Università Politecnica delle Marche
Research Doctorate School in Industrial Engineering
Curriculum in Mechanical Engineering

Methods and tools to support the early cost estimation of modular products in engineer to order contexts

Ph.D. Dissertation of:
Andrea Savoretti

Advisor:
Prof. Michele Germani

Curriculum supervisor:
Prof. Ferruccio Mandorli

XVI edition - new series

Università Politecnica delle Marche
Scuola di Dottorato di Ricerca in Ingegneria Industriale
Curriculum in Ingegneria Meccanica

Metodi e strumenti a supporto della stima preliminare dei costi per prodotti modulari in un contesto engineer to order

Tesi di dottorato di:

Andrea Savoretti

Tutor accademico:

Prof. Michele Germani

Coordinatore del curriculum:

Prof. Ferruccio Mandorli

XVI ciclo - nuova serie

Università Politecnica delle Marche
Dipartimento di Ingegneria Industriale e Scienze Matematiche
Via Brecce Bianche — 60131 - Ancona, Italy

To my family

Acknowledgements

This work started thanks to the partnership between the academia and the industrial world. In my opinion, this cooperation is important for the growth of both the academia and the industrial world.

Firstly, I would like to thank my doctoral advisor Professor Michele Germani. In these three years, I could see his strong passion for his work.

A special thanks to Professor Roberto Raffaeli who followed me step by step in the development of this work and also thanks for the stimulating discussions and for the time he invested in me.

Thanks to Meloni Tecno-Handling SRL that co-financed my doctoral scholarship. During the last years they made me feel part of the company.

Thanks also to Silvia for always believing in and trusting on me and for her precious support.

Finally, thanks to my family, for being always with me.

Andrea Savoretti

Abstract

Companies are required to adapt to the different customer needs in order to face the ever-increasing competitiveness within a global market. In fact, the mass production paradigm has been replaced in the last decades by the paradigm of mass customization. In the latter context, the customer usually requests an offer to a company, which includes the product specifications. The company which is interested in the request must answer with a technical-economical offer containing the product technical characteristics and price.

Concepts such as modularity, product platform and configuration have been introduced in order to manage a big number of product variants while maintaining design and production standards within the company through the product partition into functional blocks. Furthermore, in order to support the user in the configuration of a new product, knowledge-based tools have been introduced, which include design and manufacturing rules and allow the product architecture and CAD model generation.

The offer stage is especially critical in an engineer-to-order context, where the design effort is considerable. In this case, cost estimation must follow a preliminary product design stage, which leads to a technical solution that meets the product requirements. Once the critical issues of the offer stage have been identified, the aim of the research has been the study and the development of knowledge-based tools and methods that could support this critical stage within an engineer to order context.

At first, the thesis includes a literature review on the current tools and methods for product representation, configuration and cost estimation. A method that consists of 3 main parts has been then developed: (1) knowledge formalization and product representation, (2) support towards the solution and (3) cost estimation. Each part includes some features that are considered to be original. It is presented an innovative product meta-model that includes blocks dependencies, DSM-based methods for solution search and cost estimation methods that extend the evaluation to the product life-cycle. The proposed method has been partially implemented in a preliminary software platform and has been extensively validated by a case study thanks to the collaboration of the company Meloni Tecno-Handling S.r.l..

Riassunto

Le imprese si sono dovute adattare alle diverse esigenze dei clienti in modo da poter fronteggiare la sempre crescente competitività del mercato globale. Il paradigma della produzione di massa è stato infatti sostituito nelle ultime decadi da quello della personalizzazione di massa. In quest'ultimo caso il cliente normalmente emette una richiesta di offerta ad un'impresa corredata di specifiche prodotto. L'azienda che è interessata deve rispondere con un'offerta tecnico-economica che include le caratteristiche tecniche ed il prezzo del prodotto.

I concetti di modularità, piattaforma prodotto e configurazione sono stati introdotti in modo da poter gestire un gran numero di varianti prodotto pur mantenendo uno standard di progettazione e produzione attraverso la suddivisione del prodotto in blocchi funzionali. Inoltre, al fine di supportare l'utente nella configurazione di un nuovo prodotto, sono stati introdotti strumenti basati sulla conoscenza aziendale che inglobano regole di progettazione, produzione e che permettono la generazione di architetture e modelli CAD.

La fase di offerta risulta particolarmente critica in un contesto engineer to order, dove l'impegno progettuale è consistente. In questo caso, la stima dei costi deve seguire una fase di progettazione preliminare del prodotto necessaria per giungere ad una soluzione tecnica che soddisfi le richieste del cliente. Dopo aver individuato le problematiche della fase di offerta, l'obiettivo della ricerca è stato lo studio e lo sviluppo di strumenti e metodi basati sulla conoscenza aziendale, che vadano a supporto di questa fase critica in un contesto engineer to order.

Inizialmente la tesi comprende un riesame della bibliografia esistente sui metodi e gli strumenti a supporto della rappresentazione, della configurazione e della stima dei costi di un prodotto. È stato quindi sviluppato un metodo che consiste di 3 parti principali: (1) formalizzazione della conoscenza e rappresentazione del prodotto, (2) supporto alla ricerca della soluzione e (3) stima dei costi. Ciascuna parte presenta delle peculiarità che si ritengono innovative. Viene infatti presentato un modello innovativo per la rappresentazione del prodotto costituito da blocchi e relazioni, un metodo basato sulla DSM che facilita la ricerca della soluzione e dei metodi per la stima dei costi che estendono la stima al ciclo vita del prodotto. Il metodo proposto è stato parzialmente implementato in una piattaforma software preliminare ed è stato validato in maniera esaustiva da un caso studio grazie alla collaborazione dell'azienda Meloni Tecno-Handling S.r.l..

Contents

Nomenclature

AI	Artificial Intelligence
API	Application Programming Interface
ASP	Answer Set Programming
ATO	Assemble To Order
BOM	Bill Of Material
CAD	Computer Aided Design
CAE	Computer Aided Engineering
CAPEX	CAPital EXpenditure
CBR	Case Base Reasoning
CDS	Computational Design Synthesis
CEF	Cost Estimation Formula
CSP	Constraint Satisfaction Problem
DB	DataBase
DFMA	Design For Manufacturing and Assembly
DL	Description Logics
DMtC	Direct Material total Cost
DOE	Design Of Experiment
DSM	Design Structure Matrix
DTC	Design To Cost
ERP	Enterprise Resource Planning
ETO	Engineer To Order
FBC	Feature Based Costing
FEM	Finite Element Method

HC	Hybrid Configuration
IT	Information Technologies
KBE	Knowledge Based Engineering
KBS	Knowledge Based Systems
KE	Knowledge Engineering
LCC	Life Cycle Cost
MOKA	Methodology and software tools Oriented to Knowledge based engineering Applications
MTO	Make To Order
OO	Object Oriented
OOAD	Object Oriented Analysis and Design
OOP	Object Oriented Programming
OPEX	OPerating EXpenditure
PDM	Product Data Management
PLC	Production Labor Costs
PLM	Product Life cycle Management
RFP	Request For Proposal
RSM	Response Surface Methodology
SAT	Boolean SATisfiability problem
SME	Small and Medium Enterprises
UML	Unified Modeling Language
VMfC	Variable part of Manufacturing Costs

List of Figures

List of Tables

Chapter 1.

Introduction

1.1. Context of the thesis

The ever-increasing competitiveness, due to the market globalization, has forced the industries to modify their design and production strategies. A key point is the development of products that fulfil the individual customer needs as close as possible. There is therefore a trend of moving from mass production to mass customization. In many cases, in order to fulfil the individual customer needs, a new technical solution that requires a substantial engineering effort needs to be designed, which is the case of the engineer to order (ETO) context.

The business model of many companies is based on the engineer to order (ETO) model and the customization of the products in the portfolio. In particular, ETO companies manufacture new products according to the customer technical requirements given in the request for proposal (RFP). For these companies, offer is a critical and complex stage, which precedes the purchase of the product (the contract). A simplified scheme of the offer stage is shown in Figure 1.

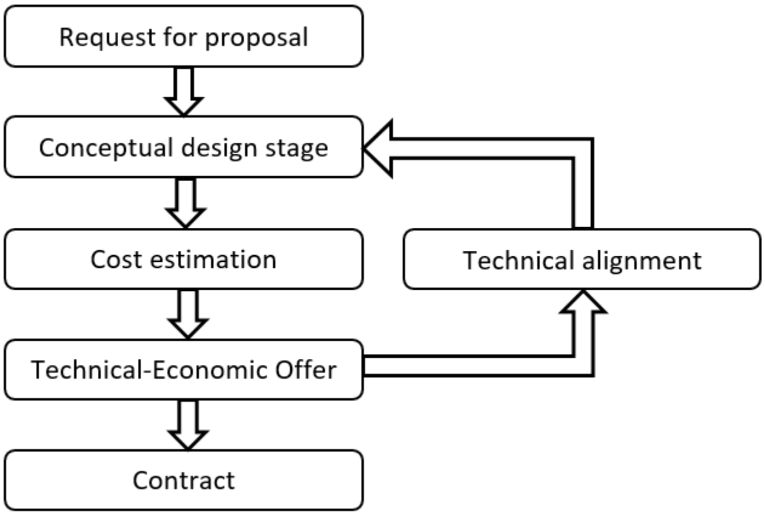

Figure 1 - Offer stage scheme

The RFP moves from the product technical requirements and starts a negotiation phase ending with the offer generation and eventually the contract. Usually, a simplified but complete design process is needed, however limited to the most significant choices and the main dimensioning activities. This early design stage is known as conceptual design. Product configuration containing a preliminary design solution that meets the requirements reported in the RFP is one of the output of this phase. It is clear that the RFP by a possible customer often leads to a significant number of alternatives.

The definition of the right price in an offer is a critical activity that involves expertise, product knowledge and the correct estimation of design and production efforts. Compiling technical proposals is a time-consuming activity and the strong competition on the market generally leads to poor success rates in the order acquisition. In addition, the customer intervenes heavily on the product throughout the offer stage, e.g. proposing changes or discussing the product costs and its main features. Therefore, it is mandatory to employ consistent approaches to rapidly formulate reliable offers as new requirements comes from a potential customer.

In this first stage, it is important to identify the requirements and the constraints within the RFP, in order to prepare a reliable offer. It is well known that in the early design phase some activities require a priori choices, which lead to different alternatives depending on the chosen decision path. Moreover, the compliance of several constraints, e.g. weight and overall dimensions, will be known only after some activities of the design process have been performed, leading to a rough configuration process and thus an inaccurate cost estimation. In fact, the cost estimation that follows the conceptual design stage is mostly done in an empirical and unstructured way.

Once the conceptual design and the cost estimation stages are completed, the technical offer, comprising the main characteristics of the product and, in many cases, a drawing, can be sent to the possible customer together with the economic offer, which includes the product price. In general, the customer evaluates the technical-economic offer, often comparing it to different offers of other possible suppliers. From this, technical alignment stages follow, in which many product characteristics and requirements are clarified and more precisely defined. Technical alignment stages require much time and effort, because product design and costs shall be revised considering any possible change in product requirements. The company should answer to the new requests with new offers, until the customer accepts the offer, which becomes a contract.

The awareness of these problems has led, over the last 20 years, to the definition of some methodologies to assess cost before product realization in order to optimize product and process design, such as design for manufacturing and assembly (DFMA) approach (Boothroyd, Dewhurst, & Knight, 1994) to feature based costing (FBC) (L.S. Wierda, 1991), to design to cost (DTC)

(Ehrlenspiel, Kiewert, & Lindemann, 2007). However, their application in industry is poor due to complex data analysis and knowledge structuring as well as numerous resources to be involved in the project. On the other hand, different approaches like knowledge based engineering (KBE) (McMahon, Lowe, & Culley, 2004), artificial intelligence (AI) algorithms (Negnevitsky, 2005), object-oriented (OO) design and functional programming (La Rocca, 2012) have been conceived to easily capture, structure and reuse the design knowledge in a fast and easy way, by automating repetitive tasks and optimization stages.

Although there is the possibility of using such tools, it has been observed that many companies just base the process on poor empirical models working by analogy on the basis of the expertise of senior designers and searching for similar past solutions. Then, product BOM is adapted and costs are updated accordingly. Such an approach mostly leads to some inaccuracies in the cost estimation, especially for ETO products. In fact, it has been seen that these products need a preliminary design phase, i.e. the conceptual design, before evaluating the product cost.

