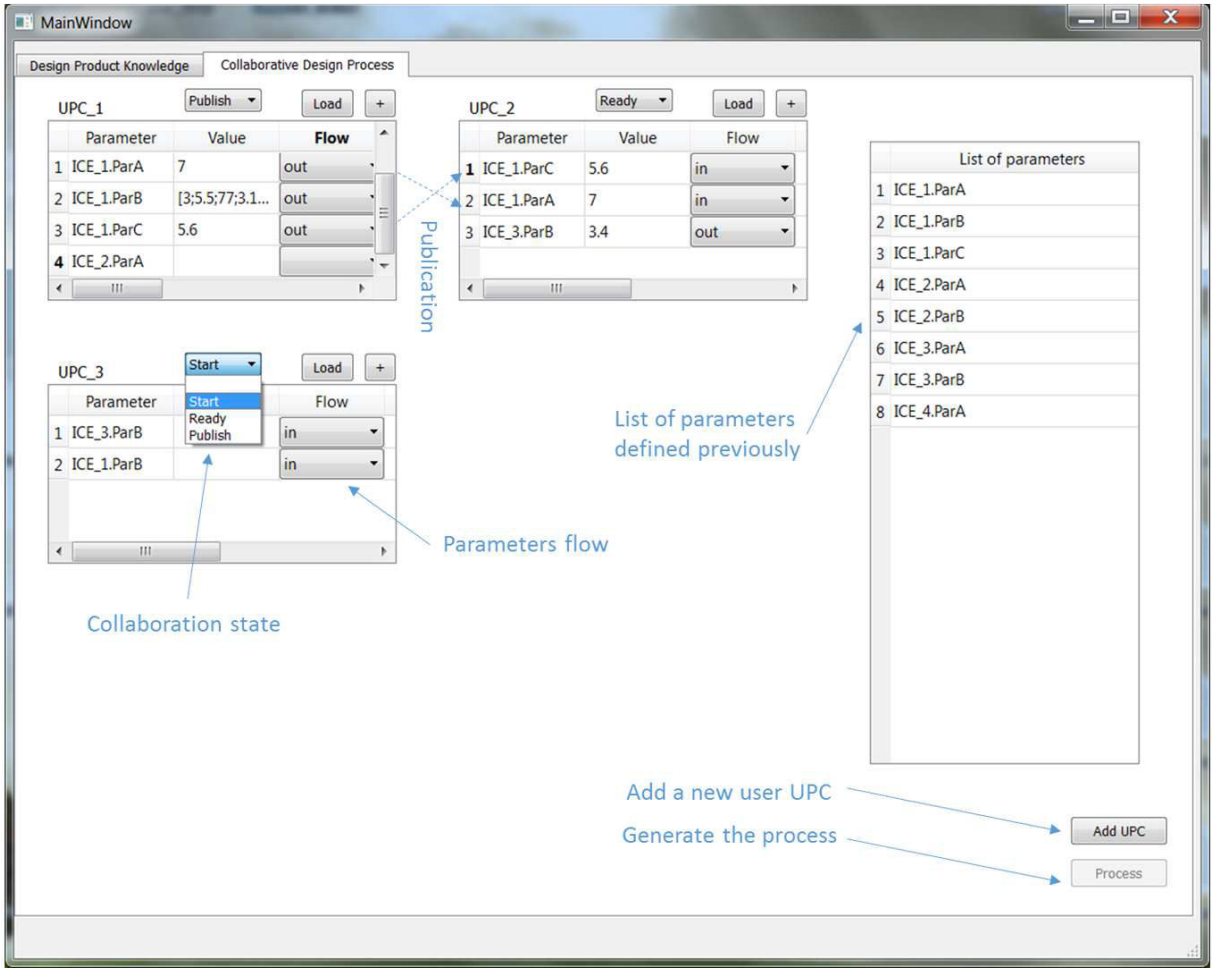



Figure 4-8. Collaborative Design Process in the Python demonstrator

### 4.5.3 Graph generation

When all the UPCs are in “Ready” state, the process button is activated and we can generate automatically the design collaboration graph using the principle described in the previous section. The graph and the connections are generated automatically using Python (Figure 4-9); It is possible to generate an XML file based on this process for KADMOS platform in order to generate automatically an MDO problem.

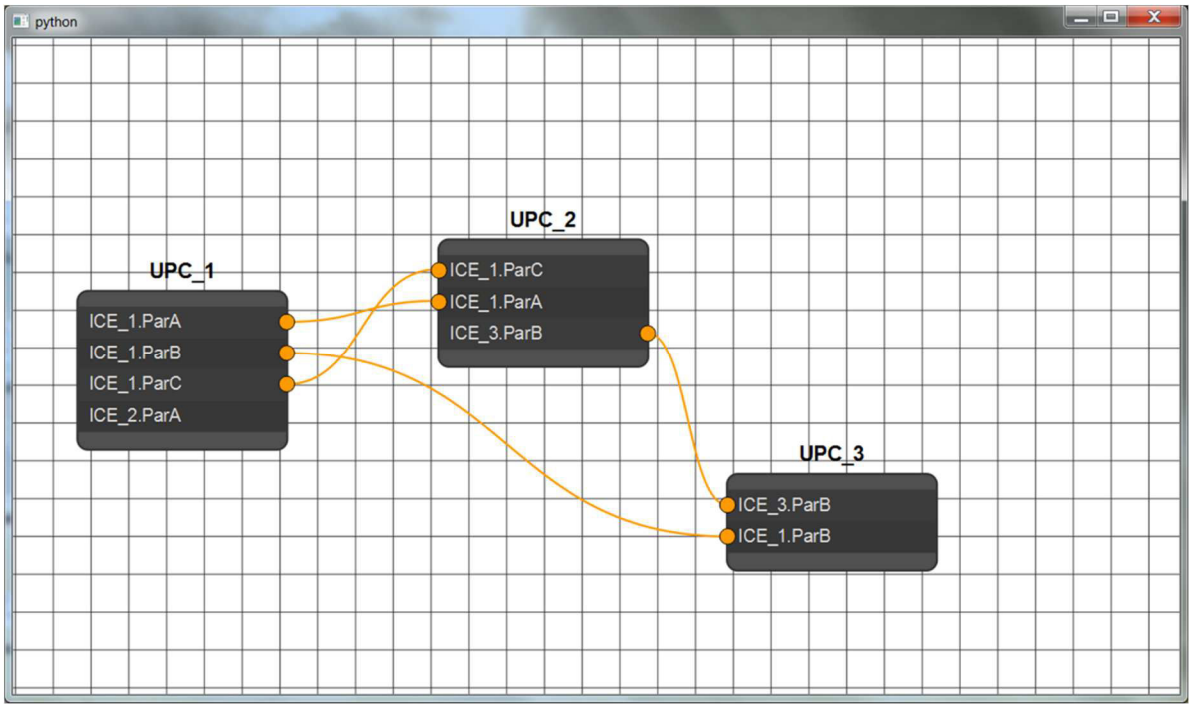



Figure 4-9. Graph generation in Python

## 4.6 Conclusion

For mechatronic design, informal ways of collaboration are not sufficient when the system is complex and many iterations can appear between stakeholders to solve conflicts. Using CDDPK, the actors will benefit from a holistic view of the system and conflict resolution ways to reduce iterations. Then, the formal product knowledge and the process knowledge generated during the collaboration will considerably improve the reuse and the MDO problem automatic generation.

### 4.6.1 SE-DE and KM

Our solution is based on a bidirectional connection between the product knowledge and the process knowledge. The list of parameters and constraints are instantiated by stakeholders in their UPCs. A knowledge process is then generated with the exchanges between the stakeholders. When the conflicts are solved, the final parameters are saved in the DPK.

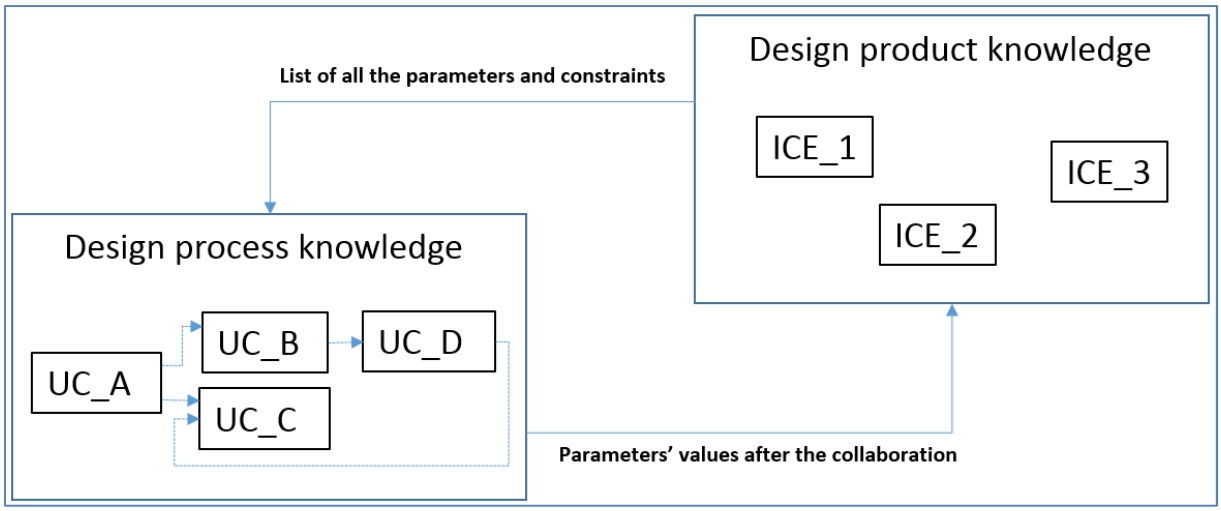



Figure 4-10. Connection between design product and process knowledge

The product knowledge cycle in CDPPK is the following:

– Data: parameters and constraints organized in ICEs. Without a context, these ICEs are considered as data at the beginning of the project.

– Information: parameters and constraints instantiated in UPCs. In fact, UPCs are used in specific activities and they are synchronized with specific DE models.

– Knowledge: when the collaboration progresses, UPCs and ICEs evolve from a version to another depending on the conflicts' resolution. The access to these versions containing the different sets of values and the history of changes is the product knowledge.

### 4.6.2 DE-MDO and KM

The connection between DE and MDO is carried out using the collaborative design graph. This graph represents the workflow between the DE models and can be used as a basis to generate automatically an MDO architecture. The results of the optimization can be linked directly to DPK using the connector defined in Chapter 3.

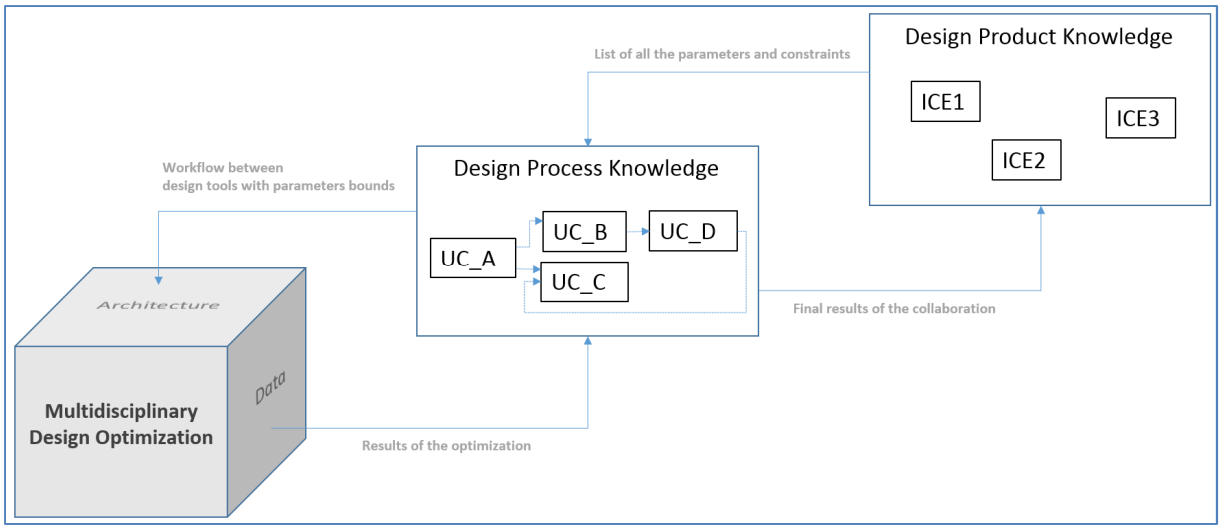


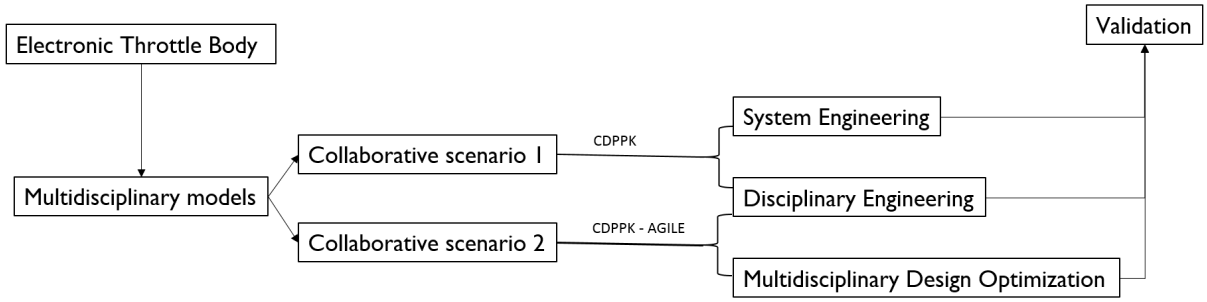

Figure 4-11. Knowledge management in CDPPK

One of the main goals of KM is the automation. By using our approach we can learn from the collaborative design process and when we have enough knowledge about the process we can automate it using MDO.

To validate this approach we propose collaborative scenarios in the next chapter based on the ETB design cycle.

# Chapter 5 Validation of the methodology

*This chapter validates our approach using the Electronic Throttle Body example. Multidisciplinary models and the test bench of ETB are presented. First, a collaborative scenario is implemented to validate the SE-DE connection. Second, an optimization problem is solved to validate the DE-MDO connection.*



## 5.1 Multi-disciplinary development of the ETB

In order to demonstrate our approach, we created different models used in the ETB as well as a test bench to reproduce the industrial design cycle. Based on industrial requirement, we applied our demonstrator in two use cases: SE-DE use case and DE-MDO use case. In this section, we introduce the disciplinary models of the ETB as well as the test bench.

### 5.1.1 Modelica model

The Modelica model of the ETB encompasses five physics (electric, mechanical, thermal, fluid, control) and was created for optimization purposes (Figure 5-1. Fast simulations can be generated using this model but it needs 3D enrichment to improve the accuracy. More details about the analytic equations of the ETB are detailed in Appendix 1.