According to Pahl et al. it is useful to divide the planning and design process into four main phases: (1) planning and task clarification, (2) conceptual design, (3) embodiment design and (4) detail design pointing out that it is not always possible to draw a clear borderline between these main phases, neither it is possible to avoid backtracking (Pahl, Beitz, Feldhusen, & Grote, 2007). Thus, engineering design is a highly iterative process. Phase (1) leads to the product requirements list, which the later stages are based on. While the aim of the conceptual design stage is to define the principle solution (concept), the embodiment design must lead to the specification of the layout. Indeed, in many cases a layout assessment has been needed since the conceptual design stage to make sure the solution meets the product requirements. Moreover, the awareness that the decisions made in the conceptual design phase have a huge impact on the life-cycle costs (LCC) of the product, has resulted in the need of a shift of the knowledge typically required in the later stages to the earlier stages (Chandrasegaran, Ramani, Sriram, Horváth, & Bernard, 2013). From this follows the importance of the design process knowledge formalization to be reused at the conceptual design stage.

Conceptual design also requires cost estimation to evaluate variants against economic criteria. During the offer stage, customer requirements may change due to a negotiation phase in which costs and technical requirements are involved. In these cases, a rapid evaluation of the variants and their costs is needed.

Cost assessment needs a knowledge formalization not limited to the design process, indeed it must include the product life-cycle from inception, through engineering design and manufacture, to service and disposal.

The motivation of the work stems from the need to face the following problems of cost estimation:

(1) The resulting cost is strongly affected by the subjectivity of the cost estimator;
(2) The elaboration of the technical proposal and the commercial offer is time consuming;
(3) Expertise in a wide area is required;
(4) Difficulties in retrieving information of past cases;
(5) Difficulty of connecting information, e.g. customer specifications and BOM;
(6) Difficulty of making the most economic choices during the product configuration;
(7) Difficulty of taking into account the product life cycle within the proposal.

The above problems lead to the need of a method and a tool of product configuration and early cost estimation. The tool should manage the company knowledge on the product life cycle, particularly the product design and manufacturing knowledge. Knowledge should be elicited and formalized, so that it can allow the past cases retrieval and the connection between customer specifications and the product configuration. Finally, the tool should allow a rapid definition of a new product configuration and its costs.

KBE tools are helpful to automate the design process avoiding the subjectivity of the cost estimator (1) and allow saving time for offer generation (2). KBE systems use stored knowledge for solving problems in a specific domain, making use of inference mechanism to derive an answer to the problem.

In order to develop a KBE application, a previous step of knowledge formalization is required. It is important to collect both tacit and formal knowledge. Formal knowledge is embedded from product documents, drawings and engineering dimensioning algorithms, while tacit knowledge, which is made of implicit rules, comes from the experience of people with technical expertise. Knowledge formalization is a critical issue which involves capturing, representing and reusing knowledge of various forms.

Knowledge representation can be classified into five categories: pictorial, linguistic, virtual, algorithmic and symbolic (Owen & Horvát, 2002). Chandrasegaran et al. show the various forms of knowledge involved at each stage of the design process and state that in the early design stages, knowledge representation is predominantly linguistic and pictorial (Chandrasegaran, Ramani, Sriram, Horváth, & Bernard, 2013). This is also why there is a lack of tools able to support the conceptual design stage. Moreover, the knowledge about design requirements and constraints available during conceptual design phase is often imprecise, approximate or incomplete (Wang , Shen, Xie , Neelamkavil , & Pardasani , 2002).

Research focuses in developing standard languages and ontologies to support the capturing, representing and modelling the engineering design knowledge. Issues (3), (4) and (5) address to the knowledge formalization. Issue (3) concerns the translation of the human expertise that is tacit knowledge into formal knowledge which can be reused. Past projects data should be stored and organized in a repository, including connections and dependencies (5) to be easily retrieved when needed. In fact, the goal is to reuse the knowledge originated from the latter stages, in order to provide information for the early stages, in particular for the product concept design.

A way to reuse the product knowledge consists in building a product platform. Product platform design has been widely studied in the last decades, because of the importance for companies to offer a large variety of products. Product platforms allows to manage product families, which are the key players to achieve the paradigm of mass customization. Product family has been defined as the process of capturing and modeling multiple product variants within a single data model (Mannisto, Peltonen, & Sulonen, 1996).

Sabin and Weigel (1998) assert that given a set of customer requirements and a product family description, the task of configuration is to find a valid and completely specified product structure among the alternatives the generic structure describes.

Configurators are the most common tools to support product configuration, mostly a software that guides the users through the configuration process. Indeed, there are different types of product configurators depending on their different goals and on the way the user can interact with the software. Configurators are mostly thought to be used by customers for selecting the product that matches his needs, therefore for a commercial purpose. In many cases, configurators allow to be aware of the costs of every choice made.

Felferning et al. distinguish several types of product configurators basing on their different goals and on the level of configuration freedom (Felferning, Hotz, Bagley, & Tiihonen, 2014). Engineer to order (ETO) configurators have a high level of complexity since the knowledge domain of ETO products is vast and ever-changing. In addition, the product configuration requires in these cases an engineering effort so that it is impossible to make the task automatic.

In order to make the most economic choices (issue 6), cost estimation should be made step by step during the configuration process, being aware of the economic impact of every design choice. Moreover, design choices should not be guided only by economic criteria, in fact several parameters should be considered during product configuration, since the conceptual design stage. Optimization criteria should also extend the cost assessment by considering the entire product life-cycle (issue 7) (Norris, 2001).

1.2. Research goals

In the context described above, research efforts have been concentrated in developing a methodology and a related framework to support early cost estimation, particularly suitable for the companies during the offer stage.

The methodology and software focus but it is not limited on mechanically based products that include also non-mechanical systems, i.e. electrical, hydraulic, electronic, software and automation. Moreover, the research is focused on engineer to order (ETO) products, which need an important effort in designing before producing. Unlike make to order (MTO) or assemble to order (ATO) products, ETO products require a considerable effort during the configuration stage, which must precede the cost estimation.

This work started with the understanding of the effort and time needed during the offer stage for a company in the context above and with the awareness that the effort and time are often unremunerated because of the poor success rate in the order acquisition. Consequently, the main focus of the research is on reducing time and effort in the offer stage, which allows a company to properly manage a greater number of offers in less time. The awareness of these issues, which comes from the industrial world, is well documented in the literature. Section 3.1 discusses the most important issues that stemmed from an interview with several companies. The objectives of this work derived mostly from the results of this interview that reports the most important company needs regarding the offer stage and the cost estimation.

The methodologies and tool discussed in this thesis are particularly suitable for products having a good degree of modularity. Companies need to produce different products as per technical requirements, but they rarely start from new ideas when designing. Usually, most of the parts of the products (modules) are reused according to the required product functions. For this reason, some research goals are to support the functional analysis of the product together with the past projects reuse.

The framework is aimed at building a company knowledge repository, which can then be used to design new products and estimate their costs and at supporting the user in finding solutions for new products having different requirements. In particular, the support towards the solution aims to simplify the management of complex design problems characterized by a great number of variables to be assigned and by several design choices to be done. The support focuses on dividing the complex problem into more sub-problem more manageable and on identifying the design choices to be taken at the early stages (a priori choices).

As regards the cost estimation, the aim is to extend the evaluation to the entire product life-cycle, in order to obtain the life-cycle cost of the product. In fact, the objective has been to go beyond the manufacturing costs, developing cost models that include product design, maintenance and running costs such as power consumption and consumables.

1.3. Research approach

In order to achieve the above goals, the research approach that has been followed in these years consists in the followings points reported in the scheme of Figure 2.

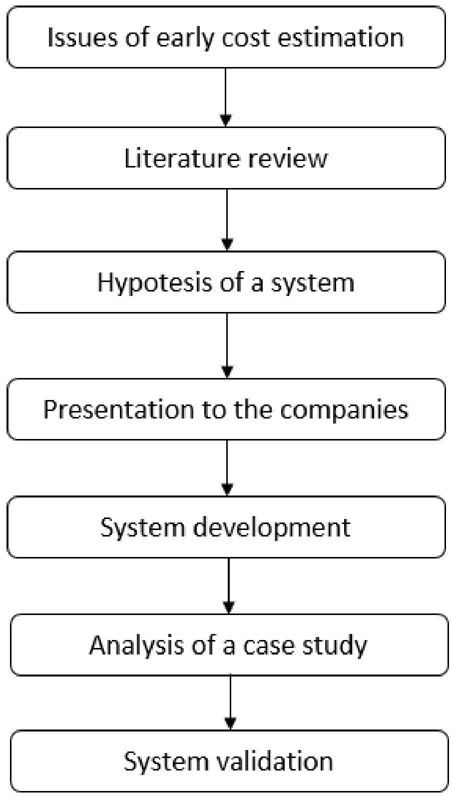

Figure 2 - Research approach scheme

The work began with the identification of the main issues regarding the offer stage and the cost estimation at early stages. The time spent in the company that contributed to the doctoral studies gave the opportunity to be inside the problem and be aware of the most important issues of offer phase and cost estimation. These issues derived mostly from people having a multi-year experience within the company. Moreover, the investigation of company needs regarding this topic, which is documented in section 3.1, confirmed and expanded the already known issues. A broad literature review of the topics reported in the state of the art section allowed me to know the possibility of the current approaches and to understand their shortcomings. Several methodologies have been found with the purpose of achieving the research goals above. However, most of these approaches, which have been broadly discussed in the state of the art section, require a good knowledge in IT and programming. The aim of building a framework easily usable in the companies from any kind of people has led to the choice of simple methods and tools to face the problems discussed above.

From these, a hypothesis of a tool for early cost estimation has been thought, then it has been presented to different companies in order to check the consistency of the approach. The results of the company interview and of the presentation of a framework demo have allowed to decide what methods and tools were the most appropriate to be included in the system, basing on the feedback of the possible future users of the framework. Therefore, the system has been developed taking into account the company needs and suggestions.

Finally, the framework has been validated in a case of study within the sponsoring company proving a good usability and giving reliable results in a real industrial situation, thus demonstrating the goodness of the methodology that has been chosen.

1.4. Thesis overview

This thesis consists of three main parts. The first part, which corresponds to chapter 1, is the theoretical foundation of the research area. Literature about product platform, modularity, design structure matrix (DSM), knowledge based engineering (KBE), configuration, computational design synthesis (CDS) and early cost estimation has been reviewed.

After an introduction that reports the results of an interview for discovering the needs and expectations of companies about the early cost estimation topic, chapter 2 focalizes on the methodologies and the tools used for achieving the research goals above discussed. In particular, the approach comprises (1) the knowledge formalization and representation, (2) the support towards a solution and (3) the cost estimation.

The definition of an approach and the correlated framework is the result of a broad literature review and the direct experience in the company that has contributed to the doctoral scholarship.

In chapter 3, the methodologies and the tools that have been discussed in chapter 2, are applied to the product overhead crane, in order to prove the validity of the proposed approach.

1.5. Author contributions

This thesis includes methods and tools for supporting product configuration and early cost estimation, especially of modularity products in the ETO context. The following features are believed to be original:

- Elaboration of an approach to formalize the knowledge of a company
- Elaboration of a meta-model to conveniently represent modular product structure eliciting knowledge and supporting product modifications by means of the reuse of the past projects
- Introduction of new typologies of relationships in the meta-model
- Elaboration of a DSM-based method to support the user in finding the solution
- Elaboration of an early cost estimation model that considers the product life-cycle

- Developing of a framework that is possible to be integrated to the most common company tools

Chapter 2.

Background and related work

This chapter provides a background for the tools and methods used in this thesis and gives a literature review regarding similar approaches found in literature.

At first, modularity and product platform, design structure matrix and knowledge based engineering constitute the theoretically background for the company knowledge formalization and representation described in the section of the method. The section of the design structure matrix describes also some tools that have been used to support the design solution search.

Configuration and computational design synthesis have been presented because they are the most common approaches for automating product design. A section is dedicated to the current methods for early cost estimation and contains the basis for the proposed method.

Finally, a section presents the limits of the current approaches that have been described in this chapter.

2.1. Modularity and product platform

This section reviews the concepts of modularity and product platform, which play a key role in mass customization strategies.