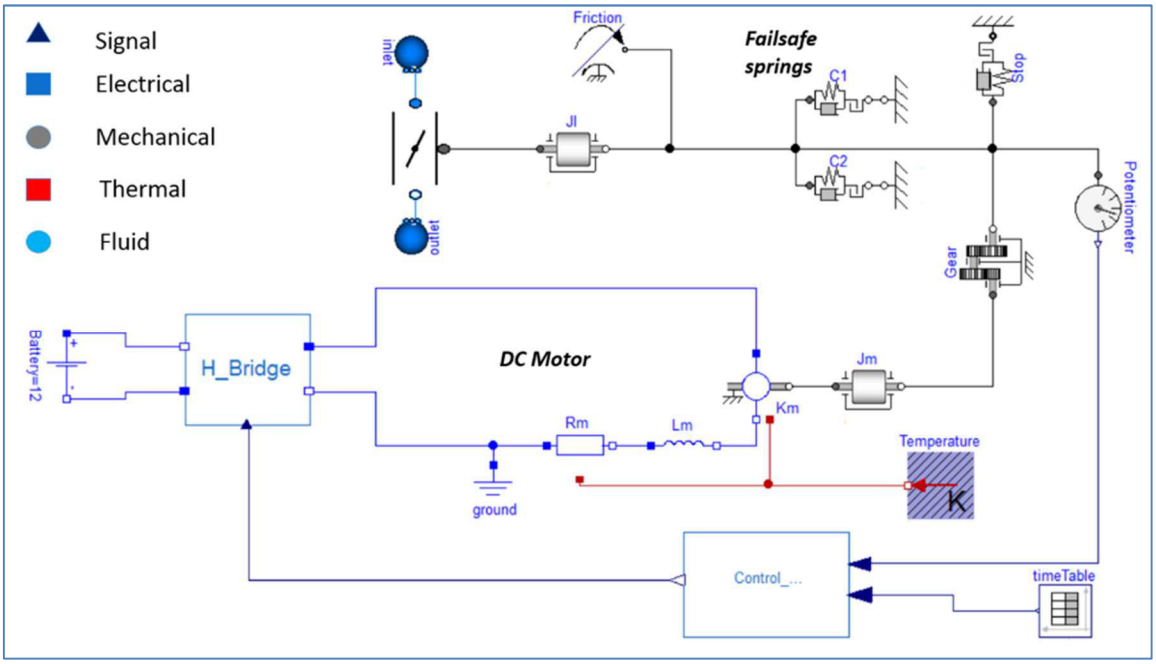



Figure 5-1. Multi-physical model of the ETB using Modelica

### 5.1.2 Control model

The control model in Simulink software contains 4 control systems (PI, PID, Sliding 1, and Sliding 2) with the possibility to switch from one control to another (Figure 5-2). This model is connected to the dSPACE environment for experiment purposes. Sliding control was tested to provide more robustness for parameters (Amini, Razmara et al. 2017). More details about the analytic formulation of the sliding mode controller are detailed in Appendix 2.

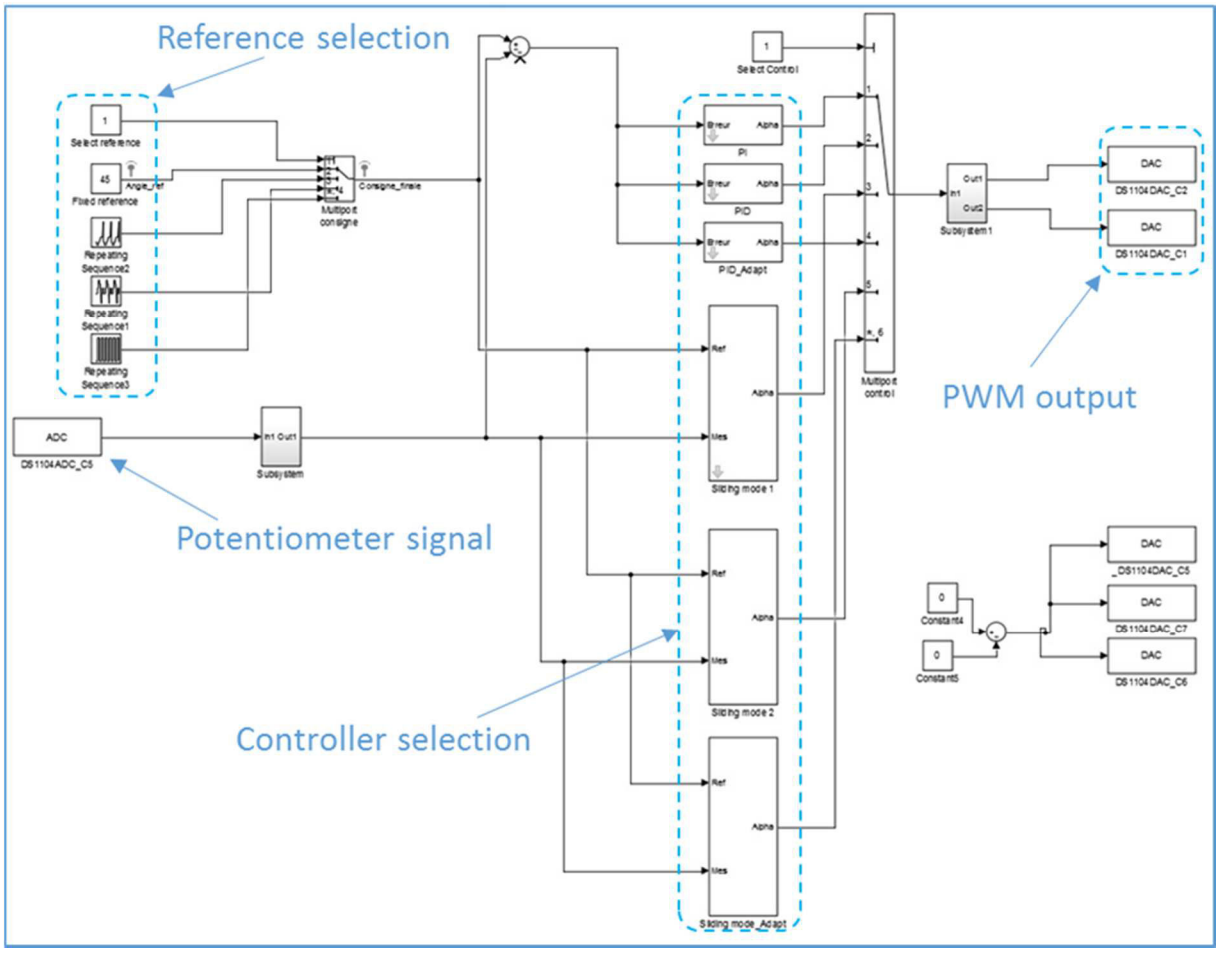



Figure 5-2. Control model of the ETB using Simulink software

### 5.1.3 Fluid model

A CFD model is realized using Fluent software (Figure 5-3). The aim is to simulate the airflow with precision in different angles and find the discharge coefficient for each angle in order to calibrate the Saint-Venant model (Appendix 1). A pipe is realized before and after the valve long enough to avoid turbulence effects and results distortion with parasite vortex (Kumar, Ganesan et al. 2013). The k-ε model was chosen as the majority of previous studies and automatic meshing was used depending on the throttle angle. Therefore, the number of elements was between 440.000 and 500.000. The inlet boundary condition was set to a pressure of 101325Pa and outlet boundary condition was set to the theoretical air-velocity of the vacuum pump for each studying point. The temperature was assumed to be 293K at the boundary condition. The selected fluid was air with a density following ideal gas law.

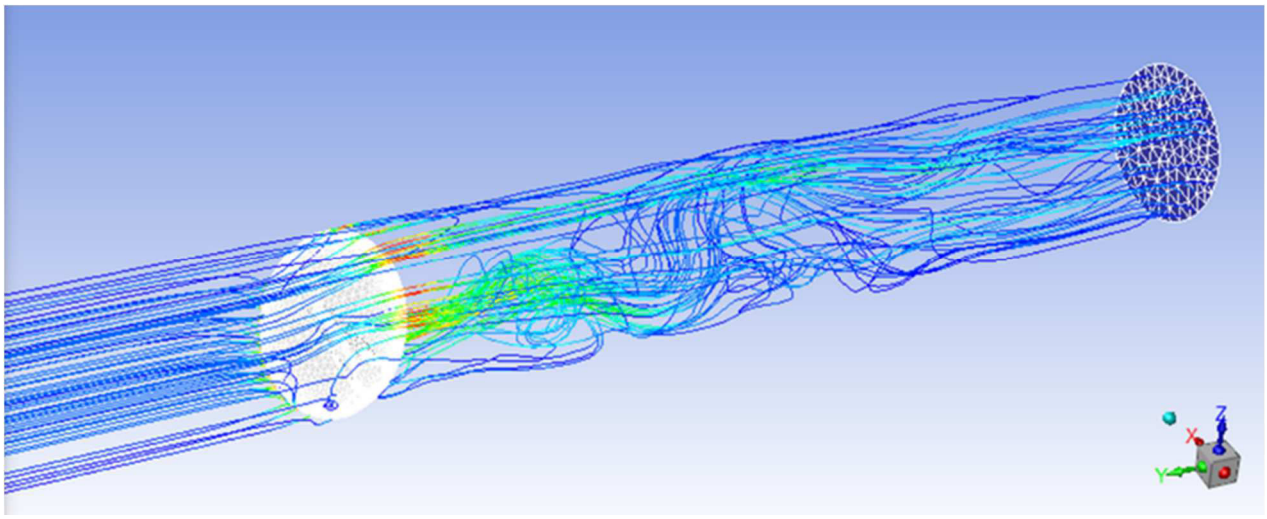


Figure 5-3. Fluid model of the ETB using Ansys

### 5.1.4 Experimental test bench

For experimental measurements an ETB fluid test bench was set up (Figure 5-4). First, we have the inlet straight pipe with 60mm diameter and long enough to avoid any unwanted vortex and have always a laminar flow at the inlet of the ETB (140cm). At the outlet of the ETB, another straight pipe (60 cm) with the same diameter is connected to a vacuum pump that vacuums the gas from the ETB as a real engine. For measurement, a differential pressure sensor is mounted with probe on each part of the ETB. An absolute pressure sensor is placed in the upstream of the ETB. The two pressure sensors are connected with 4 ports to have an average of the pressure in the section. A pitot probe to calculate the air flow is mounted on the upstream pipe to avoid any false measurements due to vortexes generated by the valve since the airflow is considered the same along the pipe. For the ETB control and power supply, a dSPACE card is used with a 12V power source for the DC motor, 5V power source for the potentiometer and an H-Bridge (Amini, Razmara et al. 2017). The control model is loaded in the dSPACE card. The position curve is displayed in real time in the control environment. To summarize, this test bench allows us to validate the dynamic behavior of the ETB, to realize fluid measurements in static positions and to study the behavior of the ETB in real conditions.

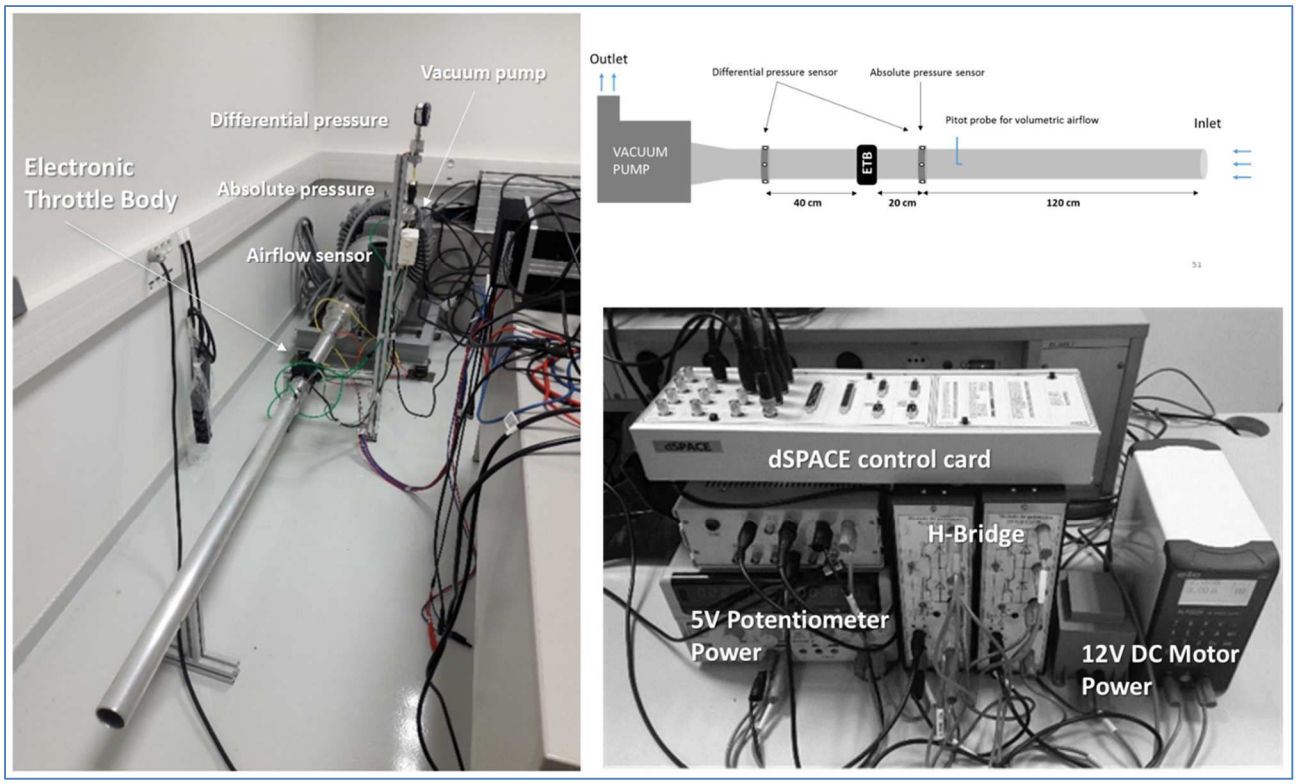



Figure 5-4. Experimental test bench of the ETB

## 5.2 First case study: SE-DE connection

In order to illustrate the effectiveness of our methodology, we present a concrete mechatronic collaborative design scenario to connect SE and DE. The case study refers to an Electronic Throttle Body (ETB) controller. A collaborative scenario implying various fields of expertise is presented in this section. The five steps of CDPPK methodology are applied using the Python demonstrator.

### 5.2.1 Step1: Project Requirements

ETB is a nonlinear mechanism which contains several physical effects, such as friction, the return spring load, and aerodynamic forces, causing a variable time response, an unsteady behavior and static errors. The throttle angle has to be precisely maintained to provide an enhanced throttle response. Thus, the goal is to develop a robust and effective PID controller for Siemens VDO Electronic Throttle Body with the following representative requirements:

- Rising time (from 11,5deg to 90deg) lower than 240ms.
- Return time (from 90deg to 11,5deg) lower than 140ms.
- Static error at maximum aerodynamic torque lower than 0.8deg.

The environment of the system is:

- A pressure drop between 20 and 120mbar.
- Temperature between -20 and 140°C.

The project manager analyzes the requirements and affects the actors of the project.

### 5.2.2 Step2: Conceptual Design

In this step, the system functions are outlined with the associated functional subsystem. In the Design Product Knowledge, system engineers define the following ICEs:

- Motorization function: ICE_DCMotor.
- Control function: ICE_Controller.
- Transmission function: ICE_Gearbox.
- Air regulation function: ICE_Valve.

An ICE_System is also created to affect general requirements concerning all the system like the temperature or the global mass. The controller used here is a proportional-integral-derivative (PID). This type of controllers is widely employed in the industry. However, in practice, it's hard to optimize when conflicting objectives are to be achieved (Ang, Chong et al. 2005). As shown in our demonstrator (Figure 5-5), the first parameters coming from requirements and general knowledge are filled in this step in DPK.