Ulrich and Eppinger (2016) state that the functional elements of a product are the individual operations and transformations that contribute to the overall performance of the product. Instead, the physical elements of a product are the parts, components, and subassemblies that ultimately implement the product functions.

The architecture of a product is the scheme by which the functional elements of the product are arranged into physical blocks and by which the blocks interact. Mostly, products belong to a family of products, i.e. product variants based on a common platform. Product family can be defined as a group of related products that share common features, components, subsystems and yet satisfy a variety of market niches (Farrel & Simpson, 2003).

Product platforms allow to manage product families using shared assets to enable cost-effectiveness. Otto et al. (2016) review the main activities for product platform design and examine a set of product platform development processes used at several different companies. Modularity is a crucial part of platform thinking.

Modules are defined as physical structures that have a one-to-one correspondence with functional structures (Ulrich & Tung, 1991). Hölttä-Otto specifies that a module is an independent building block of a larger system with a specific function and well-defined interfaces. In addition, she states that a module has fairly loose connections to the rest of the system allowing an independent development, outsourcing, manufacturing, recycling, etc. of the module as long as the interconnections at the interfaces are carefully considered (Hölttä-Otto, 2005).

Modular products have been defined as machines, assemblies or components that fulfil various overall functions through the combination of distinct function units (building blocks) or modules (Pahl, Beitz, Feldhusen, & Grote, 2007). Each module provides one or more well-defined functions of the product and includes several variants that deliver different performance level for the function(s) the product is intended to serve. For example, any hook of a crane has the function to handle a load. Although the hook function is always the same, several variants of hooks can be identified, depending on the magnitude of the load or on the number of attachment points.

In developing a platform architecture, two alternative approaches can be used: a function-based approach or a component-based approach. While the function-based approach takes a more general view to define common functions independent of any particular embodiment decision, the component-based approach makes use of a priori knowledge of the most relevant components needed (Otto, et al., 2016).

Höltta & Salonen (2003) identified three consolidated and recognized systematic modularity methods: (1) function structure heuristic methods (Stone, Wood, & Crawford, 2000), (2) clustering of Design Structure Matrix (DSM) (Pimmler & Eppinger, 1994) and (3) the Modular Function Deployement (MFD) method (Ericsson & Erixon, 1999).

The Stone et al.'s method grounds on the functional model of the product, that is defined following the well-known guidelines of Pahl et al. (Pahl, Beitz, Feldhusen, & Grote, 2007). A function structure is a functional decomposition, represented through a block diagram, of all the functions of the product and of the flows, of material, energy, and information, that are established between them.

Functions denote the intended input/output relationships of a system, or of a product, whose purpose is to perform a task. They are expressed by pairs of verb-noun, connected by flows of energy, material and signal. The overall function of the product is split into sub functions of lower complexity, which have to be connected through the flows of energy, material and signals to produce a functional structure as shown in Figure 3.

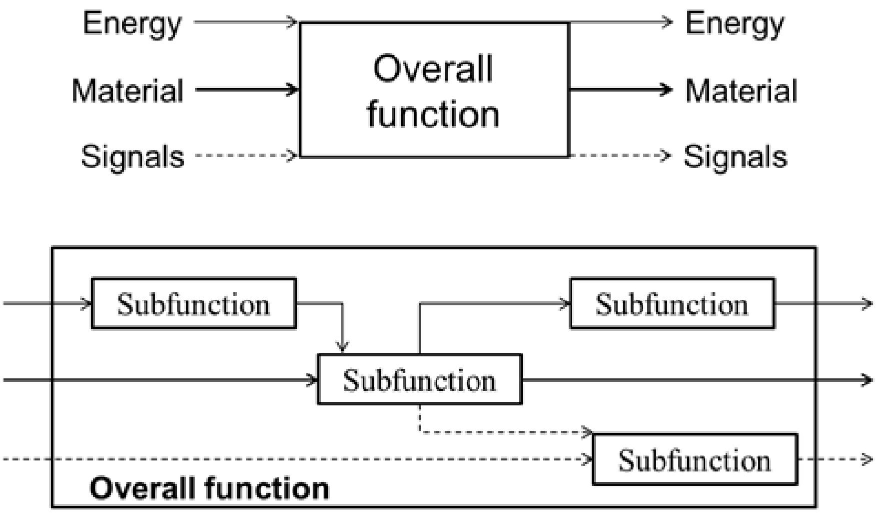

Figure 3 - An example scheme of a functional decomposition

Function-based representations are mainly used during the conceptual design stage and several modeling approaches can be found in literature. Summers and Rosen (2013) discuss three different modeling approaches with a focus on conceptual design and make a comparison of the types of information supported by these representations. The above cited function structure heuristic methods (Stone, Wood, & Crawford, 2000) consist in three heuristics (see Figure 4) that identify groups of functions so that each group constitutes a module.

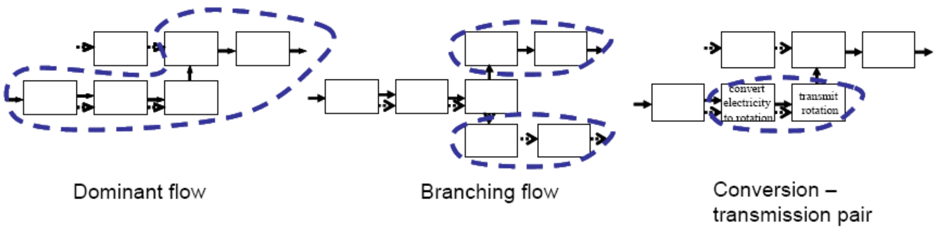

Figure 4 - Function structure heuristics

MFD (Ericsson & Erixon, 1999) is also based on functional decomposition but in this method modularity drivers other than functionality are considered. Modularity drivers are then mapped against functions (see Figure 5).

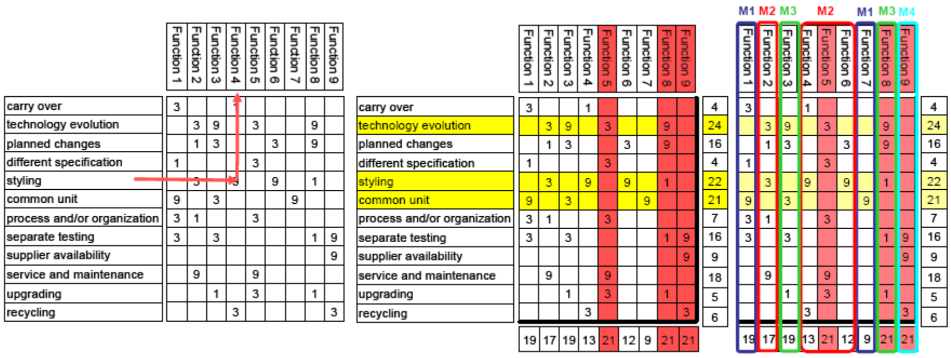

Figure 5 - Main steps of Modular Function Deployment

DSM is born to manage complex system (Ulrich & Eppinger, 2012). It can be used to provide a compact representation of the product architecture in terms of relationships between its constituent components. Clustering algorithms (section 2.2) allow to identify modules within a DSM, i.e. group of components, based on their interactions.

While the function structure heuristic methods require function structures and are suitable for the function-based approach, DSM method is applicable without function structures using the component-based approach.

2.2. Design structure matrix

A design structure matrix (DSM) provides a simple, compact, and visual representation of a complex decisional system that supports innovative solutions to decomposition and integration problem (Browning, 2001). The DSM is able to model and analyze dependencies of one single type within one single domain.

DSM is a square matrix NxN used to represent the dependencies between N elements of a system.

DSMs can be divided into (a) static and (b) time-based. Browning identifies four applications for DSMs:

1. Component-Based or Architecture DSM
2. Team-Based or Organizational DSM
3. Activity-Based or Schedule DSM
4. Parameter-Based DSM

While (a) static DSMs can be used to organize (1) product architecture or (2) teams of people, (b) time-based DSMs bring practical benefits to sequence (3) activities or (4) parameters.

In section 2.1 the component-based DSM for modeling system component relationships and for identifying modules in product architecture have been discussed.

The team-based DSM is quite similar to the previous, but it is used to analyze inter-team interfaces, which provide the greatest leverage for improving the organizations.

Static DSMs are used for representing the interactions between elements, with the aim to identify several blocks of elements in which the interactions are maximized. Binary DSMs allow to represent only the existence of a relation, whereas numerical DSMs use a numerical value to show the strength of a relation.

Once the components and their interactions are placed in the DSM, a clustering algorithm can be applied to group the components so that the interactions within clusters are maximized while among the clusters are minimized. The example given in Figure 6 shows the original DSM and two alternative clustered DSM. Various clustering algorithms have been created with the goal of module definition in mind, e.g. Thebeau developed a clustering algorithm based on simulated annealing (Thebeau, 2001).

Figure 6 - DSM example: original form and two different clustering types

On the contrary, time-based DSMs are used to show dependencies between the elements of a system, i.e. activities or parameters. Time-based DSMs are generally binary where the presence of a 1 (or a X) in ij item means that the activity (or the parameter) i depends on the activity (or the parameter) j. If a 1 (or a X) is both in ij and ji items, the activities (or the parameters) are interdependent. The convention used in this thesis to represent dependencies is, according to Eppinger et al. (1991), the IC/FBD convention, i.e. DSM feedback marks with inputs shown in columns and outputs in rows.

Activity-based DSM is used to sequence a set of activities according to their dependencies and to identify blocks of activities which are affected by mutual dependencies. These activities must be solved in an iterative manner in order to proceed with the following activities.

Parameter-based DSM is actually a low-level of the activity-based DSM. In fact, while an activity-based DSM can be used to model a design process, a parameter-based DSM can be used to represent the relationships between the parameters of the design process.

In order to sequence an activity (or parameter)-based DSM, several partition algorithms can be found in literature, e.g. the reachability matrix method calls for finding a multi-level hierarchical decomposition for the matrix (Warfield, 1973). By means of partition tools, it is possible to reorder the DSM and eventually find the coupled blocks of parameters that are affected by mutual dependencies (see Figure 7).

	A	B	C	D	E	F	G
A			1	1			
B							1
C	1	1				1	1
D		1					
E			1			1	
F							
G				1		1	

(a) Original DSM

	F	G	D	B	A	C	E
F							
G	1		1				
D				1			
B		1					
A			1			1	
C	1	1		1	1		
E	1					1	

(b) Partitioned DSM

Figure 7 – Example of a DSM before (a) and after (b) partitioning

Aggregation, decomposition and tearing are the possible alternatives to solve the coupled blocks (Browning, 2001). While aggregation consists in collapsing two or more activities in order to remove interdependencies, decomposition attempts to explode coupled activities in lower-level activities which are not mutually dependent. Finally, tearing consists in removing dependencies and making assumptions in order to exit the loop of coupled activities (or parameters) and proceed with the process.

2.3. Knowledge based engineering

Knowledge based engineering (KBE) is a technical domain that comprehends methodologies and tools to acquire, formalize and represent in IT system the knowledge of a specific application field. KBE is a special type of knowledge based systems (KBS) with a particular focus on product engineering design and downstream activities such as analysis, manufacturing, production planning cost estimation and even sales.

KBE stands at the cross point of several disciplines, such as artificial intelligence (AI), computer aided design (CAD) and object-oriented programming (OOP). The historical roots of KBE come from expert systems of the 1960s.

An expert system (or KBS) has been defined as a tool that stores and accumulates specific knowledge of different areas and generates solution in a user interface to given problems (Steinbichler, 2008). KBE systems are a subset of KBS where methodologies and technologies are studied to capture and re-use knowledge of the product and of the design process to reduce production time and costs, which is primarily achieved through automating repetitive design tasks (Verhagen, Bermell-Garcia, Van Dijk, & Curran, 2012).

A KBE system is a general-purpose tool which does not contain any knowledge about any specific domain. Developing a KBE application is mostly about writing code using a KBE programming language. State-of-the-art KBE systems provide the user with an object-oriented language, which allows modeling the domain knowledge as coherent network of classes (La Rocca, 2012).

Object-oriented analysis and design (OOAD) is a software engineering approach that models a system as a group of interacting objects (Coad &

Yourdon, 1991). Each object represents some entity of interest in the system being modeled and is characterized by its class, its state (data elements), and its behavior. Various models can be created to show the static structure, dynamic behavior and run-time deployment of these collaborating objects. There are many different notations for representing these models, such as the unified modeling language (UML).