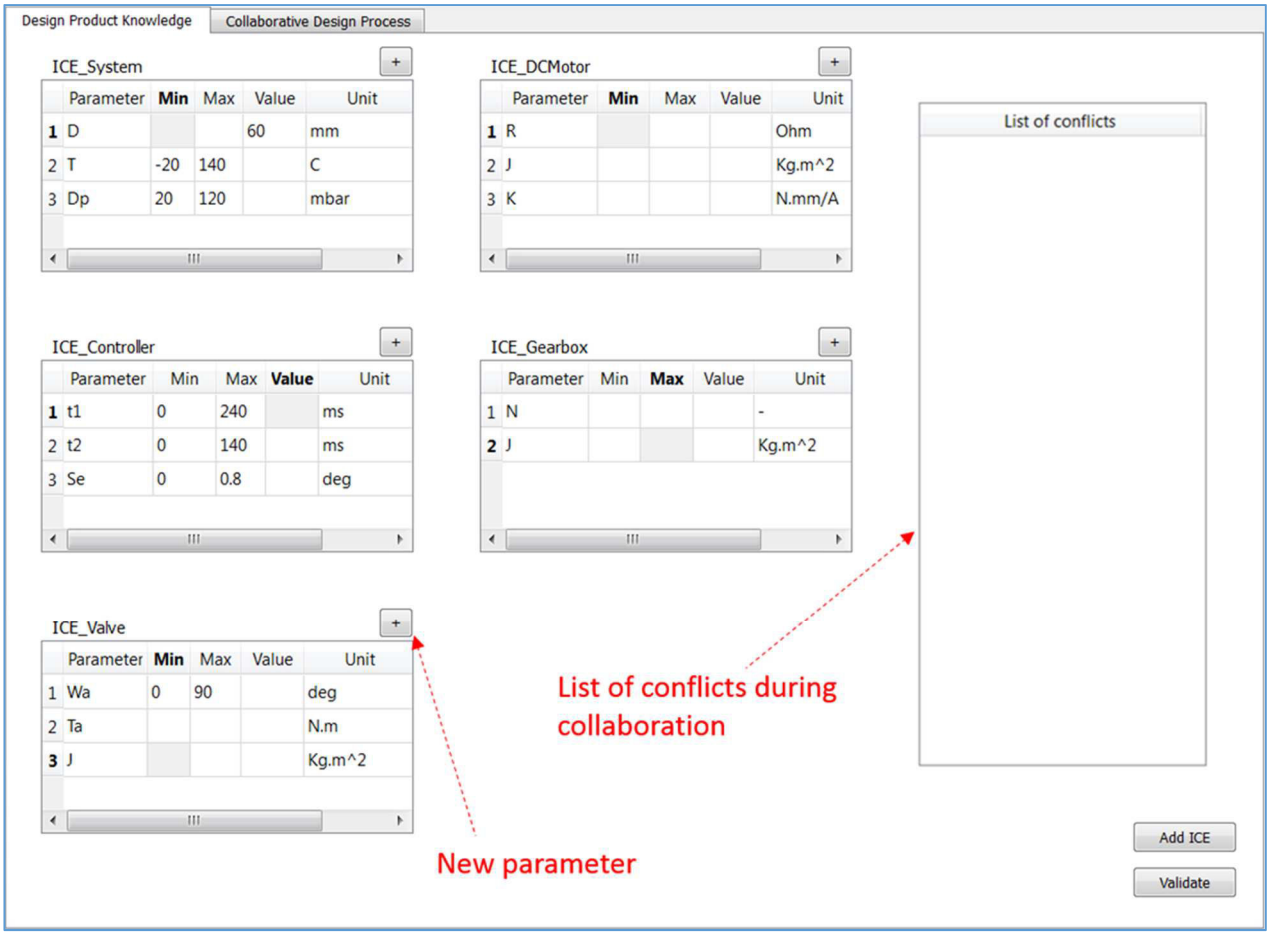



Figure 5-5. ICEs definition in DPK

### 5.2.3 Step3: Detailed Design

In this intermediate step, effective communication between stakeholders is necessary in order to determine all the parameters necessary for the project and to have a common understanding of the design problem. A meeting between system engineers and disciplinary-specific engineers is proposed to fill all the parameters in the DPK structure. Table 5-1 contains all the crucial parameters in this scenario. Some parameters are filled by system engineers in the previous step but the technical parameters are filled by disciplinary engineers. In this step, a refinement of requirements is possible by adding constraints to the newly added parameters.

Table 5-1. The list of crucial parameters and the values found by users

| Parameter | Description | Unit | Min. | Max. | UPC_ Identification | UPC_ Dynamic | UPC_ Fluid | UPC_ Test |
|---|---|---|---|---|---|---|---|---|
| ***ICE_System*** | | | | | | | | |
| *D* | ETB diameter | mm | | | 60 | - | - | |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| *T* | Temperature | °C | -20 | 140 | - | - | - | - |
| *Dp* | Pressure drop | mbar | 20 | 120 | | | 80 | - |
| ***ICE_DCMotor*** | | | | | | | | |
| *R* | Motor Resistance | Ohm | | | 4 | - | | |
| *K* | Motor constant | N.mm/A | | | 16.9 | - | | |
| *L* | Motor inductance | H | | | 1.5e-3 | - | | |
| *J* | Motor inertia | $Kg.m^2$ | | | 1e-6 | - | | |
| *F* | Friction coefficient | N.mm | | | 1.5 | - | | |
| ***ICE_Controller*** | | | | | | | | |
| *P* | Proportional gain | [] | | | | 0.07 | | - |
| *I* | Integral gain | [] | | | | 0.008 | | - |
| *D* | Derivative gain | [] | | | | 0.001 | | - |
| *Se* | Static error at maximum aerodynamic torque | deg | 0 | 0.8 | | 0.59 | | 0.62 |
| $t_1$ | Rising time from 11.5deg to 90deg | ms | 0 | 240 | | 218 | | 215 |
| $t_2$ | Return time from 90deg to 11.5deg | ms | 0 | 140 | | 112 | | 108 |
| ***ICE_Gearbox*** | | | | | | | | |
| *N* | Gearbox ratio | [] | | | 44 | - | | |
| $C_1$ | Main spring stiffness | N.m/rad | | | 0.17 | - | | |
| $C_2$ | Return spring stiffness | N.m/rad | | | 0.82 | - | | |
| *J* | Gearbox inertia | $Kg.m^2$ | | | 0.7e-6 | - | | |
| ***ICE_Valve*** | | | | | | | | |
| $T_a$ | Maximum aerodynamic torque | N.m | | | | - | 0.15 | |
| $W_a$ | Angle at $T_a$ | deg | 0 | 90 | | - | 30 | - |
| *J* | Valve inertia | $Kg.m^2$ | | | 2e-6 | | | |

### 5.2.4 Step4: Verification & Integration

Four disciplinary-specific engineers are involved in this scenario (Figure 5-7):

- UPC_Identification: Traction tests are made to identify the springs' stiffness and calibration test are made to identify the motor and friction parameters.

- UPC_Dynamic: A 0D model in Modelica using Dymola environment studies the dynamic response of the system based on differential equations. This model takes into account the system non-linearity and operating constraints.

- UPC_Fluid: The role of the Fluid model is to identify the maximum aerodynamic torque applied and the associated angle by CFD calculations in Fluent software.

- UPC_Test: The test bench is equipped with dSPACE card for a real-time control of the ETB using Simulink environment. A wind tunnel and pressure measurement instruments are used to adequately reflect tests under real conditions.

As shown in Figure 5-6, each user instantiates his parameters and indicates the in/out flow. Each user has a collaboration state and when the analysis is finished, the user can publish his work and send the values to related collaborators. The collaboration is then managed as explained in section 3.4. The collaborative scenario is composed of four iterative steps (Figure 5-8): (1) The identification of the ETB parameters are made and saved in UPC_Identification. The later sends their values to UPC_Fluid and UPC_Dynamic (2) The fluid calculation are performed to determine the maximum aerodynamic torque and angle. This information is saved in UPC_Fluid, then sent to UPC_Dynamic and UPC_Test to adapt their profile and test the maximum torque angle condition (3) Dynamic calculations are made in Modelica to optimize the PID controller and respect the requirements in term of rising time, return time and static error in maximum aerodynamic torque. The optimized PID parameters are sent to UCP_Test for validation (4) The test bench is used to verify the requirements in real conditions. The experimental values are saved in UPC_Test (5) The validation is made by comparing the results of UPC_Test and UPC_Dynamic as explained in the next section.

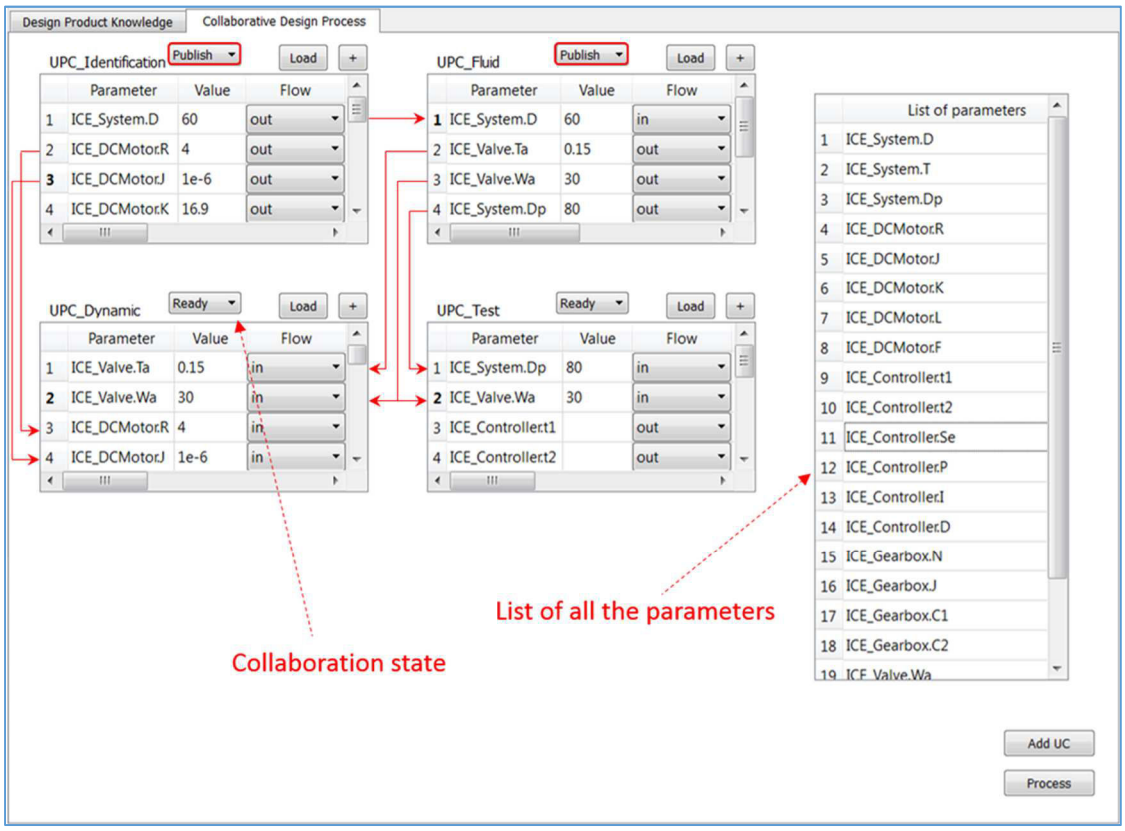



Figure 5-6. UPCs involved in the collaboration

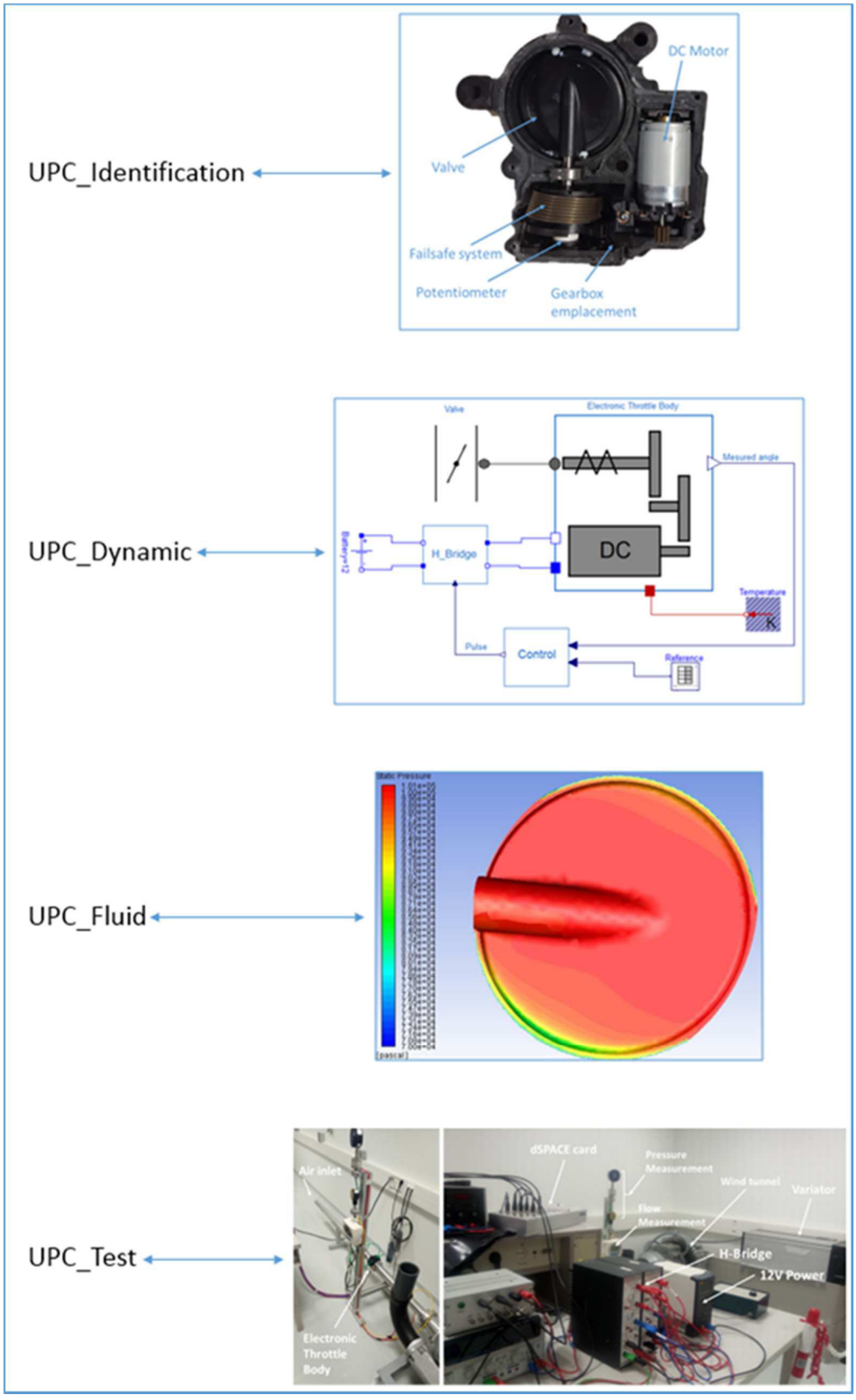



Figure 5-7. The different users involved in the collaborative scenario

### 5.2.5 Step5: System Validation

In this last step, the final results are saved in the PDK and the final collaboration graph is saved in CDP. In order to validate the design, the project manager has to ensure that the simulation results and the experimental validation satisfy the control design requirements (step (5) in Figure 5-8). The project manager has now access to the collaboration process and the product design knowledge. In this scenario, the control system is validated with a good agreement between simulation and experimental results (Columns UPC_Dynamic and UPC_Test in Table 5-1). The system response is also represented in Figure 5-9 and follows the profile to illustrate the rising time, the return time and the aerodynamic effect at 30° when the wind tunnel is activated. This collaborative process leads to the validation of the designed controller. Product and process knowledge are well formalized for future reuse.