The language of KBE systems is characterized by a declarative style, i.e. the order in which object are declared is not relevant. Most of KBE languages are based on object oriented dialects of Lisp programming language (La Rocca, 2012) which was used in artificial intelligence research and implementation. The define-object operator is the basic means to apply the object-oriented paradigm in KBE application, allowing to define classes, sub-classes, objects and relationships of inheritance, aggregation and association.

One of the most important features of KBE systems is the integration with CAD environment, which allows to generate and manipulate geometry by means of the programming language. This reveals the close link between KBE and the fields of design automation and design optimization.

The focus of engineering design automation is on automating repetitive or routine design tasks, e.g. CAD models and dimensional drawings generation. For instance, Frank et al. (2014) have presented a framework of engineering design automation for creating complex CAD models, which has been applied to some parts of cranes. Instead, the aim of design optimization is to find out a valid design solution, within a solution space, which minimizes or maximizes a given objective function (e.g. the weight of a steel structure).

Different contributions to design optimization problems have been provided, mostly regarding structural optimization. Cicconi et al. (2016) develop a platform in which an optimization tool and a FEM analysis software have been integrated in order to support the automatic optimization of a steel structure. Solution exploration has been done by means of a design of experiment (DOE) plan. In fact, DOE theory is a traditional way to explore possible solutions, which makes use of a multi-variation approach (Montgomery, 2008).

Investigators define the number of levels for each variable and vary them simultaneously during the tests. Maximum information is thus obtained by means of a full factorial plan, in which all the possible level combinations are performed. Thanks to a regression model it is possible to obtain the system response at different operating conditions. Since full factorial plans require a number of tests that increases exponentially with the number of parameters and levels and that easily outgrows the resources of most investigations, fractional factorial plans are adopted in order to reduce the number of tests.

In order to support the development of KBE applications, knowledge must be converted in a structured representation and be formalized. Thus, knowledge acquisition and codification, which are actually the field of the knowledge engineering (KE), play a fundamental role in developing a KBE application.

Several methodologies for knowledge capture and formalization are available in literature for KBE. One of the most well-known KBE methodologies is the methodology and software tools oriented to knowledge based engineering applications (MOKA). This methodology, based on eight KBE life-cycle steps and expressed in accompanying case-specific informal and formal models, is designed to take a project from inception towards industrialization and actual use (Stokes, 2001).

Verhagen et al. (2012) made a review of KBE and identified the current shortcomings and the research challenges of this topic. In particular, the improvement of the methodological support for KBE has been identified as the most important research challenge.

2.4. Product configuration

Sabin and Weigel (1998) define configuration as a special case of design activity with two key features: the artifact being configured is assembled from instances of a fixed set of well-defined component types and the components interact with each other in predefined ways. Component types represent sets of alternative components, namely instances, and they are characterized by properties. Constraints restrict the way in which components can be combined each other and thus the solution space. Configuration task must provide the list of the selected component (BOM) as well as the product structure and topology (product architecture).

A configuration task involves two different steps: (1) representing the problem and (2) finding a solution for the problem posed.

Representing the problem (1) refers to the description of the domain knowledge, which is the most critical issue. The knowledge acquisition process leads to the definition of the configuration model, which describes the component types within the domain and the relations among these.

Each component (instance) is uniquely identified by its set of properties, describing its function and performance. Properties can be classified into attributes, resources and ports. Attributes are used to describe the features of a component, i.e. geometry, weight, functions. A resource is something provided by some components and consumed by others. Resources are used in configuration models to check for balance issues. Ports describe the way in which a component can interact with other components of the domain. Ports are used in a component to impose restrictions on the type and number of components that can be connected to it.

Configuration has become an important application of Artificial Intelligence (AI) techniques with the introduction of the XCON system (McDermott, 1982). Although XCON was actually an expert system (see section 2.3), it allowed to solve configuration problems. XCON was a rule-based configurator, i.e. production rules (if-then structures) were used to express configuration knowledge.

The main problem of rule-based configurators is the interdependence between domain and problem knowledge, which causes enormous efforts in knowledge base development and maintenance (Soloway, Bachant, & Jensen, 1987). The problems encountered with the use of rule-based configurators have resulted in the development of model-based configurators, in which the domain knowledge is completely separate from the problem-solving knowledge.

Model-based knowledge can be represented by means of a constraint satisfaction problem (CSP). While if-then rules presuppose a direction from the condition to the action part, constraint-based representation allows multidirectionality, which can be considered the main difference between rule-based and constraint-based problem solving.

A constraint satisfaction problem (CSP) can be defined on a finite set of variables whose values belong to the definition of finite domains and a set of constraints. A constraint on a set of variables is a restriction on the values that those variables can take simultaneously. Conceptually, a constraint is seen as a set of all the values that the variables can take on at the same time and it can be represented by means of matrices, equations, inequalities or relations.

In a similar manner, engineering design is the process of satisfying requirements by developing and synthesizing building blocks into meaningful designs that meet the requirements to fulfill needs and desires, but always respecting design constraints, which are fundamental in determining the resulting solutions (Chakrabarti & Bligh, 1996). Numerical constraints and restrictions on the possible values of a design parameter have been used in parametric design (Mullineux, Hicks, & Medland, 2005) or, alternatively, constraints may represent high-level qualitative information being part of the problem definition.

Design process can be considered as a CSP under a set of requirements, assumptions and design limits (Münzer, 2014). A design is consistent if it does not violate any of the constraints and is complete if it includes all variables. CSP on finite domains are typically solved using a form of search.

The approach for the resolution is based on ad-hoc algorithms that compare all found solutions at the same depth (with usually the same cost) to avoid running in infinite paths. The most used techniques are variants of backtracking, constraint propagation, and local search. The branching factor at each level of the tree is equal to the number of variables not yet assigned, multiplied by the number of values that each variable can take.

Several CSP languages and solvers have been introduced to reduce the branching factor. Gecode is a toolkit for the development of constraint-based systems and applications; MiniZinc is a modeling language to mid-level constraints, which exploits the advantages of a library of pre-defined constraints; Cream is a Class Library for Constraint Programming; Jacop is a Java-based constraint solver; Choco is a free Java library dedicated to Constraint Programming.

Optimization tools can work in adjunction to CSP solver and allow finding out a valid solution that minimizes or maximizes an objective function.

In order to deal with complex configuration tasks, generative configuration approaches are needed. While static (but also dynamic) CSPs are able to deal only with a finite number of components and constraints, generative CSPs are not limited to a predefined set of components or constraints, by moving components and constraints to the level of component types and meta-constraints (generic constraints) respectively.

Besides the constraint-based knowledge representation, graphical and logic-based knowledge representation have been developed. Both feature models and UML configuration models fall into graphical knowledge representation. While feature models provide a description of the product mainly from the perspective of its requirements, UML configuration models use the OO paradigm (see section 2.3) to provide a detailed description of the product structure. UML configuration models can be translated into a corresponding logic-based representation (Felferning, 2007).

Answer set programming (ASP), description logics (DL) and hybrid configuration (HC) are different approaches to perform a logic-based knowledge representation. In particular, ASP maps a logic-based configuration model onto propositional logic theories, which are solvable by means of boolean satisfiability (SAT) solvers. Since ASP allows a component-oriented representation by using a first order logic-based knowledge representation, it is a valid alternative to CSP.

Today, most of the configuration environments are component-based, e.g. SAP, Siemens, ConfigIT (Moeller, Andersen, & Hulgaard, 2001), EngCon (Felferning, Hotz, Bagley, & Tiihonen, 2014), Tacton (Orsvärn & Axling, 1999). Moreover, configuration environments have been made more usable and are often integrated into enterprise resource planning (ERP) systems, which are widely used in the industrial area.

2.5. Computational design synthesis

Besides KBE and configuration, Computational Design Synthesis (CDS) is another research field that makes use of AI methods to find design alternatives in engineering design.

As discussed in section 2.3, the focus of KBE research field is in the automation of repetitive and non-creative design tasks and the design optimization of parts, thanks to a strong connection with the CAD environment. Instead, in product configuration (section 2.4) AI is mainly applied to guide the selection of predefined components in order to meet the needs of a particular customer.

Engineering design synthesis has been defined as the portion of the engineering design process having to do with the creation of new alternatives and candidate concepts (Antonsson & Cagan, 2005). Thus, a major goal of design synthesis is to support the creation of a large number of alternatives of high value, while in KBE and product configuration systems the product architecture is almost predefined within certain limits. In fact, the aim of CDS is to support designers at a crossroads not knowing which direction to pursue or when the range of possible solutions for a task is too wide to be graspable manually (Cagan, Campbell, Finger, & Tomiyama, 2005).

The representation of product architecture as a structure showing design components and their interrelations has been named topology (Shea, Cagan, & Fenves, 1997).

Chakrabarti et al. (2011) made an overview of CDS research, dividing the three major approaches of function-based synthesis, grammar-based synthesis and analogy-based synthesis.

The first one includes several methodologies coalesced around the functional decomposition of solutions, taking designers through steps that help decompose a design problem and build conceptual solutions based on the product functionality (see section 2.1). All these models try to find connections between functional aspects of product and behavioral, structural and environmental ones. This approach has led to an automated morphological matrix generation from the design repository (Bohm, Vucovich, & Stone, 2008).

About the second category, generative graph and spatial grammars are used to computationally encoding knowledge about creating designs: defining a vocabulary and a rule-set that operates over it, the generated design language can be used to rapidly generating standard or novel design alternatives.

For instance, Wyatt et al. (2012) represent a product architecture as a network in which nodes symbolize components and links symbolize connections or other relations. The space of possible architectures is the complete set of possible arrangements of elements that satisfy the constraints. Along the same lines, Müenzer (2014) discusses an approach for the generic transformation of product concepts to Bond-graph-based simulation models in order to generate, explore and evaluate large solution spaces.

The challenge for the future in grammar-based synthesis is the automatic learning of grammar rules. In fact, engineering grammars would be more readily applied if rules could be learned and adapted on their own by the software.

Finally, analogy is defined as the cognitive process of transferring information from a particular subject to another one. In particular, the case base reasoning (CBR) is used to adapt old situations to meet new demands, using old cases to critique and interpret new solutions (Kolodner, 1993).

2.6. Early cost estimation

Cost estimation of new products is a critical issue since it is a time-consuming activity which involves technical expertise and a knowledge base. In particular, early cost estimation plays a fundamental role in the offer generation stage and for company strategic evaluations.

The former starts with a request for proposal (RFP), containing the product requirements and ends with the offer generation, including the technical solution and the price of the product. On the contrary, when a company wants to be aware about the convenience of placing a new product on the market, a cost estimation is needed in order to make a comparison between the estimated cost and the cost of similar products already on the market. In this case, it is useful a target cost reasoning, i.e. the cost becomes a product requirement so that the estimated cost must be less or equal to the product cost of competitors.

The overall cost of producing a product can be divided into direct costs and indirect costs. Direct costs are those costs that can be allocated directly to a specific cost center, for example material and labor costs. Indirect costs are those costs that cannot be allocated directly, for example the costs of running the stores and illuminating a workshop.

Product costs can be also divided into variable and fixed costs. Variable costs are costs that change in proportion to the goods or services that a business produces. Fixed costs are those that are incurred in a certain period and do not change, e.g. management salaries, rent of space and interest on borrowings.

Considering the life-cycle cost, i.e. the sum of the costs taking into account the product life-cycle, it is possible to divide into product cost (CAPEX), which is the investment, from the running cost (OPEX), which depends on product performance.

The manufacturing cost is the total cost imputable to a product for material and production. Although manufacturing cost consists of fixed and variable costs, only variable costs are of interest in decision making during the design process.

Duverlie and Castelain (1999) have identified four different methods of cost estimation: (a) the intuitive method, (b) the analogical method, (c) the parametric method and (d) the analytical method.

The first is based on the experience of the estimator, leading to the problem of the subjectivity of the estimation. Although (d) is the most accurate method for cost estimation, it requires much time and a lot of information from embodiment phase, thus becoming not applicable at early design stages or during the offer generation. For this reason, (b) and (c) remain the most feasible methods for cost estimation at early stages.

A comparison between (b) and (c) has led to the conclusion that these methods can be combined using a case based reasoning system to search for similar cases and then adapting the case selected with a cost estimation formula (CEF) on the basis of similar extracted cases. Similar past cases can be easily retrieved by means of similarity measurement, e.g. using Minkowski formula (Ramesh, Yip-Hoi, & Dutta, 2001).

Ehrlenspiel et al. (2007) discussed several methods for estimating costs particularly suitable for product families, i.e. for similar or semi-similar components. The methods are used to estimate the variable part of the manufacturing cost, comprising direct material costs and production labor costs.

The material costs can be easily determined by the weight or the volume of the part, knowing the material unit cost. The production costs can be calculated from the times needed for the individual operations multiplied by a labor cost factor.

Since such a calculation requires too much effort, estimation methods are based on part geometric similarity. In particular, regression analysis is a way to correlate variable costs and influencing factors, basing on a statistical analysis of data. The aim is to find a regression equation the simplest possible which can be considered valid for a wide range of parameters.

Another method for estimating costs makes use of similarity relationships of parts to derive cost-growth laws. The method considers only the variable part of manufacturing costs (VMfC), which comprises the direct material costs (DMtC) and the production labor costs (PLC). The ratio between the VMfC of a new design and the VMfC of a basic design is the following:

$$\varphi_{VMfC} = \frac{VMfC_s}{VMfC_0} = \frac{DMtC_s + \sum PLC_s}{DMtC_0 + \sum PLC_0}$$

Given a basic design whose cost structure is known, the contributions of the material and the individual production cost to the total manufacturing cost are calculated. These coefficients are company specific and are part of the company knowledge.

$$a_m = \frac{DMtC_0}{VMfC_0}; \quad a_{p_k} = \frac{PLC_{k_0}}{VMfC_0} \quad for\ the\ k - th\ production\ operation$$

When the individual terms of the cost growth laws are known, the overall cost growth law is

$$\varphi_{VMfC} = a_m\, \varphi_{VMfC} + \sum a_{p_k}\, \varphi_{PLC_k}$$

The ratios φ_{VMfC} and φ_{PLC_k}, which regard the material and the labor costs respectively, must be derived from similarity laws.

Unlike the regression, such a method allows to make designers more aware of existing dependencies between costs and product design. For this reason, in this work the regression analysis has been used in order to estimate costs of purchased material, while a similarity-based approach has been used to assess costs of parts to be produced within the company.

2.7. Conclusions

The application of configuration systems has shown many advantages in terms of time and usability; besides the object-oriented structure allows to manage complex configurations. The mechanism of encapsulation and inheritance could support, ideally, also the development of a multi-level design system to link the different product representation (functions, modules, instances/variants).

Generative approach is very effective in the generation phase, but it is very limited in providing support during the definition of the product architecture. Moreover, if a valid variant has not been included into the tree structure, it will never be generated.

A configuration problem can be treated conveniently as a constraint satisfaction problem (CSP). Although CSP solvers are able to assign numerical variables within a certain domain, engineering design problems often consist of geometrical constraints, which cannot be fully managed by means of CSPs.

Computational design synthesis (CDS) goes beyond the default configurations of configurators, searching for new variants whose architectures are not predefined. Although CDS offers the possibility of an automated solution search within a larger solution domain, it requires such an onerous preliminary work, that makes it difficult to be used in the industrial field. Moreover, a CDS system is free to navigate the solution space without limits, unlike the normal design practices, which use to explore the domain starting from past solutions.

Due to the above discussed limitations of CSP and CDS, DSM has been then chosen as a support towards the solution search for its simplicity and versatility.

In fact, DSM is well known and adopted to solve many different problems, since it allows to provide advice and support on transversal issues.

As regards the early cost estimation, the methods reported in the literature background are limited to the manufacturing cost estimation, neglecting the product life cycle cost (LCC), which is important to take into account since the early stages.

The approach proposed in this thesis aims to overcome the state of the art limitations given above. In particular, the proposed approach wants to be easily usable in the industrial field, especially during the offer stage. Moreover, the proposed approach aims to obtain a preliminary layout, a BOM and the estimation of LCC of a new product. The solution search begins by looking at past solutions, according to the current best practices of the companies, making changes to meet the technical requirements.

Chapter 3.

The framework

3.1. The requirements of an early cost estimation tool

Early cost estimation is a critical issue for many companies concerning both offer generation and company strategic evaluations. The former occurs when a company receives a request for proposal (RFP) of a possible customer, containing the product specification. Product specification mostly includes only the main features of the product, leaving the choice of many product characteristics to the supplier. Design choices, together with the manufacturing process and several factors related to the company affect the product costs.

In this context, it is important to evaluate costs as accurate as possible in order to generate a reliable offer. In fact, an underestimation of costs would produce an economic loss for the company, if the offer is accepted by the customer, while an overestimation would lead probably to the failure of the offer.

In many cases, a company needs a cost assessment to make a strategic evaluation, for example about placing a new product on the market. In this case, a kind of product specification could consist in features and costs of similar products on the market and the aim of a company could be to sell a product with the same features but at a lower cost, i.e. the target cost.

An accurate cost estimation mostly requires a simplified but complete design process, considering the most significant design choices and the main dimensioning activities. As a result, cost estimation is a time-consuming activity which requires big effort and technical expertise.

In section 1, the issues regarding cost estimation have been identified and discussed. Figure 1 underlines the critical activities of the offer stage, which leads to an increase of the lead time. In particular, several steps of technical alignment can follow some changes in the product specifications. At each step, the product design solution and its cost must be updated. In addition, since the probability of success is never high, it is required to increase the number of the issued offers by reducing the individual offer stage time.

Aware of these problems, a preliminary framework to support early cost estimation has been presented to several companies for an assessment of its main features (Savoretti, Mandolini, Raffaeli, & Germani, 2017). The aim of the industrial survey was to promote a discussion on the needs and the expectations regarding cost estimation in order to obtain feedbacks to be addressed in the implementation of a software tool.

The approach has been introduced to 9 companies, designing and manufacturing products of different categories, ranging from automotive (test benches, motors) to highly customized machines (machining centers, agriculture machines, defense systems, cranes). The involved companies span from SME to large enterprises.

A demo of the framework has been proposed as a series of features, in order to give a score to each feature basing on the company appreciation. In this way, the rank of the features has been used to understand the most appreciated features and the companies most interested in the framework.

The features have been chosen based on the experience within the company that has co-financed the doctoral scholarship and on a broad literature review discussed in the state of the art section.

The list of the proposed features follows.

- The company *knowledge storage* concerns the building of a product variants database for future re-use.
- *Knowledge formalization* regards the use of methods and tools that allow formalizing the product knowledge, e.g. the dimensioning rules and the manufacturing process.
- The inclusion of the product *life-cycle cost* within the cost estimation to go beyond the manufacturing and material costs.
- The management of the *product requirements* allows a product functional description making the offer stage easier for the company.
- The use of *parametric cost* models allows to perform a rapid but effective cost evaluation without going into too much detail of an analytical estimation.
- A representation of the *product structure* in terms of product parts and relations among them, in order to have a cost breakdown structure.
- The use of tools that allow to *support/guide the user* towards the design solution.
- The possibility for the framework to be *integrated* with the most common company tools in order to retrieve or exchange information with them.
- The inclusion of a *CAD tool* to perform a preliminary layout assessment before evaluating the product cost.

Although some differences have emerged in the needs, it has been found that all the companies expressed the necessity of estimating the product cost at an early design stage. In many cases, it is required when a RFP comes from a possible customer and an offer is needed, while in other cases it is important to perform a rapid cost evaluation for placing new products on the market. In the latter cases, the target cost achievement is often required, meaning that the cost becomes a product constraint.

Table 1 summarizes the most relevant aspects emerged from the workshops.

Table 1 - Main aspects emerged from the workshops with the companies

Company	Products	Needs	Suggested integration	Comments and advices
Loccioni IT manager	Highly customized test benches	Offer generation for customized products	Product hierarchy must be derived from PLM.	The tool is useful for offer stage, not in the design stage because of the difficulty in making parametric and reusable technical solutions.
Biesse Cost Engineering	Machining centers (mainly for wood)	Offer generation for product variants	PLM can be used to extract the list of the functional modules of the machine.	Development costs (engineering, test, quality) and the effect of batch quantities must be included in cost estimation.
Fip industriale Mechanical Engineering	Anti-seismic devices	An analytical cost estimation is required	-	A software for analytical cost estimation is already implemented. Given the standardization of the product, additional tools are not desirable.
Maschio Gaspardo CAD/PLM manager	Agriculture machinery - medium batch	Early cost estimation for product variants	A commercial configurator must manage the new product variants.	The company perceives the system more useful for customized products than small or large series.
GE Power Cost manager	Turbogas power plants	Early cost estimation for new power plants	An integration with performance evaluation tools is required.	The company needs parametric cost models at different detail levels depending on few parameters, modules and main parts list.
Leonardo Cost Engineering	Defense systems	Cost estimation for new products	Cost estimation shall be integrated with the CAD environment.	Parametric cost estimation shall not be limited to the company knowledge base. The tool must contain a cost driver repository.
Fosber Purchasing manager	Paper-making machinery	Target cost achievement	-	Parametric cost models must also consider the machine technology and the market field,

				supporting a target cost reasoning.
Fiat Power Train Cost manager	Industrial diesel motors	Cost estimation of new or customized products	-	The framework appears more useful in the customization of an existing product rather than in new products design.
Meloni T.H. Commercial manager	Material handling machines	Offer generation for customized products	The tool must be integrated with spreadsheets used for parts dimensioning	A CAD generation of the machine layout for the customer assessment allows time saves in the offer stage.

Table 2 shows a concise view on the appreciation of the platform features by the interviewed companies. The score 2 has been assigned if a company has shown a big interest in the feature, 1 for a medium interest and 0 for no interest. Firstly, the interviews have confirmed the hypothesis that the current cost estimation approach mostly follows analogic methods based on product similarity. Moreover, cost estimators mainly work on product BOM, adapting previous projects according to the new product requirements.

People involved in the workshops have shown that the framework is consistent with the actual company needs. The interviews have shown the importance of integrating this framework with the systems currently used by the company, which are mainly CAD, PLM, ERP, configurators and spreadsheets. In some cases, companies use customized tools, e.g. for product performance evaluation, which must be integrated with this framework. Companies have underlined that the tool must not overlap with the current tools and must not replicate information, because the data redundancy could lead to errors and cause waste of time.

The discussion concerning the perceived needs has led to several suggestions, resulting in features to be possibly included in the development stage. The needs of a rapid cost estimation combined with the lack of accurate information at the early design stage, require the parametric cost models to be defined at different level of details, i.e. general and approximated at the first stages and more precise as the design process progresses.

Table 2 - Ranking of the software features

Feature	Loccioni	Biesse	FIP ind.le	Maschio Gaspardo	GE Power	Leonardo	Fosber	FPT	Meloni TH	∑ (feature appreciation)
Knowledge storage	2	2	2	1	2	2	1	2	1	**15**
Knowledge formalization	1	2	2	1	2	2	1	1	2	**14**
Life-Cycle Cost	1	2	0	1	2	1	2	1	1	**11**
Product requirements	2	2	0	1	1	0	1	2	2	**11**
Parametric cost	1	1	0	0	2	2	2	1	2	**11**
Product structure	1	1	1	2	2	1	0	0	2	**10**
Detail level	1	2	0	1	2	1	1	1	1	**10**
Support to solution	0	1	0	1	2	1	2	1	2	**10**
Target cost	0	1	2	2	1	0	2	2	0	**10**
Integration	2	1	0	1	2	1	0	0	2	**9**
Cost repository	0	1	1	1	0	2	2	1	0	**8**
CAD tool	0	0	1	1	0	2	1	1	2	**8**
∑ (global appreciation)	**11**	**16**	**9**	**13**	**18**	**15**	**15**	**13**	**17**	**-**

It has been observed that cost estimation cannot be only based on internal company knowledge, because it would limit the design choices, as in case of the use of new manufacturing technologies. A cost driver repository, which contains shared manufacturing cost models and design solutions would allow to explore new technical solutions and perform a wider cost estimation.

A power plants manufacturer has underlined the importance of a LCC estimation, dividing the product cost (CAPEX), which is the investment, from the running cost (OPEX), which depends on plant performance. Furthermore, it has been noticed that the customer often requires a model of the product layout attached to the technical proposal, in order to verify the product geometry and constraints (e.g. the capability of the hook of a crane to cover the whole working environment), thanks to preliminary CAD model generation.