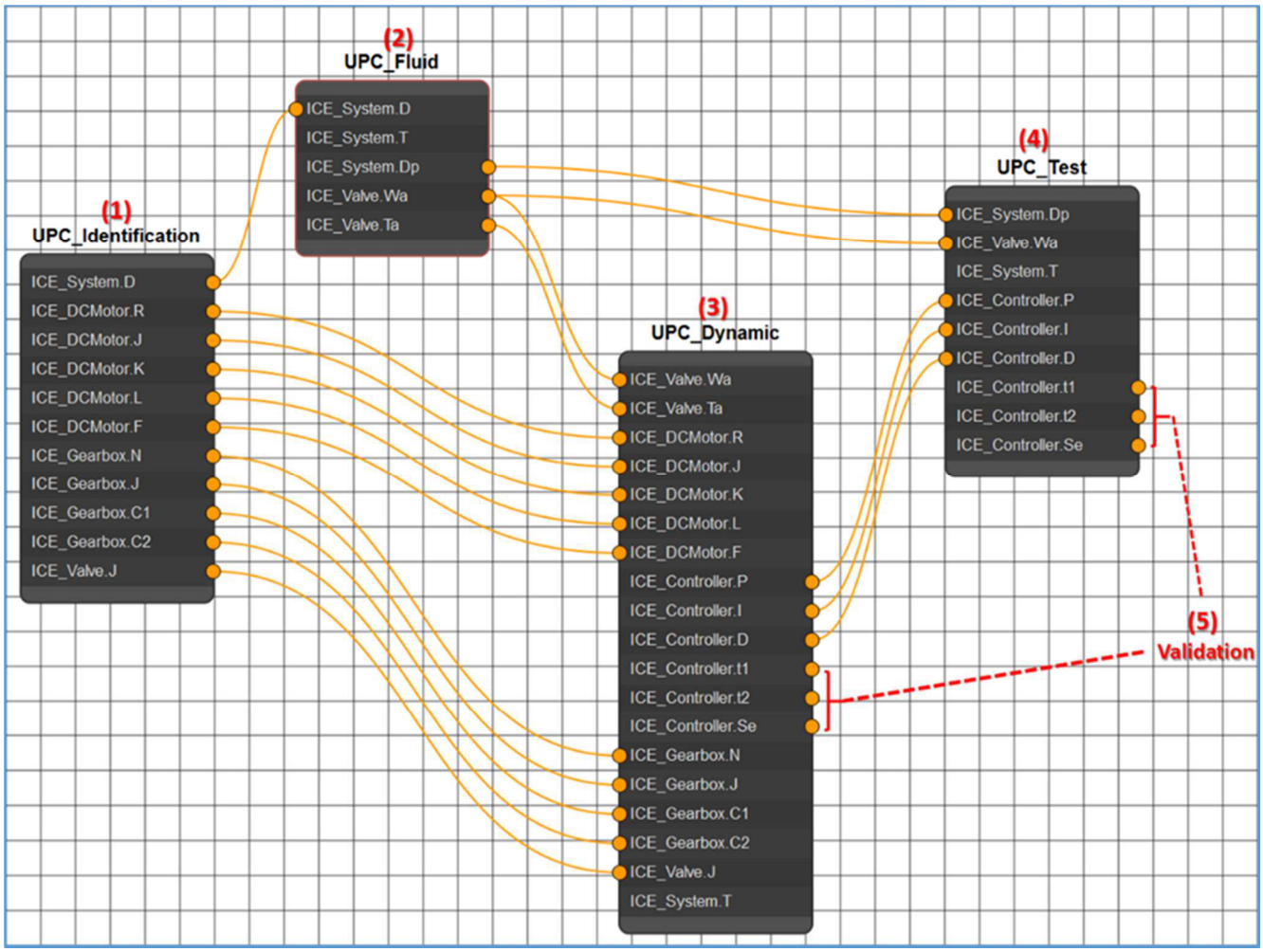



Figure 5-8.Collaboration graph generated by Python demonstrator

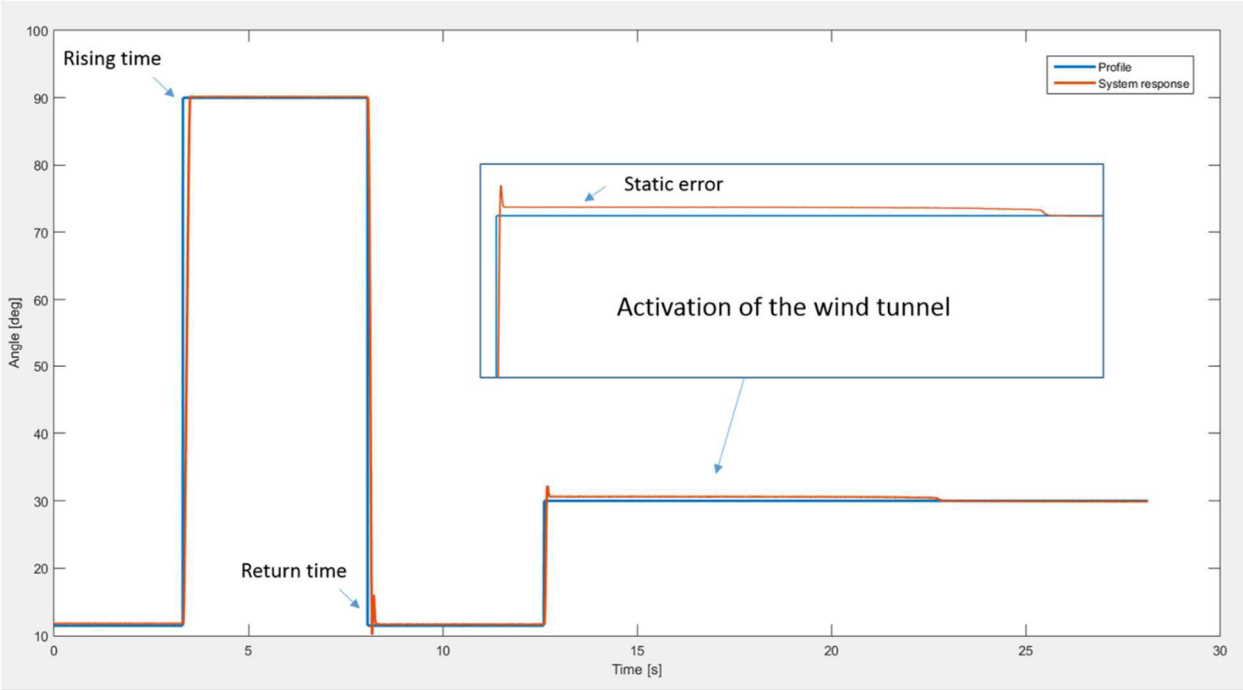



Figure 5-9. The system response to the complete profile reference

## 5.3 Second case study: DE-MDO connection

Starting from the Modelica model identified in the previous section, we propose here a more complicated scenario (more iterative and involving more objectives). The goal of this section is to illustrate the connection between CDPPK and KADMOS. We start from a collaborative process using CDPPK and we convert it automatically into an MDO problem.

### 5.3.1 Requirements

The Euro6 standard is currently the main purpose of all automotive industrials to look for new technologies to reduce emission and enhance driving comfort. Table 5-2 summarizes industrial requirements for ETB. We notice that the requirements are multi-physical (control, electrical, fluid, thermal) with the interaction between these physics (aerodynamic torque on the valve for example). In this section we will evaluate the Siemens VDO model and optimize the sliding mode controller as well as the valve geometry to meet these requirements.

Table 5-2. . Industrial multi-physical requirements for the ETB

| **Parameter** | **Description** | **Conditions** | **Requirements** |
|---|---|---|---|
| **T1 [ms]** | Rising time | From 11.5 deg to 90 deg | < 200 ms |
| **T2 [ms]** | Return time | From 90 deg to 11.5 deg | < 150 ms |
| **S_error [deg]** | Static error | Considering aerodynamic torque | < 0.1 deg |
| **I_max [A]** | Maximum current | At rising phase and T = 23° | < 1.5 A |
| **I_avr [A]** | Average current | In stepped profile (10 deg) | < 0.5 A |
| **Flow_LH [Kg.s$^{-1}$]** | Airflow at limp home position | $P_o/P_i = 0.97$ and $T_i = 23°$ | [0.006 , 0.01]kg/s |
| **Flow_max [Kg.s$^{-1}$]** | Airflow at full open position | $P_o/P_i = 0.97$ and $T_i = 23°$ | > 0.3 kg/s |
| ***Flow_coef* [Kg.s$^{-1}$/deg]** | Fluid progressivity coefficient | $P_o/P_i = 0.85$ and T = 23° | < 0.02 deg |

### 5.3.2 Collaborative process

We defined 4 UPCs to solve the problem defined below:

- UPC_Fluid: The fluid model estimates of the airflow for different angles and pressure ratios (Dm () vector). It estimates also the aerodynamic torque (Tm () vector).
- UPC_Area: The 3D model calculates the airflow section for the different angles (Am () vector) depending on the diameter D and the inclination angle of the valve (In) as illustrated in Figure 5-10

- UPC_Flow: This 0D model calculates the discharge coefficient (Cd () vector) to optimize the Saint Venant Model (Appendix 1). It calculates also F_max, F_LH, F_coef and estimates the inclination angle (In) if these requirements are not reached.
- UPC_Dynamic: The modelica model uses the parameters calculated previously to simulate dynamically the ETB and to optimize the sliding mode controller.

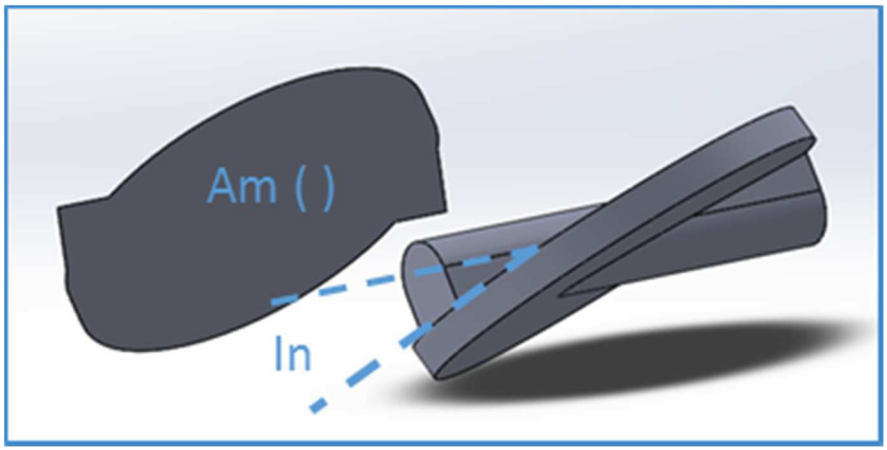



Figure 5-10. 3D model to estimate the airflow section of the ETB

The collaborative process is generated using CDPPK (Figure 5-11). The goal here is to validate the requirements by varying the inclination angle (In) and by optimizing the sliding mode parameters (c1, c2 and c3). The fluid part here affects the control performance, therefore, many iterations are required to reach the requirements. We use this collaborative process to generate an MDO architecture in the next paragraph.

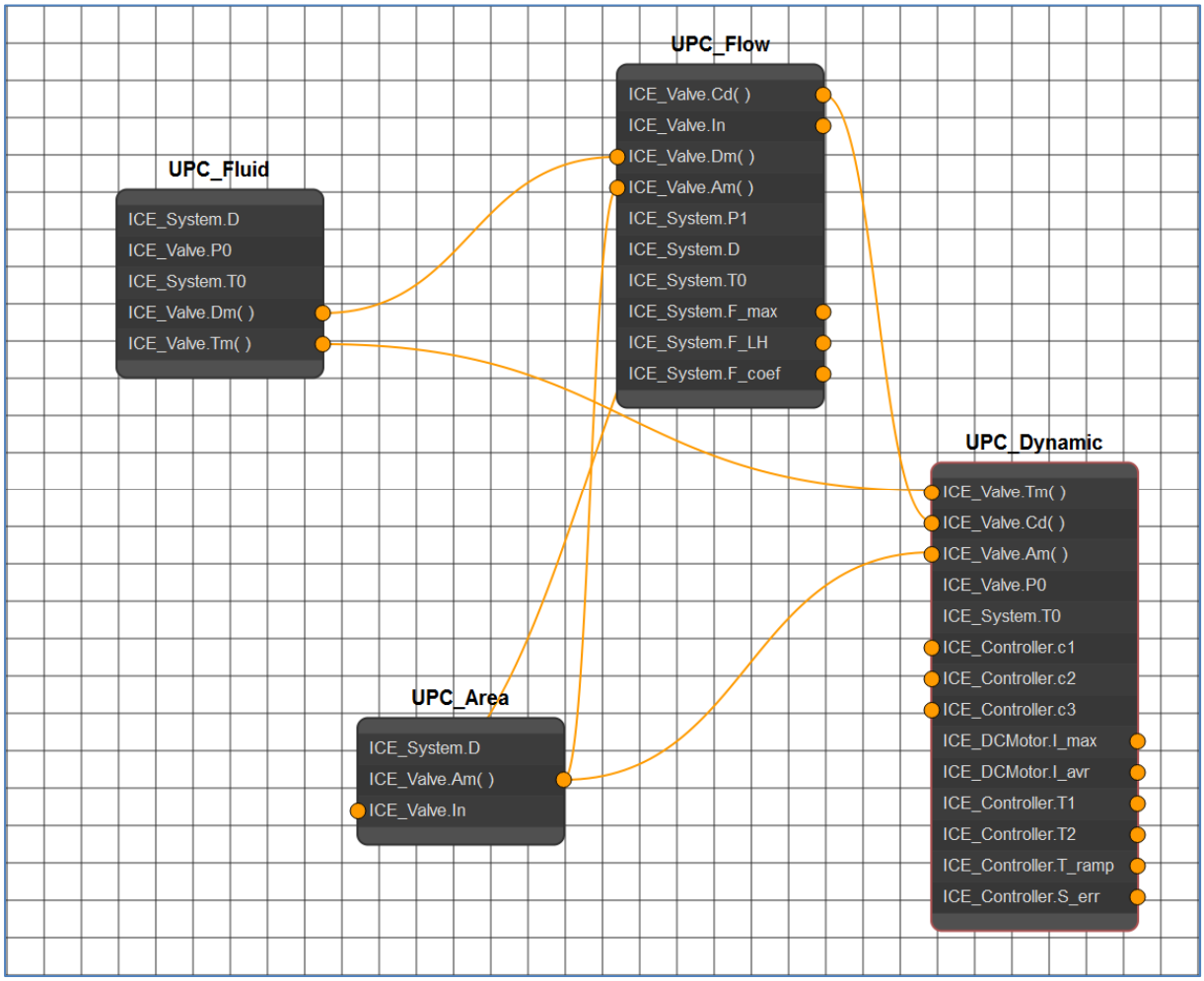



Figure 5-11. Collaborative process generated by CDPPK

### 5.3.3 MDO problem generation

The collaborative process generated by CDPPK can be automatically converted into an MDO problem using CDPPK-KADMOS connection. Figure 5-12 shows the IDF architecture applied to this problem using XDSM generated by KADMOS. In this architecture, UPCs are considered as analyses and the optimization process is controlled by two drivers (coordinator and optimizer).

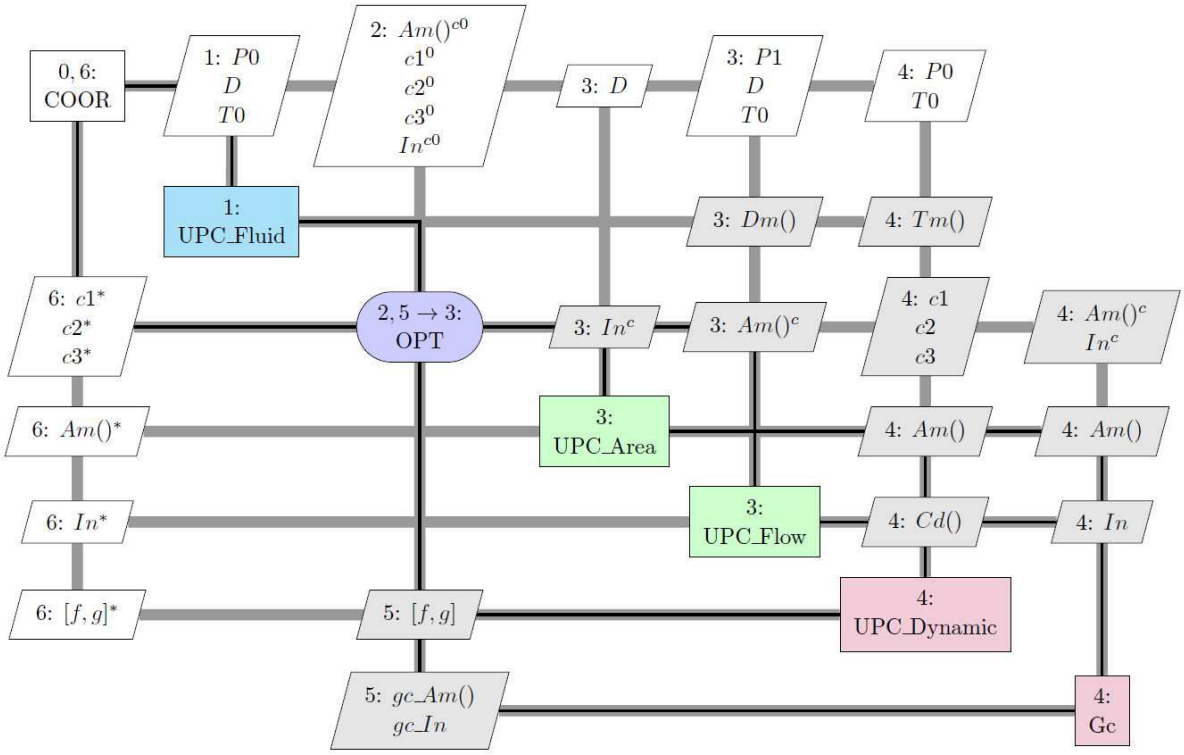



Figure 5-12. XDSM of the IDF architecture applied to the ETB problem

The next step is to automatically create this MDO architecture in RCE software as shown in Figure 5-13. Unfortunately, for a license problem, we did not launch the optimization in RCE software. We used another software for the results as we explain in the next paragraph.

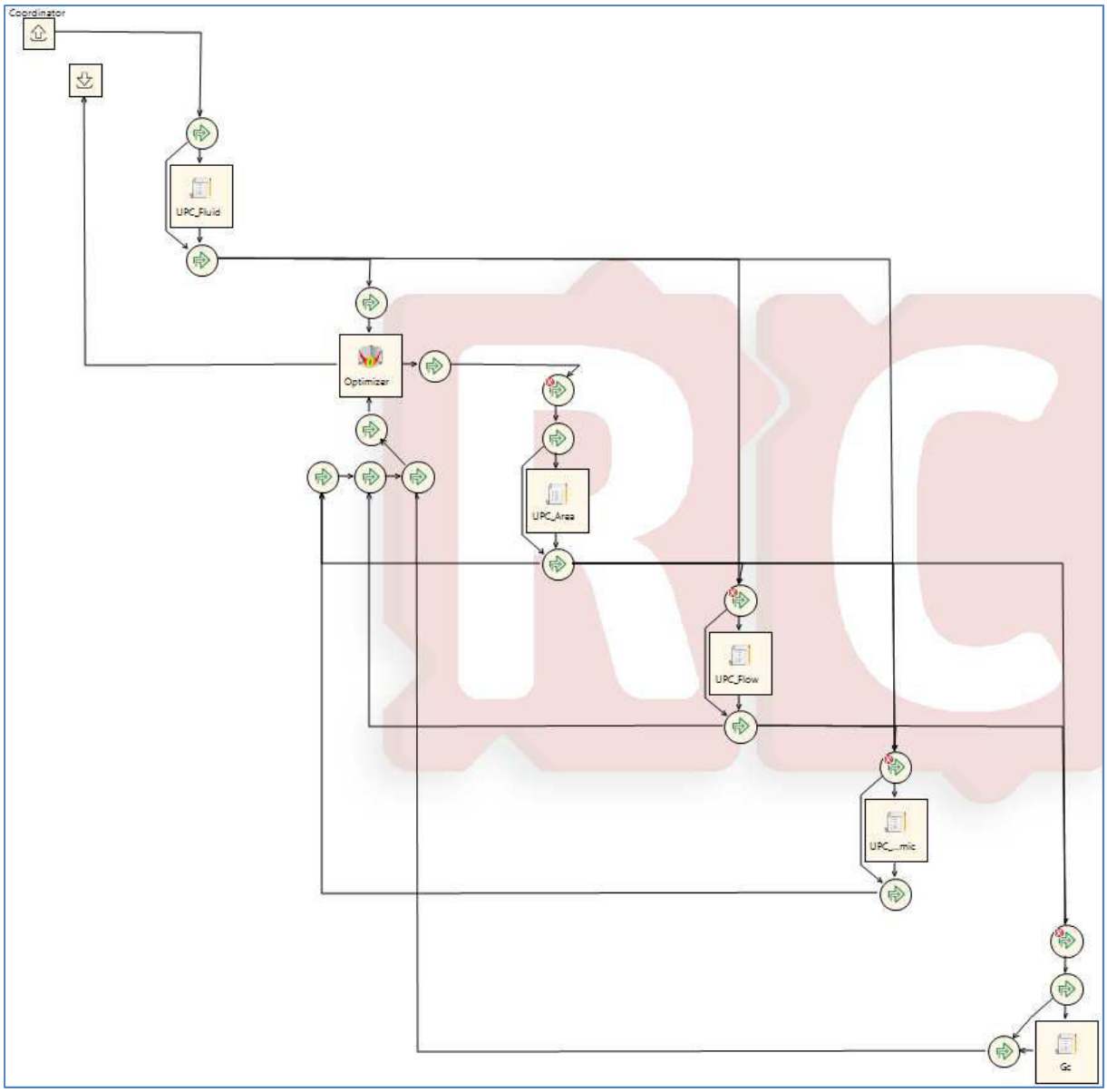



Figure 5-13. Automatic MDO problem generation in RCE environment

### 5.3.4 Results

The results of the optimization are detailed in this section and compared to the requirements presented previously in Table 5-2. First, the mass flow curves and the 3D model calculations are illustrated for different angles “Am ()” (Figure 5-14). Second, the discharge coefficient “Cd ()” that was calculated by minimizing the error between the Saint Venant model and the values

generated by the 3D Fluid model (Figure 5-15). We can notice that the Saint Venant model was calibrated correctly thanks to the enrichment data in Figure 5-16. The progressivity coefficient (F_coef) at the ratio Po/ Pi = 0.85 is calculated using this curve. The fluid behavior of the system respects the requirements.

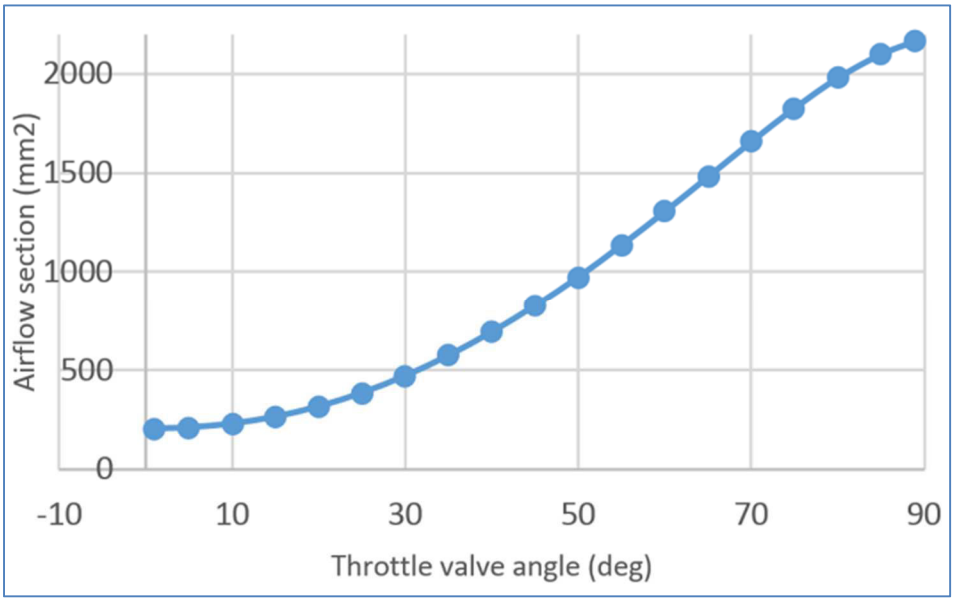



Figure 5-14. Airflow section estimation

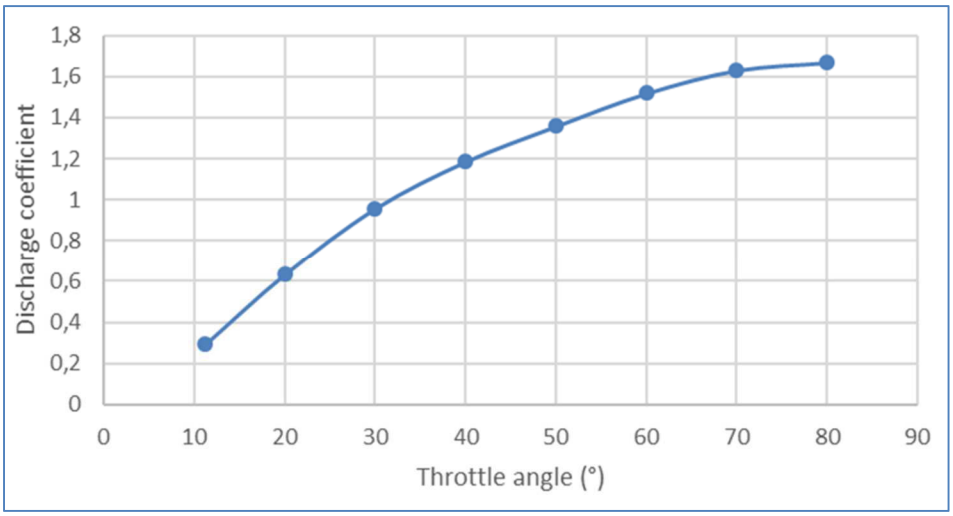



Figure 5-15. Discharge coefficient for each throttle angle

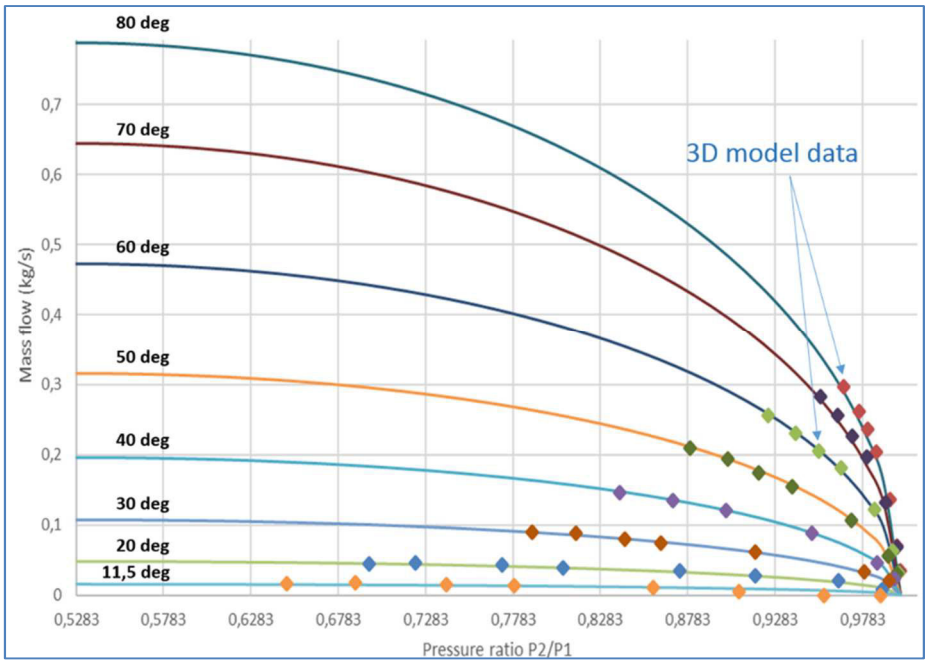



Figure 5-16. Optimized mass flow model

For the control part, position tracking curves are generated with the current. The rising time, the return time and the maximum current are calculated in Figure 5-17. We notice that the response time is respected and that the behavior of the system is stable in the two senses of movement. It is important to guarantee this stability because the system is highly nonlinear and presents different characteristics in the two senses. The Figure 5-18 shows a stepped profile with the associated current. The average current calculated respects the initial requirements and gives an idea about the regular consumption of the ETB. In addition, the optimized controller is robust and works correctly on the different operating points. The average static error considering the aerodynamic torque is less than 0.1 deg. Finally, all the optimization results are regrouped in Table 5-3.