The debate has also shown the importance of considering the product from the perspective of functional modules, which are able to carry a functional description and allows an easier knowledge formalization.

From the discussion, the authors concluded that the framework needs to be adapted and extended in order to meet the following three features:

(a) inclusion of cost drivers that go beyond the current company knowledge base;
(b) supporting target cost reasoning;
(c) cost estimation should be carried out at different level of details.

In particular, the third point (c) highlights the need of managing several cost models at different detail levels following the project stages: from a pure parameter-based model for a rapid cost estimation when few information is available, to a module-based model for a more accurate medium-detail cost level, to an analytic model working in the final BOM for a precise cost estimation of the parts to be produced. In fact, as the level of details become finer, the level of uncertainty decreases and the costs become more accurate.

Finally, from table 2 is evident that companies manufacturing customized products are the most interested in the presented framework, because it satisfies their needs of early cost estimation and offer generation for new products. Moreover, since the framework is suitable for modular products, companies used to reason with product families and functional modules, have shown a higher interest.

This preliminary work has been useful to take an approach for the implementation of a software tool that is the most suitable for the companies having the need of cost estimation at early stages.

In order to sum up, the framework must support the knowledge storage and the knowledge formalization of a company, which both contribute to the building process of a company knowledge base. Moreover, parametric cost estimation has been recognized useful since it leads to a rapid but powerful cost estimation at early design stages, which should be not limited to the manufacturing process, but also the LCC should be considered. Product requirements management has been found to be very important, mainly in ETO companies, which need a product functional description in order to compare product functional characteristics and they also require cost evaluation during the offer stage. Since the product customization of the portfolio is today an important means to be more competitive, ETO companies get the most benefit from the framework for achieving the paradigm of mass customization.

3.2. The method

The methodologies and tools described in section 2 have been used to develop an approach for reuse company knowledge in order to estimate the costs of new products.

Synthetically, the proposed method comprises the following 3 steps (see Figure 8):

1. Knowledge formalization and representation
2. Support towards a solution

3. Cost estimation

The first step represents a preparatory phase in which the company knowledge, coming from different sources, is formalized and represented in order to be reused. Past project data and product knowledge are embedded in a database of product variants and the product architecture is represented by a meta-model in terms of blocks and dependencies. By combining in different ways blocks and dependencies, it is possible to obtain different product variants, with the aim of satisfying different product requirements.

The second part of the method aims to support the user in finding a technical solution that meets the product requirements. A DSM represents and manages the dependencies between the attributes, allowing to divide the problem into sub-problems of reduced size and to give priority to the attributes assignment.

A parameter-based cost estimation follows the attribute assignment. Retrieved past cost data are used to build cost models for the product blocks and to estimate the cost of the new product.

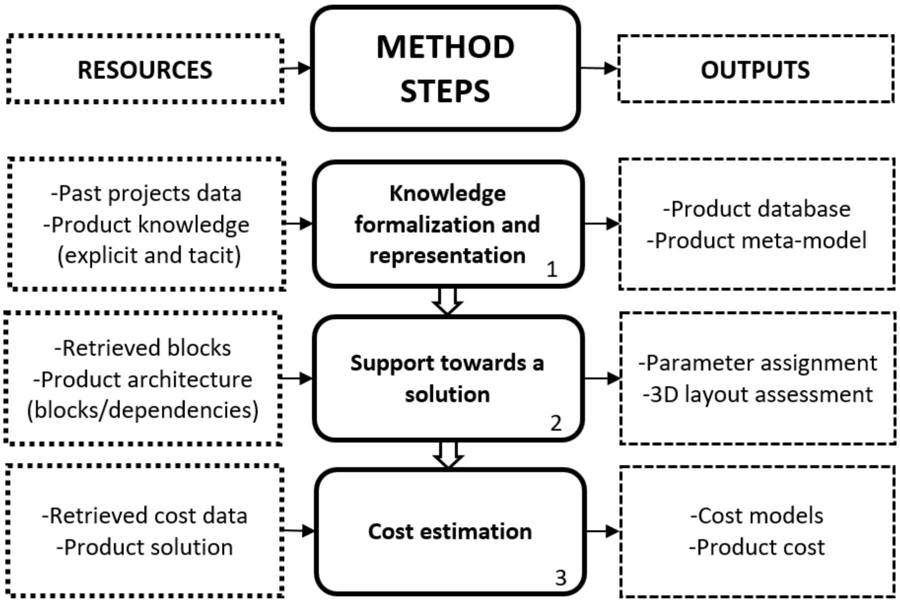

Figure 8 – The method: general scheme

3.2.1. Knowledge formalization and representation

This section discusses how company knowledge can be formalized and storage into the framework and how the proposed framework is integrated with the other company software to exchange information.

In order to configure a certain design solution according to the design-to-cost principles and to the requirements of the specific application, a preparatory work must be done, which is based on the functional modeling of the product to be designed and the representation of the company design knowledge.

It is important to include both the explicit and tacit knowledge related to the product, the latter based on the company best practices as well as the human expertise.

Company knowledge is mostly included in other software tools, that manage certain company data and perform specific tasks. In order to avoid redundancy and inconsistency of information and to easily retrieve data, the proposed system has been integrated with consolidated systems within the company, trying not to overlap already covered functionalities of the following tools:

- CAD, which provides a geometric environment for the embodiment design phase;
- CAE, which provides an environment containing simulation tools for design validation and optimization;
- PLM, which provides product classification and product life-cycle management;
- ERP, which provides an integrated environment for the management of the core business process

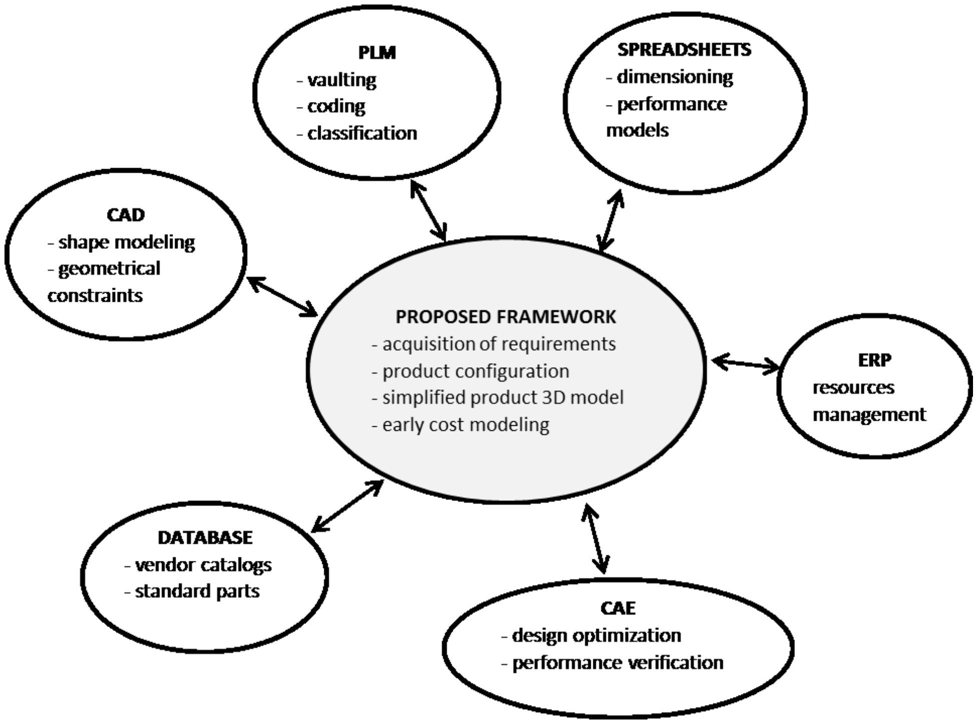

Figure 9 - Framework integration

Figure 9 represents how the proposed framework has been thought in order to be integrated with other company tools. The framework must cover the following functionalities.

- Acquire and manage the product requirements
- Support the user towards the product configuration, i.e. the instantiation of the product attributes
- Support the user in defining a simplified product 3D layout
- Define cost models for early cost estimation

It is worth pointing that these functionalities are particularly useful in the offer stage of ETO products.

The framework needs many information from other tools. For instance, PLM systems provide product structure and product variants classification, which are very useful in the preparatory phase of the framework concerning the company knowledge formalization and the product representation through a meta-model. ERP systems give information about materials and resources used in the past projects, that will be retrieved to estimate the new product cost. Several companies use vendor catalogs and standard parts databases which contain technical and economic information that are useful to build cost models.

The framework shares geometrical information with CAD tools, to provide a preliminary 3D layout. Geometrical attributes of the product are used to generate parametric CAD models having a simplified geometry. The simplified 3D layout, which is the output of the conceptual design phase, becomes the starting point of the embodiment design phase. CAE tools receive information from the framework such as product attributes or a preliminary 3D layout while they give results coming from design optimization and performance verification analysis.

Finally, the integration can be extended to a large variety of customized company tools, such as performance models or simulation software. In fact, the system shall work with a large variety of data coming from different tools. For instance, several companies make use of spreadsheets for component dimensioning and evaluating several possible scenarios.

Integration has been implemented by using API library which are commonly provided by commercial tools. Thus, the proposed framework has been thought as a part of an integrated environment, which links product knowledge coming from design, manufacturing, marketing, maintenance gathered from the company departments: technical, production, service, commercial and purchasing departments. Managed knowledge extends to the product life cycle, enhancing standardized interfaces to acquire information.

Because of the importance of product geometry in many industrial fields since the early stages, an environment for a 3D layout assessment has been provided. Parametric CAD models of parts that contain geometric descriptions are generated respecting overall dimensions and interfaces for part connections, even if they have a simplified representation.

The configuration of a simplified layout provides the relevant information for a product geometry assessment. This information can flow from the 3D layout to the product attributes in order to verify constraints or calculate some quantities, e.g. the painting costs by means of the rough external surface of a geometry. Parametric template CAD models are linked to the system. Such templates are interactively defined by the user to the necessary level of detail. The tool drives commercial CAD systems by API connections and allows updating geometries as parameters are changed, while the internal 3D environment is just limited to some functionalities that are discussed below.

Before discussing the method for knowledge formalization, the next sub-sections clarify the terminology and the fundamental concepts used in this thesis.

3.2.1.1 Product meta-model representation: the terminology

This sub-section explains the terminology used in this thesis to represent the product knowledge.

The product architecture represents the product at a conceptual level, consisting of blocks and dependencies. The blocks incorporate some attributes that serve to instantiate them and the dependencies restrict the possible attributes values and thus the solutions space.

The product layout represents the product geometry and is connected to the geometrical attributes of the blocks in the product architecture.

The precise definition of each meta-model object is given below.

- The **block** is an instance of a technical solution for a sub-set of the functional structure. The block may be a novel solution or recover already implemented solutions, possibly in parametric and standardized form.
- The **block attributes** are the set of parameters, properties, characteristics of the blocks. Geometric attributes are used to generate parametric CAD models.
- The **dependencies** are used to represent constraints among the blocks of the structure and the attributes. They can also be used to implement integrations with spreadsheets and with the CAD environment. A classification of the dependencies is given below.
- The **product architecture** is a hierarchical tree of blocks capturing the needed product blocks and the dependencies between them.
- The **product layout** is a simplified 3D CAD model of the product

The company knowledge is embedded in a model-based architecture, which is able to describe the product families representing the company business.

Products are represented by a graph-tree structure in the tab architecture. A component tree, whose nodes are the assemblies, sub-assemblies down to all significant components, represents the product architecture. Screws, bolts, fixtures, wirings can be neglected and their functions represented by a subset of the relations between parts. Each block, i.e. each tree node, is defined by a name, attributes, implemented functions. It is linked to some documentation files. A block stores geometrical and mainly non-geometrical characteristics of the part. It represents a node of an oriented graph structure that is superimposed to the block tree structure. Dependencies connect these nodes and are grouped in bundles between blocks groups.

Each block attribute basically represents a geometrical parameter or a material, a specification, a characteristic of a block. Attribute data are name, value, units of measurement, data type and express the content of a certain parameter. In addition, fields such as category, notes and suggestions are provided for the designer convenience and store information about the product part. In particular, the category is used as a way to filter properties for specific design domains as mechanics, electronics or material.

A dependency is a connection between attributes. It is possible to manage many-to-many connections such as dimensioning formulas. They are defined by a name, an optional value (when it makes sense), a category and notes to provide the designer a way to input more extended explanations. Dependencies are input by designers on the basis of their experience and shared among the design department. Each designer will mostly contribute to the model within his specific knowledge domain.