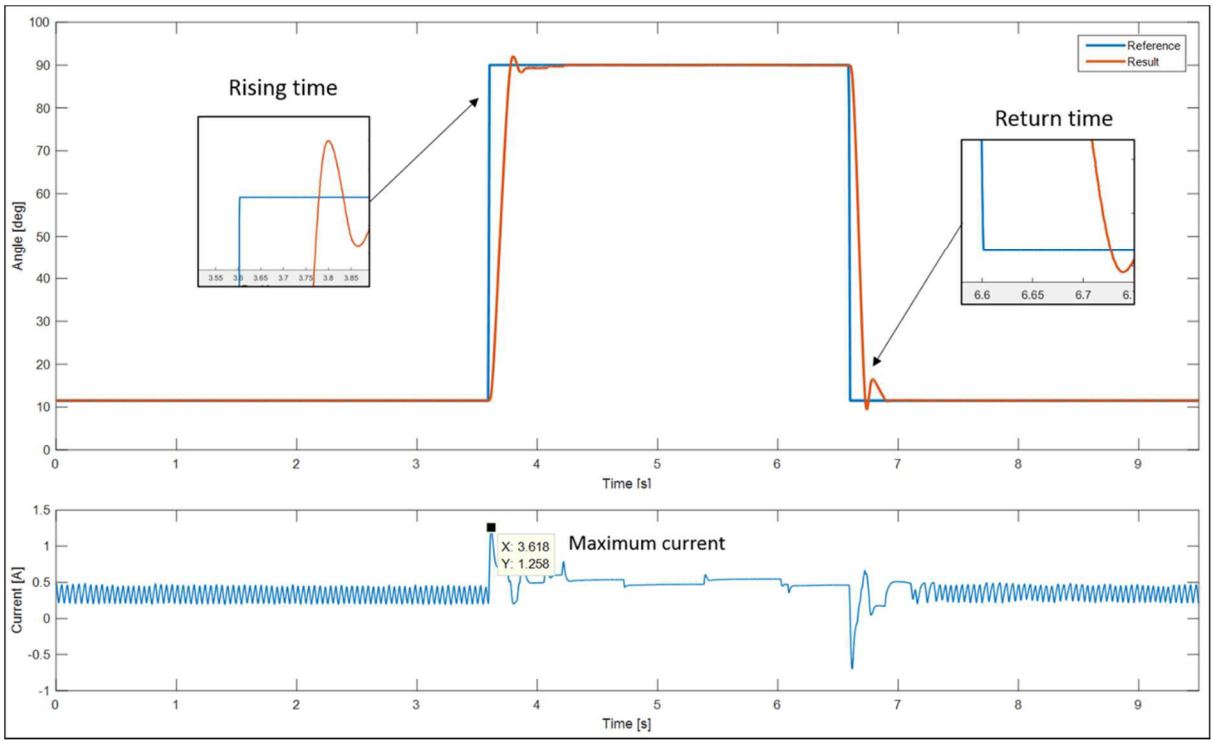



Figure 5-17. Rising time, return time and the maximum current of the ETB

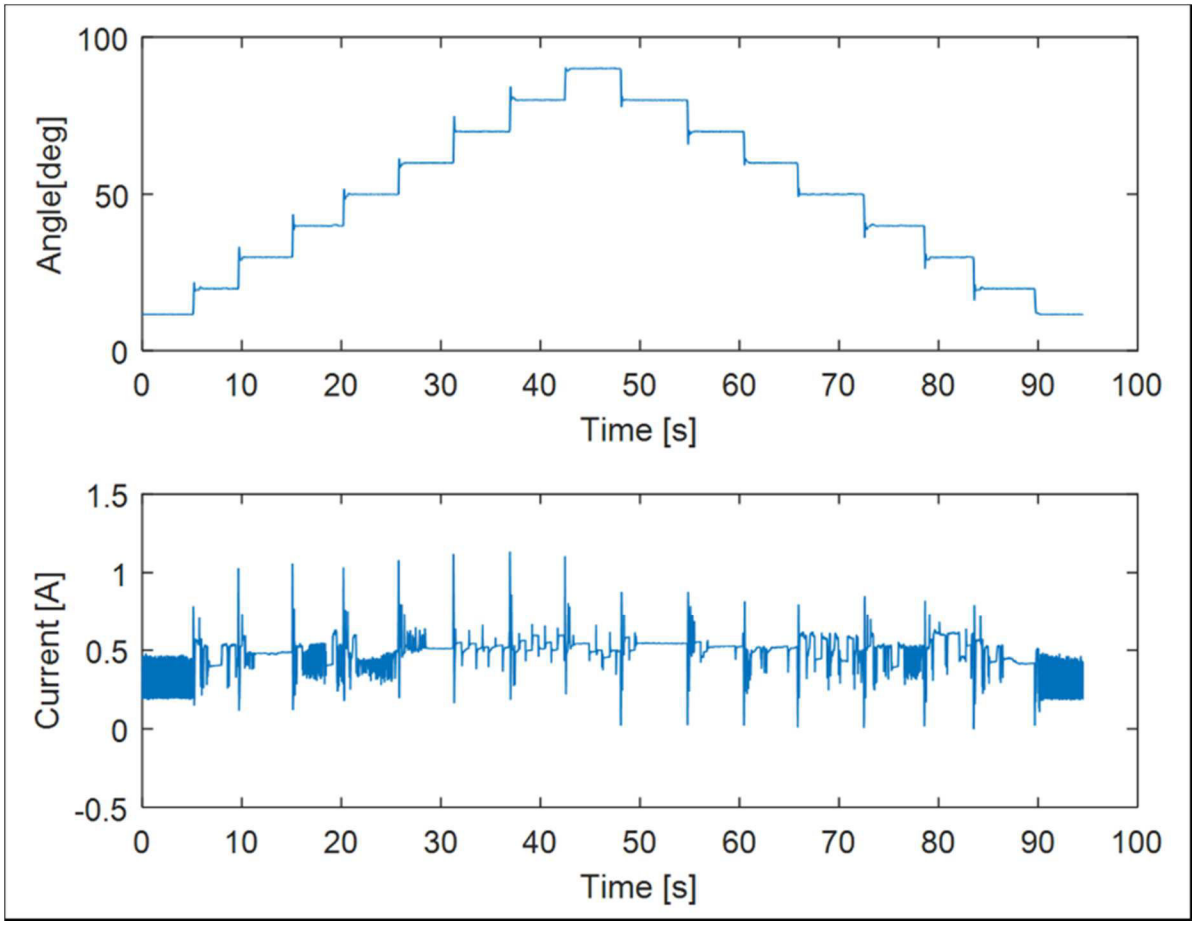



Figure 5-18. Step function response of ETB

Table 5-3. Simulation and experimental results

| Parameter | Constraint | Simulation |
|---|---|---|
| **Requirements** | | |
| ***T1* [ms]** | [0 , 200] | 181 |
| ***T2* [ms]** | [0 , 150] | 134 |
| **S_error [deg]** | [0 , 0.1] | 0.07 |
| **I_max [A]** | [0 , 1.5] | 1.312 |
| **I_avr [A]** | [0 , 0.5] | 0.440 |
| ***Flow_LH* [Kg.s$^{-1}$]** | [0.006 , 0.01] | 0.009 |
| ***Flow_max* [Kg.s$^{-1}$]** | [0.3 , -] | 0.310 |
| ***Flow_coef* [Kg.s$^{1}$/deg]** | [- , 0.02] | 0.015 |
| **Optimized sliding mode and inclination angle** | | |
| **c1 ( = β )** | - | 4.1 |
| **c2 ( = V)** | - | 0.5 |
| **c3 (= γ)** | - | 2.2 |
| **In [deg]** | | 4.5 |

## 5.4 Summary

In this chapter, we illustrated how our approach can be implemented with industrial use cases. We started from industrial requirements and using the disciplinary models and the test bench to realize the collaborative scenarios in CDPPK.

In the first part, we applied step by step the CDPPK methodology starting from the SE view-point. We generated the collaborative design process and we managed the exchanges between disciplinary designers. This first part allowed as to answer the first research question related to the dynamic disciplinary collaboration and system coherence.

In the second part, we were able to generate from a collaborative design process an MDO architecture. This automation is possible thanks to the connection between CDPPK and KADMOS. This second part allowed as to answer the second research question related to automatic MDO problem generation.

# Conclusion

## A. Overview

In this work we tried to answer the questions:

- Q1: How can we ensure a dynamic collaboration at the disciplinary level while remaining coherent with the system level?

- Q2: How can we formalize the knowledge generated during the collaborative design to facilitate reuse and multidisciplinary design optimization?

For the first question, SE-DE connection proposed in CDPPK ensure a dynamic collaboration between disciplinary and system engineers as well as conflict management between disciplinary and system levels.

For the second question, the design process graph generated by CDPPK ensures efficient reuse of the collaborative knowledge and the connection between CDPPK and KADMOS ensures the automatic generation of MDO problem based on the collaborative design knowledge.

Therefore, CDPPK is capable of harmonizing the design cycle by centralizing the connection between SE, DE, and MDO as illustrated in Figure 6-1.

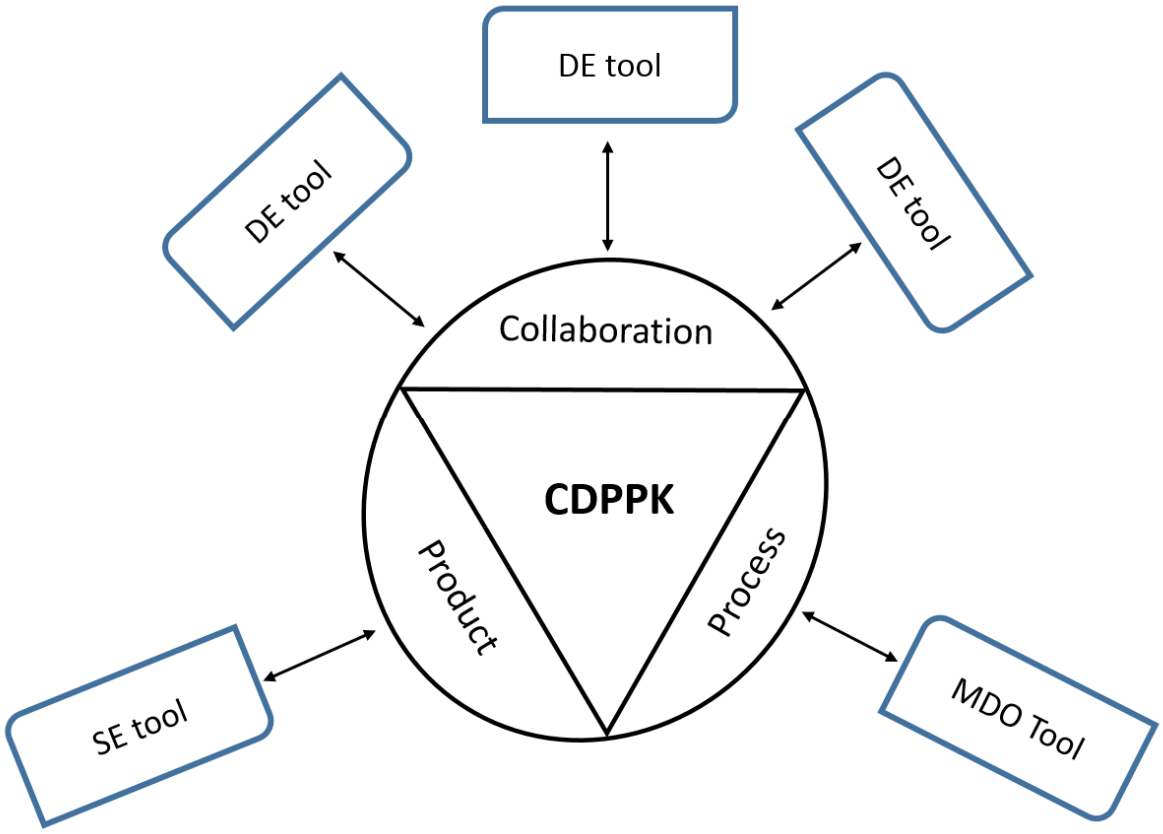



Figure 6-1. CDPPK solution for SE, DE, and MDO

In this context, the presented use case is based on the automotive industrial requirements. Various disciplines are involved in the design of an ETB system. First, by applying the different steps of our methodology, we managed to exchange parameters between stakeholders and to connect the knowledge generated by system engineers and disciplinary engineers. Second, we proposed an MDO problem based on the design process generated by CDPPK. Thereupon, the collaborative scenarios validate the practicality of the approach to connect SE-DE and DE-MDO.

The work carried out during this PhD thesis has brought several academic and industrial contributions.

## B. Contributions

To address the identified scientific challenges we have made the following key contributions:

- SE-DE connection

Traditionally, collaborative methods have no evident added value for solving simple design problems. In such a case, simple direct communication between engineers seems sufficient.

However, when the design involves many parameters and tools, the collaboration becomes time-consuming and conflicts become even impossible to resolve. In practice, stakeholders spend in average 20 percent of the workweek looking for internal information or tracking down other actors who can help with specific tasks (Åman, Handroos et al. 2015). The CDPPK proposes an exchange environment that resolves mechatronic design related conflicts. This is done through interdisciplinary constraints between discipline-specific engineers, and through system constraints between discipline-specific engineers and system engineers. Thus errors, during the design phase, and iterations, between the decomposition and integration phases, are bound to be reduced. The cost and effectiveness benefits cannot be evaluated accurately without testing the methodology in an industrial environment. Nevertheless, estimations show that the use of collaborative tools in the product development phase is likely to improve the revenues in the automotive and aeronautic industries by 0.5-0.7% (Chui, Manyika et al.).

- DE-MDO connection

Conventionally, engineers had to refer to written reports and one or two DE tools, with no need for KM support. However, in multidisciplinary mechatronic design, understanding and reusing the different DE tools are challenging. As reported recently, research indicates that wasted time comprises about 60 percent of the total operational time in most businesses. This is partially due to working with wrong data and unnecessary reinvention of the existing knowledge (Ameri and Dutta 2005). Therefore, all the manufacturing companies dealing with product families and repetitive design task should adopt a knowledge management methodology. In CDPPK, knowledge generated by the collaborative activity is stored into product knowledge and process knowledge in a way that it can be easily reused. On one side, the product knowledge serves as a source of information for system engineers and disciplinary engineers. The system engineers use this knowledge to decompose the product parameters and requirements. This is suitable for their system view. The disciplinary engineers select the parameters that they need to collaborate and find the optimal value. On the other side, the dynamic process knowledge breaks with a traditional predefined process which is no more suitable for concurrent engineering. This type of knowledge, generated with the FPG graph, can be reused in product families design. Stakeholders learn from the recorded conflicts and iterations to reduce them in the next design.

Moreover, CDPPK-KADMOS connection can achieve significant results through automation of complex engineering tasks by automatically generating an MDO problem based on the collaborative design process.

### C. Limits

Concerning the limits of our approach, we can notice that the step 3 of CDPPK methodology (formalizing knowledge) might be time-consuming. It requires a close exchange between the different actors of the design cycle. We suggested in step 3 of our methodology a meeting to formalize the parameters needed for the collaboration. The project manager, the system engineers, and the disciplinary engineers must have a close exchange to form a holistic product knowledge. But this can be difficult to organize in complex and distributed environments. Automated ontology solutions to extract knowledge exist and can be incorporated in this step. Either way, the benefits of the formalized knowledge are proven in the long term. The structured knowledge is highly valuable for companies because it can be easily accessed, mined and used for decision making.

Lately, knowledge risks have gained increasing attention in the information security related literature. Indeed, knowledge can have a competitive value which makes it at risk of inter-organizational collaboration (Ilvonen, Jussila et al. 2015). One must note that such risks are already present in traditional solutions (e.g. email social platforms and face-to-face conversations). The CDPPK proposes, in this context, to grant each actor special access privileges to reduce the probability of such threats.

For people, there is an inconvenience when adopting new collaborative and reuse tools for they require training and time to be assimilated. Therefore, to avoid rejection, the concept should be simple and easy to master. Our CDPPK methodology has not been proposed on an industrial level yet, but feedbacks from industrials in our project give us good insights.

The other people related limit is their acceptance to share their knowledge. Research indicates that, in a typical organization, only 4% of organizational knowledge is available in a structured and reusable format and the rest is either unstructured or resides in people's minds (Rasmus 2002). Therefore, the successful development of such tools requires the ability of the dedicated design team to communicate, collaborate and integrate their knowledge and know-how (Assouroko, Ducellier et al. 2014).

## D. Perspectives

Based on the discussions with industrials of our project, we estimate the following perspectives:

- The next step of our work is to upgrade the prototype for web-service applications to test it in a distributed industrial environment. This will allow us to have actual feed-backs from engineers.

- We plan to include an Agent-based solution supporting the exchange and assisting actors to avoid conflicts. This technology can be integrated into our methodology as well as constraint satisfaction techniques to advise stakeholders during collaboration.

- We plan to use state-of-the-art graph querying tools to facilitate the access to the knowledge and facilitate the reuse phase

- The possibility to define different architectures of the same system and generate a tradeoff based on common criteria. The tradeoff can be envisaged between KCs.