The generic concept of the block discussed above has been implemented by two different entities in order to describe the product architecture: the code and the occurrence.

- The **code** is an instance that is in the design repository and can be retrieved in order to be reused
- The **occurrenc**e is an instance of the hierarchical tree that describes a particular product solution.

Each block can be included in the structure with multiple variants definition (codes). This supports the general situation of different alternatives for parts such as motors, actuators or sensors, that are selected in catalogues on the basis of needed requirements. Each variant shares the same inner structure in terms of attributes but differs for the assumed values. In this way, variants can be exchanged maintaining consistency in the relation graph.

The first attribute (referred as base attribute) is automatically defined as a new block is instantiated. It contains a unique identifier of the part or assembly being employed, usually its code in the PLM system. Dependencies are instantiated among base attributes as if they connect blocks in a direct way. Moreover, for a given block with multiple alternatives, base property is used to identify the currently selected variant (code).

The product solution is captured by the concept of occurrence. An occurrence defines the relative orientations of its child occurrences. The product architecture is therefore a hierarchical structure constituted by a tree of occurrences. Such hierarchy allows pieces of configuration to be realized separately, combined and reused.

A code is an instance of a block as it appears in the company repositories (e.g. ERP or PLM), including both its geometrical representation and its non-geometrical attributes. Each occurrence contains the definition of an instance through a list of codes that act as variants of the occurrence. Only one code is active at a time.

Attributes can be defined at occurrence or at code level. Since functional attributes (e.g. the power of a motor, the nominal torque of a gearbox) are characteristics of the block, regardless of how the block is used, they should be defined at code level. On the contrary, the number of sensors connected to a PLC is a characteristic of a particular product architecture and the minimum distance between two blocks is a characteristic of a particular 3D layout. Consequently, they should be defined at occurrence level.

3.2.1.2 Product meta-model representation: the dependencies

This sub-section clarifies even more the concept of dependency and illustrates the types of dependencies that are embedded into the framework. The dependencies are relations that restrict the design solution space. Relations can be established between objects within the product meta-model or with external tools. In addition, dependencies can have or not a direction, i.e. unidirectional or multidirectional dependencies. The latter is the most general case and multidirectionality is a characteristic of the constraint-based approach. However, unidirectional dependencies are useful to represent constraints that come from product specifications, which are given by the customer.

The following dependencies have been represented within the framework:

- Math formulas and logic rules
- Topology rules
- Structure traversing rules
- Geometric relations
- Spreadsheet relations

- CAD relations

While math formulas consist in equations or inequalities which relate blocks attributes through mathematical operators, logic rules are conditional expressions between blocks attributes (e.g. if-then rules). Both can be set among the attributes of a single block or different blocks, but while in the first case, dependencies belong to the block which holds the attributes, in the second case dependencies belong to the lower parent level in the product architecture containing the blocks involved in the dependencies. For instance, this dependency type can be used to represent blocks dimensioning rules or to check some constraints. Figure 10 shows the multidirectional dependency given by the math formula $x_1=y_1+y_2$. The dependency is established among the three attributes of two different blocks, i.e. X and Y. According to this dependency, the free choice is possible only on two of the three attributes.

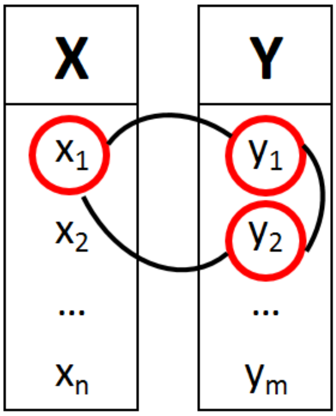

Figure 10 – Math formula example

Topology rules are used to introduce constraints in the hierarchical tree of blocks, i.e. the product architecture. For instance, topology rules can limit the compatibility between blocks within a product architecture or the parent-children relation, e.g. by giving the minimum/maximum number of children blocks. Figure 11 shows an example in which the number of child block C_2 must be from i to j.

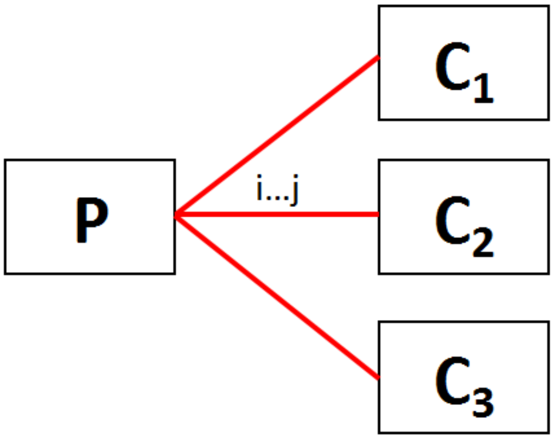

Figure 11 – Topology rule example

Structure traversing rules are introduced as a means for computing cumulative quantities provided by homogeneous attributes that belong to sub sets of blocks. The subsets are identifiable in the product structure as branches of the blocks hierarchy itself, or thanks to dependencies connecting parts from a logical point of view (e.g. parts connected by a hydraulic circuit).

Structure traversing rules can also be used to set constraints on attributes that are not known beforehand.

A structure traversing rule needs 3 elements:

1. An attribute name (or category) to be searched for
2. A method for searching, based on product hierarchy or on functional description
3. An operator (sum, min, max, …) that works on the found attributes

Figure 12 shows the calculation of the parent P weight through the sum of the weight attributes of the children C_1, C_2 and C_3.

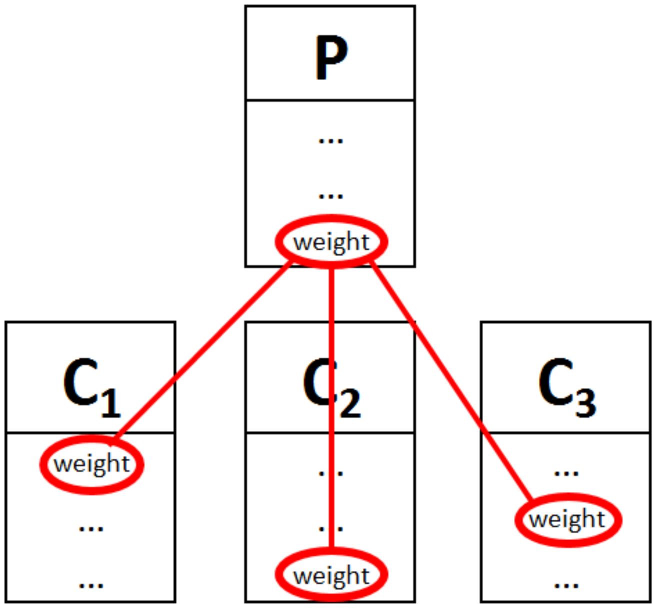

Figure 12 – Structure traversing rule example

Geometric relations (such as "attached to" or "contains") are set between geometric entities of the CAD environment. Plane and axis alignments are the constraint types that have been implemented in the CAD environment, allowing an arrangement of the product blocks in order to produce a simplified layout. The example given in Figure 13 shows the motor-gearbox coupling that consists of an axis alignment and a mating between the surfaces of the flanges.

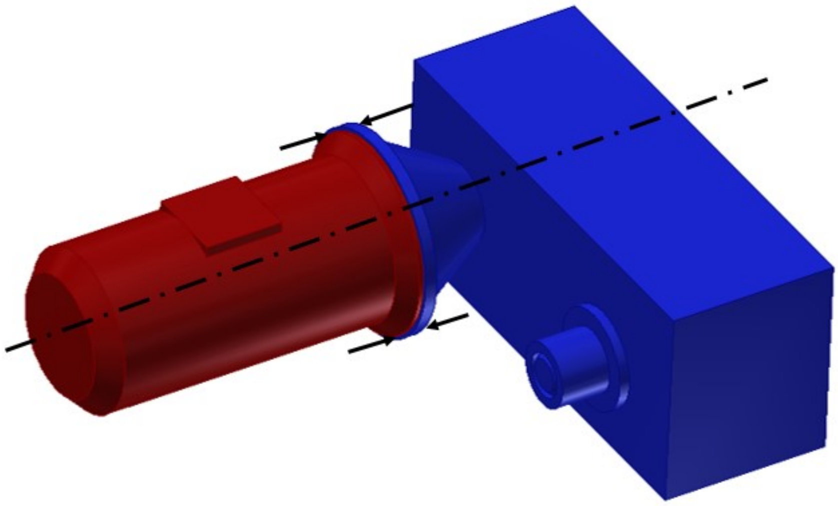

Figure 13 – Geometric relation example

Relations with spreadsheets are used to integrate the framework (blocks and attributes) with company spreadsheets, which contain calculation models, or tables. This type of relation allows both to retrieve data from spreadsheets, i.e. reading from tables, and to write data in the spreadsheets. In the latter case, input data can be inserted in spreadsheets in order to retrieve output data that have been calculated. In Figure 14, the value of the x_1 attribute of the X block is written in the A1 cell as an input value for the spreadsheet. Then, F3 cell value is retrieved by the y_2 attribute of the Y block.

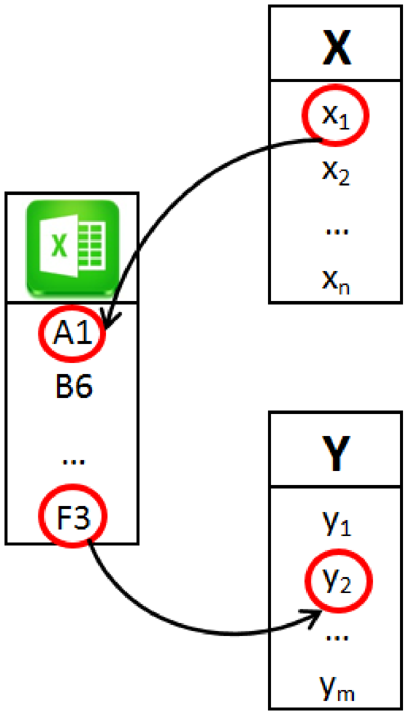

Figure 14 – Spreadsheet relation example

CAD relations regard all the tools that are used to integrate the framework (blocks and attributes) with the geometric entities of the CAD environment. Relations with the CAD environment allow information to flow from the architecture environment (blocks and attributes) to the product layout and vice versa. While in the former, geometric attributes of the blocks are related to geometric entities in order to generate parametric 3D CAD models, in the latter, geometric entities are retrieved to calculate some quantities or to check some constraints, e.g. minimum distances or overall dimensions. Moreover, CAD relations can be established between CAD constraints and other dependencies, e.g. math formulas or logic rules. For example, an axis constraint in the CAD environment between two shafts, could be related to speed and torque attributes of the two blocks. Figure 15 shows the generation of the X block CAD model from block attributes and the retrieving of surface and weight attributes from the CAD model information.

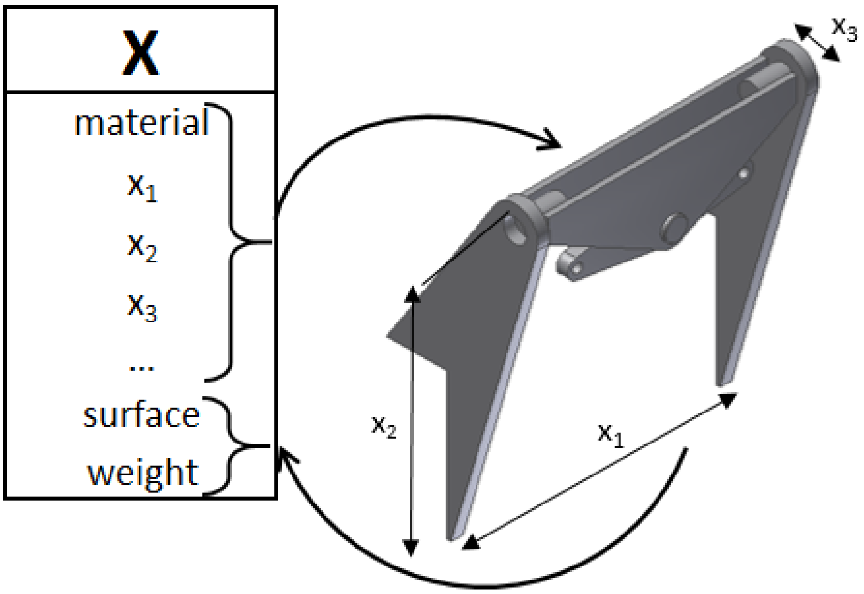

Figure 15 – CAD relation example

Several examples of the above cited dependencies will be given in the case of study section.