# Appendixes

# Appendix 1: Analytic model of the ETB

According to Kirchhoff's law, the DC motor circuit can be described as the following equation:

$$U_m(t) = L_m \frac{di_m(t)}{dt} + R_m i_m(t) K_e \Omega_m(t) \quad (1)$$

The voltage at the H-bridge output depending on the duty cycle is modeled by this equation:

$$U_m = (2\alpha - 1)U \quad (2)$$

Using the torque balance law, the kinematic equation of the DC motor is:

$$J_m \frac{d\Omega_m(t)}{dt} = K_m i_m(t) - K_e \Omega_m(t) \quad (3)$$

The gearbox reduction equations are:

$$T_m(t) = N T_v(t) \quad (4)$$

$$\Omega_m(t) = N \Omega_v(t) \quad (5)$$

The torque balance law is finally applied in the valve level:

$$J_v \frac{d\Omega_v(t)}{dt} = T_v(t) - T_r(t) \quad (6)$$

Where the resistive forces $T_r(t)$ are:

$$T_r(t) = T_s(t) + T_f(t) + T_a(t) \quad (7)$$

$i_m$ and $U_m$ are the motor current and voltage, $L_m$ and $R_m$ are the armature inductance and resistance, $E_m$ is electromotive force (EMF), $\Omega_m$ is the motor speed, $K_m$ and $K_e$ are EMF constant and torque factors, $J_m$ is the inertia of motor, $U$ is the input voltage and $\alpha$ represents duty cycle. The gearbox ratio is $N$, the valve speed and inertia are $\Omega_v$ and $J_v$. The torques cited are the motor torque $T_m$, the torque after reduction at the valve level $T_v$, and the resistive torque $T_r$ . This resistive torque is composed of springs torque $T_s$, friction torque $T_f$ and aerodynamic torque $T_a$ which will be detailed later in the next section. The thermal effects are also considered on the resistance and the electromotive force by the following equations ($\alpha_1$ and $\alpha_2$ are linear coefficients):

$$R_m(T) = R_m(T_0) * (1 + \alpha_1(T - T_0)) \quad (8)$$

$$K_m(T) = K_m(T_0) * (1 + \alpha_2(T - T_0)) \quad (9)$$

Then, the flow through the throttle valve is modelled by a steady isentropic compressible flow (Alsemgeest 2004). The most used model in literature is the Barré de Saint-Venant model which give us the mass flow rate for chocked and non-chocked flow (Bordjane and Chalet 2015):

If $\left(\frac{P_o}{P_i}\right) \geq \left(\frac{2}{\gamma+1}\right)^{\frac{\gamma}{(\gamma-1)}}$ , then the non-chocked-flow is represented by:

$$\dot{m} = C_d \times A(\theta) \times \frac{P_i}{\sqrt{R \times T_i}} \times \left(\frac{P_o}{P_i}\right)^{\frac{1}{\gamma}} \times \sqrt{\frac{2\gamma}{\gamma-1} \times \left(1 - \frac{P_o}{P_i}\right)^{\frac{(\gamma-1)}{\gamma}}} \quad (10)$$

If $\left(\frac{P_o}{P_i}\right) < \left(\frac{2}{\gamma+1}\right)^{\frac{\gamma}{(\gamma-1)}}$, then the chocked-flow is represented by:

$$\dot{m} = C_d(\theta) \times A(\theta) \times \frac{P_i}{\sqrt{R \times T_i}} \times \sqrt{\gamma} \times \left(\frac{2}{\gamma+1}\right)^{\frac{\gamma+1}{2\times(\gamma-1)}} \quad (11)$$

With $T_i$and $P_i$ the temperature and total pressure at the inlet, $P_o$ the total pressure at the outlet, $\gamma$ is the polytropic coefficient fixed to 1.4, $R$ the universal constant equal to 8,314 $J.mol^{-1}.K^{-1}$, $A(\theta)$ the airflow section and $C_d(\theta)$ the discharge coefficient which is an empirical coefficient to correct the limitations of the 1D hypothesis.

# Appendix 2: Sliding mode control

Many sources of nonlinearities exist in the ETB system and linear control tools (PD, PID, LQR…) may fail in some cases. To improve the response of the controlled system, and avoid fast transient variations, we looked to use the second order sliding mode control (Bai 2018).

The sliding surface is defined as:

$$S(t) = \varepsilon(t) + \tau \frac{d\varepsilon(t)}{dt} \tag{12}$$

With $\varepsilon(t) = \Omega_m - \Omega_{ref}$ as tracking error and $\tau$ a control parameter that determine the rate of convergence of the error to zero inside the sliding surface $S = 0$.

The considered control parameter is $u = U_m$ which generates the real control parameter given by the duty cycle ($\alpha$) according to the equation (2).

After some calculations, we can find that the derivative of $S$ can be written as follows

$$\dot{S} = \Phi(.) + \varphi(.)u \tag{13}$$

Where $\Phi(.)$ and $\varphi(.)$ are bounded functions whose expressions are given in (Azib, Talj et al. 2010). This characteristic of the system motivates us to use the second order sliding mode, super-twisting algorithm, known for its robustness against perturbation and parameters uncertainties. The control parameter $u$ is defined as:

$$u = u_1 + u_2 \tag{14}$$

Where:

$$\dot{u}_1(t) = -\beta \; sign\,(S) \tag{15}$$

$$u_2(t) = -\gamma |S|^v \; sign\,(S) \tag{16}$$

With:

$$\beta > \frac{C_0}{h_m}, \quad \gamma^2 \geq \frac{4C_0 h_M(\beta + C_0)}{h_m^3(\beta - C_0)}, \quad 0 < v < 0{,}5 \tag{17}$$

Where $C_0$, $h_m$ and $h_M$ are constants depending on the system such that:

$$|\emptyset(.)| < C_0, \quad h_m < \varphi(.) < h_M$$

# Appendix 3: French summary

**Résumé:**

Le contexte général des travaux de thèse concerne la conception collaborative de systèmes mécatroniques avec pour objectif de formaliser le processus de conception afin de pouvoir le réutiliser de manière efficace. Le problème de discontinuité entre l'ingénierie système (IS), l'ingénierie disciplinaire (ID) et l'optimisation multidisciplinaire (OMD) nous a conduit à proposer une nouvelle approche fondée sur la gestion des connaissances cruciales pour la conception mécatroniques.

En industrie, le problème de discontinuité est lié à plusieurs raisons. Premièrement, les outils de collaboration actuels ne garantissent pas la traçabilité et la résolution dynamique de conflits interdisciplinaires. Par conséquent, les connaissances générées pendant la collaboration ne peuvent pas être réutilisées. Deuxièmement, l'hétérogénéité entre les outils IS et ID complique le processus de validation des exigences. Troisièmement, la définition d'un problème OMD prend beaucoup de temps et reste incompatible avec l'ingénierie concurrente.

Dans ce travail, nous proposons d'utiliser des techniques de gestion des connaissances pour assurer la collaboration entre les acteurs, en améliorant la capitalisation, la traçabilité et la réutilisation des connaissances utilisées. Cette approche s'appelle « Collaborative Design Process and Product Knowledge » (CDPPK). Elle permet de centraliser les connaissances nécessaires à la collaboration et au suivi des exigences. Elle assure également la traçabilité des échanges entre les ingénieurs grâce à la théorie des graphes. Cette connaissance formalisée du processus de collaboration permet par la suite de définir automatiquement un problème OMD.

Pour valider notre approche, un démonstrateur a été mis en place et appliqué au système boitier papillon motorisé (BP). Des modèles multidisciplinaires du BP et un banc d'essai expérimental ont été réalisés. Ensuite, des scénarios de collaboration industrielle ont été reconstitués avec succès en utilisant notre méthodologie.

**Structure:**

La thèse commence par une introduction générale du contexte et du projet MIMe. Ensuite, la problématique de recherche est traitée sur 5 chapitres :

- Chapitre 1: Ce chapitre présente un état de l'art sur la description des systèmes mécatronique et les difficultés rencontrées par les concepteurs. Nous définissons IS, ID et OMD avec des exemples réalisés sur le boitier papillon motorisé.

- Chapitre 2: Ce chapitre présente l'état de l'art en ce qui concerne les solutions de gestion de connaissances existantes pour prendre en charge IS, ID et OMD. Nous proposons des critères de comparaison pour justifier notre choix et positionner nos travaux.

- Chapitre 3: Le modèle de configuration de connaissances (KCM) décrit dans le chapitre 2 est analysé en détail dans ce chapitre. Nous proposons d'étendre cette méthodologie afin de l'appliquer à la conception mécatronique grâce à des connecteurs IS-ID et ID-OMD. Cette méthodologie est par la suite appliquée à un cas d'étude industriel pour monter les limites pratiques et théoriques.

- Chapitre 4: Basé sur les limites de KCM, notre nouveau modèle est présenté dans ce chapitre. Le modèle CDPPK (Process Design Design and Product Knowledge) et la méthodologie associée sont expliqués pour répondre à notre problématique de la recherche. Un démonstrateur Python est présenté avec les connexions IS-ID et ID-MDO.

- Chapitre 5: Ce chapitre valide l'approche CDPPK. Nous définissons en détail le cycle de vie de la conception ETB dans l'industrie. Les modèles multidisciplinaires mis en œuvre et le banc d'essai expérimental sont présentés. La méthodologie est appliquée dans deux scénarios industriels de collaboration pour valider les connexions IS-ID et ID-OMD.

Le document se termine par une conclusion générale et des perspectives des travaux menés. Dans cette conclusion, nous mettons en avant les contributions apportées en termes de méthodes collaboratives et de formalisation des connaissances ainsi que les limites de l'approche et les extensions possibles.

# References

Abid, H., P. Pernelle, D. Noterman, J.-P. Campagne and C. Ben Amar (2015). "SysML approach for the integration of mechatronics system within PLM systems." International Journal of Computer Integrated Manufacturing **28**(9): 972-987.

Alavi, M. and D. E. Leidner (2001). "Knowledge management and knowledge management systems: Conceptual foundations and research issues." MIS quarterly: 107-136.

Alexandrov, N. (2005). "multidisciplinary design optimization." Optimization and Engineering **6**(1): 5-7.

Ali, M. I. O. (2016). Proposition d'un modèle produit liant la maquette numérique aux données hétérogènes d'un grand ensemble mécanique, Ecole centrale de Nantes.

Alsemgeest, R. W. (2004). The time-dependent flow through throttle valves: a computational and experimental investigation, University of Warwick.

Åman, R., H. Handroos, H. Kärkkäinen, J. Jussila and P. Korkealaakso (2015). Novel ICT-Enabled Collaborative Design Processes and Tools for Developing Non-Road Mobile Machinery. ASME/BATH 2015 Symposium on Fluid Power and Motion Control, American Society of Mechanical Engineers.

Amann, K. (2002). "Product lifecycle management: empowering the future of business." CIMdata, Inc.

Ameri, F. and D. Dutta (2005). "Product lifecycle management: closing the knowledge loops." Computer-Aided Design and Applications **2**(5): 577-590.

Amini, M. R., M. Razmara and M. Shahbakhti (2017). "Robust model-based discrete sliding mode control of an automotive electronic throttle body." SAE International Journal of Commercial Vehicles **10**(2017-01-0598): 317-330.

Ammar, R., M. Hammadi, J.-Y. Choley, M. Barkallah, J. Louat and M. Haddar (2017). Architectural design of complex systems using set-based concurrent engineering. Systems Engineering Symposium (ISSE), 2017 IEEE International, IEEE.

Ang, K. H., G. Chong and Y. Li (2005). "PID control system analysis, design, and technology." IEEE transactions on control systems technology **13**(4): 559-576.

Arduin, P.-E., M. Grundstein and C. Rosenthal-Sabroux (2013). "From knowledge sharing to collaborative decision making." International Journal of Information and Decision Sciences **5**(3): 295-311.

Assouroko, I., G. Ducellier, P. Boutinaud and B. Eynard (2014). "Knowledge management and reuse in collaborative product development–a semantic relationship management-based approach." International Journal of Product Lifecycle Management **7**(1): 54-74.

Azib, T., R. Talj, O. Bethoux and C. Marchand (2010). Sliding mode control and simulation of a hybrid fuel-cell ultracapacitor power system. Industrial Electronics (ISIE), 2010 IEEE International Symposium on, IEEE.

Badin, J. (2011). Ingénierie hautement productive et collaborative à base de connaissances métier: vers une méthodologie et un méta-modèle de gestion des connaissances en configurations, Université de Technologie de Belfort-Montbeliard.

Badin, J., D. Chamoret, S. Gomes and D. Monticolo (2011). Knowledge configuration management for product design and numerical simulation. DS 68-6: Proceedings of the 18th International Conference on Engineering Design (ICED 11), Impacting Society through Engineering Design, Vol. 6: Design Information and Knowledge, Lyngby/Copenhagen, Denmark, 15.-19.08. 2011.

Badin, J., D. Monticolo, D. Chamoret and S. Gomes (2011). "Using the knowledge configuration model to manage knowledge in configuration for upstream phases of the design process." International Journal on Interactive Design and Manufacturing (IJIDeM) **5**(3): 171.

Bai, R. (2018). "Adaptive sliding-mode control of automotive electronic throttle in the presence of input saturation constraint." IEEE/CAA Journal of Automatica Sinica **5**(4): 878-884.

Bajaj, M., D. Zwemer, R. Peak, A. Phung, A. Scott and M. Wilson (2011). 4.3. 1 Satellites to Supply Chains, Energy to Finance—SLIM for Model-Based Systems Engineering: Part 1: Motivation and Concept of SLIM. INCOSE international Symposium, Wiley Online Library.

Bandler, J. W., R. M. Biernacki, S. H. Chen, P. A. Grobelny and R. H. Hemmers (1994). "Space mapping technique for electromagnetic optimization." IEEE Transactions on Microwave Theory and Techniques **42**(12): 2536-2544.

Barton, R. R. and M. Meckesheimer (2006). "Metamodel-based simulation optimization." Handbooks in operations research and management science **13**: 535-574.

Belkadi, F., N. Dremont, A. Notin, N. Troussier and M. Messadia (2012). "A meta-modelling framework for knowledge consistency in collaborative design." Annual Reviews in Control **36**(2): 346-358.

Biahmou, A. (2015). Systems Engineering. Concurrent Engineering in the 21st Century, Springer**:** 221-254.

Bondy, J. A. and U. S. R. Murty (1976). Graph theory with applications, Citeseer.

Borches, P. D. and G. M. Bonnema (2010). System Evolution Barriers and How to Overcome Them! 8th Conference on Systems Engineering Research (CSER2010).

Bordjane, M. and D. Chalet (2015). "Numerical Investigation Of Throttle Valve Flow Characteristics For Internal Combustion Engines." Journal of Multidisciplinary Engineering Science and Technology (JMEST) **2**(12).

Bruun, H. P. L. and N. H. Mortensen (2012). Visual product architecture modelling for structuring data in a PLM system. IFIP International Conference on Product Lifecycle Management, Springer.

Cabrera, A. A., M. Foeken, O. Tekin, K. Woestenenk, M. Erden, B. De Schutter, M. Van Tooren, R. Babuška, F. J. van Houten and T. Tomiyama (2010). "Towards automation of control software: A review of challenges in mechatronic design." Mechatronics **20**(8): 876-886.

Cencetti, M. (2016). Evolution of model-based system engineering methodologies for the design of space systems in the advanced stages of the project, Torino: Politecnico di Torino,(Ph. D. Thesis). Dassault Systemes.

Chami, M. and J.-M. Bruel (2015). "Towards an integrated conceptual design evaluation of mechatronic systems: The SysDICE approach." Procedia Computer Science **51**: 650-659.

Chandrasegaran, S. K., K. Ramani, R. D. Sriram, I. Horváth, A. Bernard, R. F. Harik and W. Gao (2013). "The evolution, challenges, and future of knowledge representation in product design systems." Computer-aided design **45**(2): 204-228.

Chapman, C. B. and M. Pinfold (2001). "The application of a knowledge based engineering approach to the rapid design and analysis of an automotive structure." Advances in Engineering Software **32**(12): 903-912.

Chui, M., J. Manyika, J. Bughin, R. Dobbs, C. Roxburgh, H. Sarrazin, G. Sands and M. Westergren The Social Economy: Unlocking Value and Productivity through Social Technologies, McKinsey Global Institute. 2012.

Ciampa, P. D. and B. Nagel (2017). The AGILE Paradigm: the next generation of collaborative MDO. 18th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference.

Commission, E. A. (1999). "Criteria for accrediting engineering programs." Accreditation Board for Engineering and Technology Inc.

Corno, M., M. Tanelli, S. M. Savaresi and L. Fabbri (2011). "Design and validation of a gain-scheduled controller for the electronic throttle body in ride-by-wire racing motorcycles." IEEE Transactions on Control Systems Technology **19**(1): 18-30.

Cramer, E. J., J. Dennis, John E, P. D. Frank, R. M. Lewis and G. R. Shubin (1994). "Problem formulation for multidisciplinary optimization." SIAM Journal on Optimization **4**(4): 754-776.

Dave, B. and L. Koskela (2009). "Collaborative knowledge management—A construction case study." Automation in construction **18**(7): 894-902.

Dávid, I., J. Denil, K. Gadeyne and H. Vangheluwe (2016). Engineering process transformation to manage (in) consistency. Proceedings of the 1st International Workshop on Collaborative Modelling in MDE (COMMitMDE 2016) co-located with ACM/IEEE 19th International Conference on Model Driven Engineering Languages and Systems (MoDELS 2016) St. Malo, France, October 4, 2016/Muccini, Henry [edit.]; et al.

Dick, J., E. Hull and K. Jackson (2017). Requirements engineering, Springer.

Drémont, N., N. Troussier, R. I. Whitfield and A. Duffy (2013). A meta-model for knowledge representation integrating maturity for decision making in engineering design. IFIP International Conference on Product Lifecycle Management, Springer.

El Hani, M., L. Rivest and R. Maranzana (2012). Product data reuse in product development: A practitioner's perspective. IFIP International Conference on Product Lifecycle Management, Springer.

Fortineau, V., T. Paviot and S. Lamouri (2013). "Improving the interoperability of industrial information systems with description logic-based models—the state of the art." Computers in Industry **64**(4): 363-375.

Friedenthal, S., R. Griego and M. Sampson (2007). INCOSE model based systems engineering (MBSE) initiative. INCOSE 2007 Symposium.

Friedenthal, S., A. Moore and R. Steiner (2014). A practical guide to SysML: the systems modeling language, Morgan Kaufmann.

Garetti, M., S. Terzi, N. Bertacci and M. Brianza (2005). "Organisational change and knowledge management in PLM implementation." International Journal of Product Lifecycle Management **1**(1): 43-51.

Gausemeier, J., T. Gaukstern and C. Tschirner (2013). "Systems engineering management based on a discipline-spanning system model." Procedia Computer Science **16**: 303-312.

George, E. and M. Pecht (2014). "Tin whisker analysis of an automotive engine control unit." Microelectronics Reliability **54**(1): 214-219.

Gielingh, W. (2008). "An assessment of the current state of product data technologies." Computer-Aided Design **40**(7): 750-759.

Grundstein, M. (2000). "From capitalizing on company knowledge to knowledge management." Knowledge management, classic and contemporary works **12**: 261-287.

Grundstein, M. and C. Rosenthal-Sabroux (2004). GAMETH®, a decision support approach to identify and locate potential crucial knowledge. Proceedings 5th European Conference on Knowledge Management.

Hammadi, M., J.-Y. Choley, O. Penas and A. Riviere (2012). Multidisciplinary approach for modelling and optimization of Road Electric Vehicles in conceptual design level. Electrical Systems for Aircraft, Railway and Ship Propulsion (ESARS), 2012, IEEE.

Hammadi, M., J.-Y. Choley, O. Penas and A. Rivière (2012). Mechatronic System Optimization based on Surrogate Models-Application to an Electric Vehicle. Simultech.

Hehenberger, P. and K. Zeman (2007). Design activities in the development process of mechatronic systems. Advanced intelligent mechatronics, 2007 IEEE/ASME international conference on, IEEE.

Hiriyannaiah, S. and G. M. Mocko (2008). Information management capabilities of MDO frameworks. ASME 2008 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, American Society of Mechanical Engineers.

Huzar, Z., L. Kuzniarz, G. Reggio and J. L. Sourrouille (2004). Consistency problems in UML-based software development. International Conference on the Unified Modeling Language, Springer.

Ilvonen, I., J. J. Jussila and H. Kärkkäinen (2015). "Towards a business-driven process model for knowledge security risk management: Making sense of knowledge risks." International Journal of Knowledge Management (IJKM) **11**(4): 1-18.

Isermann, R. and N. Müller (2003). "Design of computer controlled combustion engines." Mechatronics **13**(10): 1067-1089.

Jackson, C. K. (2006). "The Mechatronics System Design Benchmark Report–Coordinating Engineering Disciplines." Boston: The Aberdeen Group.

Jun, H.-B., J.-H. Shin, D. Kiritsis and P. Xirouchakis (2007). "System architecture for closed-loop PLM." International Journal of Computer Integrated Manufacturing **20**(7): 684-698.

Karayel, D., S. S. Özkan and R. Keleş (2004). "General framework for distributed knowledge management in mechatronic systems." Journal of intelligent manufacturing **15**(4): 511-515.

Karnopp, D. C., D. L. Margolis and R. C. Rosenberg (2012). System dynamics: modeling, simulation, and control of mechatronic systems, John Wiley & Sons.

Kaslow, D., G. Soremekun, H. Kim and S. Spangelo (2014). Integrated model-based systems engineering (MBSE) applied to the Simulation of a CubeSat mission. Proceedings of IEEE Aerospace Conference.

Kellner, A., P. Hehenberger, L. Weingartner and M. Friedl (2015). Design and use of system models in mechatronic system design. Systems Engineering (ISSE), 2015 IEEE International Symposium on, IEEE.

Kim, H., D. Fried and P. Menegay (2012). Connecting SysML models with engineering analyses to support multidisciplinary system development. 12th AIAA Aviation Technology, Integration, and Operations (ATIO) Conference and 14th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference.

Klein, R. (2000). Knowledge modeling in design—the MOKA framework. Artificial Intelligence in Design'00, Springer**:** 77-102.

Kleiner, S., R. Anderl and R. Gräb (2003). "A collaborative design system for product data integration." Journal of engineering design **14**(4): 421-428.

Krause, F.-L., H. Jansen, C. Kind and U. Rothenburg (2007). Virtual product development as an engine for innovation. The Future of Product Development, Springer**:** 703-713.

Kumar, J. S., V. Ganesan, J. Mallikarjuna and S. Govindarajan (2013). "Design and optimization of a throttle body assembly by CFD analysis."

Kyura, N. and H. Oho (1996). "Mechatronics-an industrial perspective." IEEE/ASME transactions on mechatronics **1**(1): 10-15.

La Rocca, G. (2012). "Knowledge based engineering: Between AI and CAD. Review of a language based technology to support engineering design." Advanced engineering informatics **26**(2): 159-179.

Lambe, A. B. and J. R. Martins (2012). "Extensions to the design structure matrix for the description of multidisciplinary design, analysis, and optimization processes." Structural and Multidisciplinary Optimization **46**(2): 273-284.

Lefebvre, T., N. Bartoli, S. Dubreuil, M. Panzeri, R. Lombardi, R. D'Ippolito, P. Della Vecchia, F. Nicolosi and P. D. Ciampa (2017). Methodological enhancements in MDO process investigated in the AGILE European project. 18th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference.

Levy, M., C. Loebbecke and P. Powell (2003). "SMEs, co-opetition and knowledge sharing: the role of information systems." European Journal of Information Systems **12**(1): 3-17.

Lightsey, B. (2001). Systems engineering fundamentals, DEFENSE ACQUISITION UNIV FT BELVOIR VA.

Maier, R. and T. Hadrich (2006). Knowledge management systems. Encyclopedia of knowledge management, IGI Global**:** 442-450.

Maranzana, N., N. Gartiser and E. Caillaud (2008). "From concurrent engineering to collaborative learning of design." International Journal of Design and Innovation Research **4**(1): 39-51.

Martins, J. R. and A. B. Lambe (2013). "Multidisciplinary design optimization: a survey of architectures." AIAA journal **51**(9): 2049-2075.

Matta, N., G. Ducellier and C. Djaiz (2013). "Traceability and structuring of cooperative Knowledge in design using PLM." Knowledge Management Research & Practice **11**(1): 53-61.

Mcharek, M., M. Hammadi, J.-Y. Choley, T. Azib and C. Larouci (2016). Modeling and multi-objective optimization of an Electronic Throttle in open-loop. Mechatronics (MECATRONICS)/17th International Conference on Research and Education in Mechatronics (REM), 2016 11th France-Japan & 9th Europe-Asia Congress on, IEEE.

Mhenni, F., J.-Y. Choley, O. Penas, R. Plateaux and M. Hammadi (2014). "A SysML-based methodology for mechatronic systems architectural design." Advanced Engineering Informatics **28**(3): 218-231.

Monticolo, D., J. Badin, S. Gomes, E. Bonjour and D. Chamoret (2015). "A meta-model for knowledge configuration management to support collaborative engineering." Computers in Industry **66**: 11-20.

Navarro, A., J. Badin, M. Austruy and A. Sow (2018). Collaborative generation of configuration technical data for a product to be manufactured, Google Patents.

Navet, N. and F. Simonot-Lion (2008). Automotive embedded systems handbook, CRC press.

Nentwig, M. and P. Mercorelli (2008). Throttle valve control using an inverse local linear model tree based on a fuzzy neural network. Cybernetic Intelligent Systems, 2008. CIS 2008. 7th IEEE International Conference on, IEEE.

Niknam, M. and J. Ovtcharova (2013). Towards Higher Configuration Management Maturity. IFIP International Conference on Product Lifecycle Management, Springer.

Nonaka, I. (1991). "The knowledge-creating company Harvard business review November-December." Google Scholar.

Nonaka, l., H. Takeuchi and K. Umemoto (1996). "A theory of organizational knowledge creation." International Journal of Technology Management **11**(7-8): 833-845.

Nouidui, T., M. Wetter and W. Zuo (2014). "Functional mock-up unit for co-simulation import in EnergyPlus." Journal of Building Performance Simulation **7**(3): 192-202.

Owen, R. and I. Horváth (2002). Towards product-related knowledge asset warehousing in enterprises. Proceedings of the 4th international symposium on tools and methods of competitive engineering, TMCE, Citeseer.

Pate, D. J., J. Gray and B. J. German (2014). "A graph theoretic approach to problem formulation for multidisciplinary design analysis and optimization." Structural and Multidisciplinary Optimization **49**(5): 743-760.

Pawlak, A. (2010). Challenges in collaborative design in engineering networks. eChallenges 2010 Conference Proceedings, IIMC International Information Management Corporate.

Peluso, F. (2015). "Knowledge reuse and collaborative environments in industrial engineering."

Penciuc, D., A. Durupt, F. Belkadi, B. Eynard and H. Rowson (2014). "Towards a PLM interoperability for a collaborative design support system." Procedia CIRP **25**: 369-376.

Petty, M. D. (2009). "Verification and validation." Principles of modeling and simulation: A multidisciplinary approach: 121-149.

Pratt, M. J. (2001). "Introduction to ISO 10303—the STEP standard for product data exchange." Journal of Computing and Information Science in Engineering **1**(1): 102-103.

Price, M., S. Raghunathan and R. Curran (2006). "An integrated systems engineering approach to aircraft design." Progress in Aerospace Sciences **42**(4): 331-376.

Rasmus, D. (2002). "Collaboration, contents and communities, an update." Giga Information Group Inc.

Rocca, G. L. and M. J. L. Van Tooren (2009). "Knowledge-based engineering approach to support aircraft multidisciplinary design and optimization." Journal of aircraft **46**(6): 1875-1885.

Rossi, C., A. Tilli and A. Tonielli (2000). "Robust control of a throttle body for drive by wire operation of automotive engines." IEEE Transactions on control systems technology **8**(6): 993-1002.

Saad, I. and S. Chakhar (2009). "A decision support for identifying crucial knowledge requiring capitalizing operation." European Journal of operational research **195**(3): 889-904.

Shetty, D. and R. A. Kolk (2010). Mechatronics system design, SI version, Cengage Learning.

Shetty, D., L. Manzione and A. Ali (2012). "Survey of Mechatronic Techniques in Modern Machine Design." Journal of Robotics **2012**.

Simpson, T., V. Toropov, V. Balabanov and F. Viana (2008). Design and analysis of computer experiments in multidisciplinary design optimization: a review of how far we have come-or not. 12th AIAA/ISSMO multidisciplinary analysis and optimization conference.

Simpson, T. W. and J. R. Martins (2011). "Multidisciplinary design optimization for complex engineered systems: report from a national science foundation workshop." Journal of Mechanical Design **133**(10): 101002.

Sobieszczanski-Sobieski, J. (1995). Multidisciplinary design optimization: an emerging new engineering discipline. Advances in Structural Optimization, Springer**:** 483-496.

Srinivasan, V. (2011). "An integration framework for product lifecycle management." Computer-aided design **43**(5): 464-478.

Stokes, M. (2001). Managing engineering knowledge: MOKA: methodology for knowledge based engineering applications, American Society of Mechanical Engineers.

Tedford, N. P. and J. R. Martins (2010). "Benchmarking multidisciplinary design optimization algorithms." Optimization and Engineering **11**(1): 159-183.

Tomizuka, M. (2000). "Mechatronics: from the 20th to 21st Century." IFAC Proceedings Volumes **33**(26): 1-10.

Törngren, M., A. Qamar, M. Biehl, F. Loiret and J. El-Khoury (2014). "Integrating viewpoints in the development of mechatronic products." Mechatronics **24**(7): 745-762.

Torry-Smith, J. M., A. Qamar, S. Achiche, J. Wikander, N. H. Mortensen and C. During (2013). "Challenges in designing mechatronic systems." Journal of Mechanical Design **135**(1): 011005.

Toussaint, L., F. Demoly, N. Lebaal, S. Gomes and M. Moteurs (2010). "PLM-based approach for design verification and validation using manufacturing process knowledge." Journal of Systemics, Cybernetics and Informatics **8**(1): 1-7.

Tudorache, T. (2006). "Employing ontologies for an improved development process in collaborative engineering."

Ullman, D. G. (2010). The mechanical design process: Part 1, McGraw-Hill.

van der Aalst, W. M. (2012). A decade of business process management conferences: personal reflections on a developing discipline. International Conference on Business Process Management, Springer.

van Gent, I., P. D. Ciampa, B. Aigner, J. Jepsen, G. La Rocca and J. Schut (2017). Knowledge Architecture supporting collaborative MDO in the AGILE paradigm. 18th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference.

Van, T. N. (2006). System engineering for collaborative data management systems: Application to design/simulation loops, Ecole Centrale Paris.

Verhagen, W. J., P. Bermell-Garcia, R. E. van Dijk and R. Curran (2012). "A critical review of Knowledge-Based Engineering: An identification of research challenges." Advanced Engineering Informatics **26**(1): 5-15.

Wiesner, S., M. Freitag, I. Westphal and K.-D. Thoben (2015). "Interactions between service and product lifecycle management." Procedia CIRP **30**: 36-41.

Zheng, C., M. Bricogne, J. Le Duigou and B. Eynard (2014). "Survey on mechatronic engineering: A focus on design methods and product models." Advanced Engineering Informatics **28**(3): 241-257.

Zheng, C., J. Le Duigou, M. Bricogne and B. Eynard (2016). "Multidisciplinary interface model for design of mechatronic systems." Computers in Industry **76**: 24-37.