The product meta-model described above is a means to share information on the product and store the so called implicit or tacit knowledge, that is the part of facts that are not explicitly formalized and just maintained in the designer head. Product representation is dependent on designer background and helps sharing the knowledge of the product within the company.

3.2.1.3 The steps for company knowledge formalization

This sub-section describes the method that has been used to formalize the knowledge of a company in order to represent it through the meta-model above defined.

Raffaeli et al. (2016) provide an approach to acquire and formalize the design and manufacturing knowledge of a company moving from the data acquisition of the past projects. The method for company knowledge formalization that is used in this thesis extends the above-mentioned approach for its use within the framework.

The steps regarding the company knowledge formalization can be synthetized as follows (see Figure 16):

1. Identification of the product functional structure
2. Identification of the blocks for product architectures from the product functional structure
3. Identification of the attributes of each block
4. Definition of the dependencies between blocks

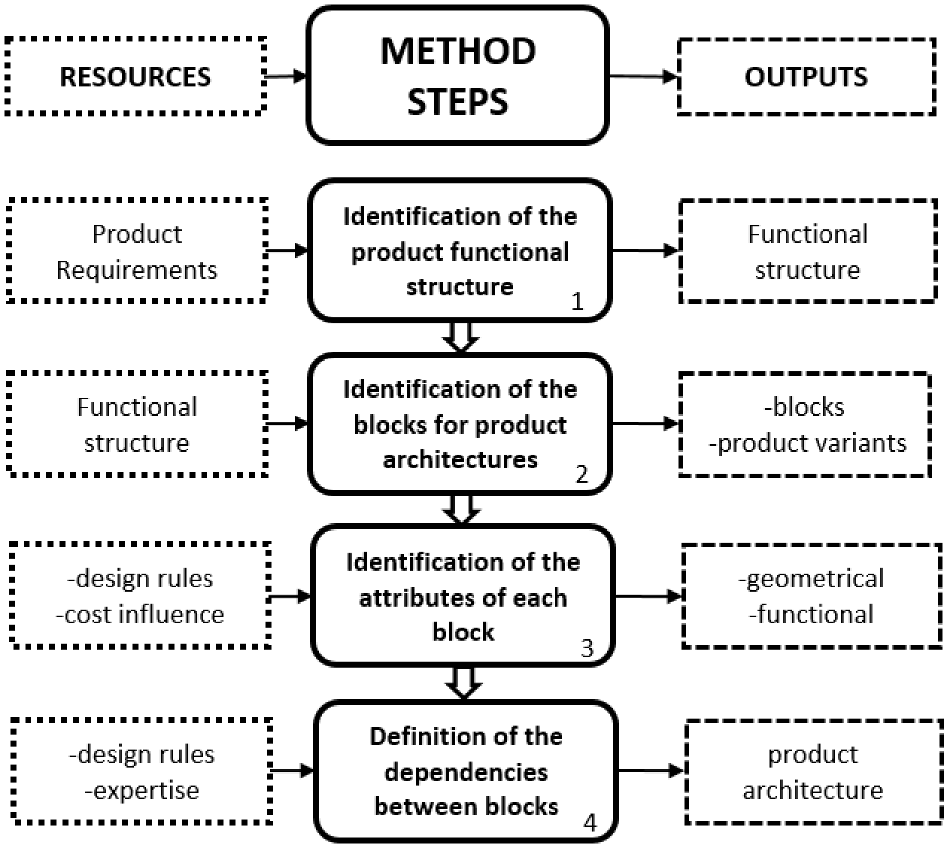

Figure 16 - Method: knowledge formalization and representation

In order to build a requirement-based functional model, the conversion of the product requirements into a functional structure is needed. As a result, the product requirements are connected to the physical blocks of the product.

The first step, which is also the most onerous, concerns the data acquisition. Customer requirements of past projects are collected. Product datasheets, specifications and technical proposals are the main sources of information. Then, design data, CAD model, drawings, and BOMs are gathered to acquire possible product architecture data. This step could benefit of product data management (PDM) systems. These data are collected for several product variants in order to make an exhaustive analysis. Customer requirements are converted into functional requirements in order to build a product functional structure and then (step 2) functions are grouped in order to identify modules (Stone, Wood, & Crawford, 2000) that are linked to the physical components, i.e. the blocks of the product architecture.

Step 3 concerns the identification of the most important attributes of each block. Block attributes are used to determine the main drivers in the module instantiation and to estimate its costs. If a block can be geometrically represented, geometric attributes should also be identified in order to build a parametric template CAD model that is linked to the system.

Block attributes can be divided into inputs and outputs, where inputs are the required data for the block instantiation and outputs are the resulting data from the instantiation (e.g. weight, cost). The inputs come from the product technical specifications (requirements) or they can be output attributes of other blocks, which means that there is a dependency between the blocks. Product documents, drawings, spreadsheets, design standards along with senior expertise are used to identify dependencies between input and output parameters (step 4).

Once blocks and dependencies have been identified, product can be represented as a network of blocks connected through input-output attributes. An IDEF tool can be employed for an initial exploration of the design process as a sequence of dimensioning activities, to be aware of dependencies between the block attributes. As some activities are mutually dependent, they must be solved together in an iterative manner.

As mentioned above, in the proposed framework the product model (known as meta-model) is described in terms of blocks and dependencies. The block is a basic part which has relevance to the designer in the conceptual phase and is described by a set of attributes. The block, initially defined in abstract terms, becomes a real part code when attributes have been completely defined. Blocks interact each other by means of dependencies, which are relations or constraints between attributes. Dependencies can connect two or more blocks, forming in general a hypergraph (see Figure 17).

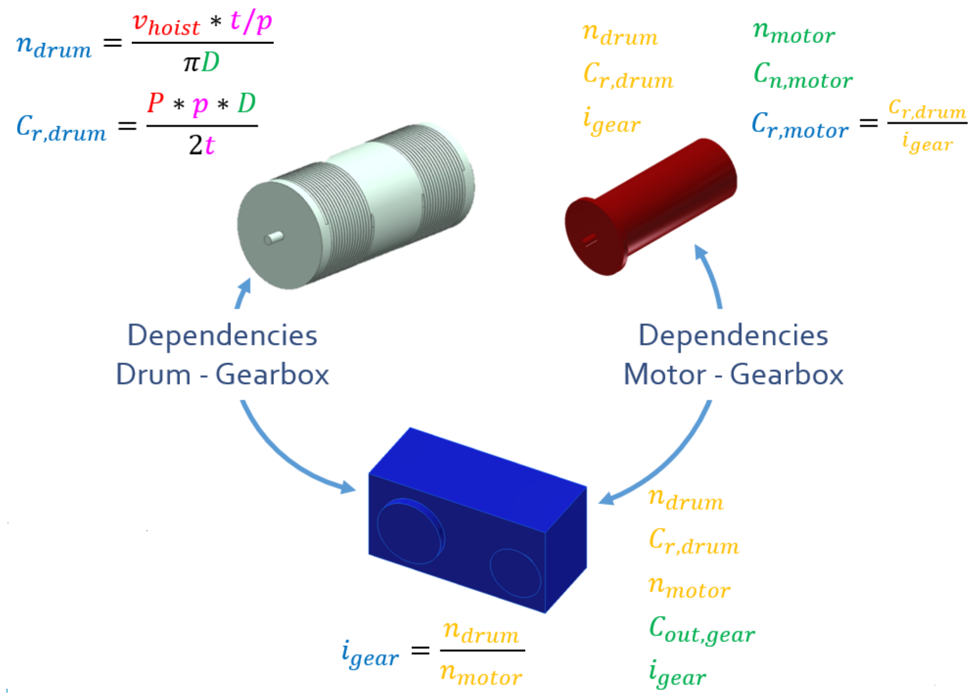

Figure 17 - Example of dependencies between blocks

Dependencies are then collected in a DSM in order to be used in the solution research phase.

A set of predefined blocks and dependencies constitutes the starting database for product configuration and cost estimation. Therefore, a block can be retrieved from past project for fostering reuse of existing solutions (see Figure 18). The storage of the defined parts forms a DB that continually grows. The system, thanks to a recommender system, supports the user in finding and selecting codified parts from the DB, with the advantage of reducing the codes number and moving towards parts standardization.

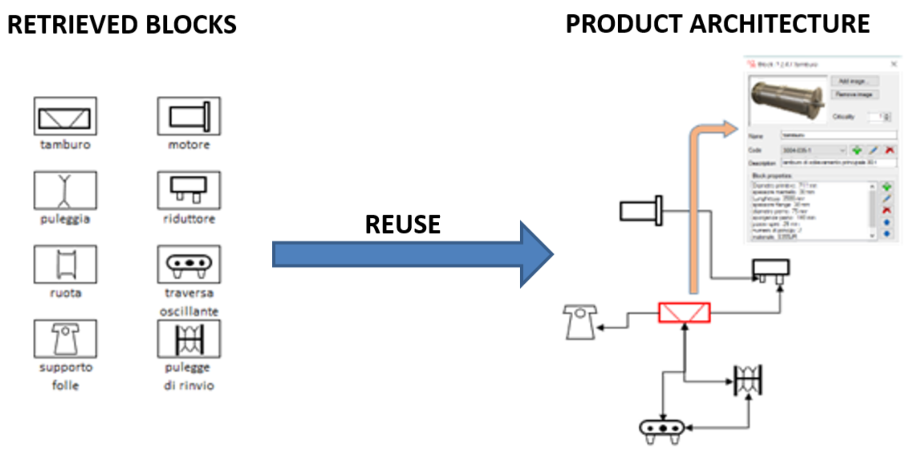

Figure 18 - Reuse of the past project data

In this section, the steps for formalizing and storing the company knowledge have been discussed, which lead to the definition of the meta-model and to the formulation of the problem. The framework takes advantage of the possibility of retrieving past solutions that can be selected by filtering the database or on the basis of the requirement similarity. Product requirements constitute a special block (or a series of blocks) of the product meta-model that affect all the other blocks. In fact, since requirements does not depend on the product attributes, dependencies between requirements and product attributes are unidirectional. Product requirements management is very important, because it is related to the product portfolio of a company.

Minkowski formula has been used in order to perform similarity measurement in the n-dimensional space of the product requirements.

Given X the product requirements of the new product and Y the product requirements of a past project

$$X = (x_1, x_2, \dots, x_n) \text{ and } Y = (y_1, y_2, \dots, y_n) \in R^n$$

the distance between the product requirements of a new product and that of a past project is the following:

$$\left(\sum_{i=1}^{n} |x_i - y_i|^p\right)^{\frac{1}{p}}$$

where p≥1 determines the type of distance (e.g. p=1 is the Manhattan distance, p=2 is the Euclidean distance)

Requirement for similarity measurement should be selected basing on the importance in affecting product main characteristics and costs. Section 3.2.3 illustrates some methods to assess the influence of block attributes in the product cost. In general, portions of product solutions, i.e. blocks, block groups and dependencies, can be retrieved separately and then re-combined to form the desired product architecture. In some cases, similarity measurement could not retrieve any solution that comply with the product specifications, i.e. the Minkowski distance is very big. Then, if needed, new blocks must be created in order to satisfy the product requirements. In other situations, product specifications could require a radical redesign. In these cases, design costs could be very expensive, then an evaluation about the convenience of issuing an offer to the potential customer must be performed.

The possibility of discarding quickly the RFP having less economic convenience allows to spend more time for the offers having more convenience and more probability of success because the company has experience in similar past projects. This results in increasing the success rates in the offers, which is a huge benefit of the tool.

3.2.2. Support towards the solution

Once the product requirements have been defined, together with the product architecture and the dependencies, the aim is to find a set of valid solutions to the problem that can be optimized basing on one or more parameters, e.g. the product cost. The aim of this section is to describe how the framework can support the user in finding one or more solutions and then optimize them.

The step that regards the support towards the solution are the following (see Figure 19):

1. Instantiation of the DSM
2. Clustering of the DSM
3. Partitioning of each cluster
4. Clusters resolution
5. Checking for constraints satisfaction and global DSM

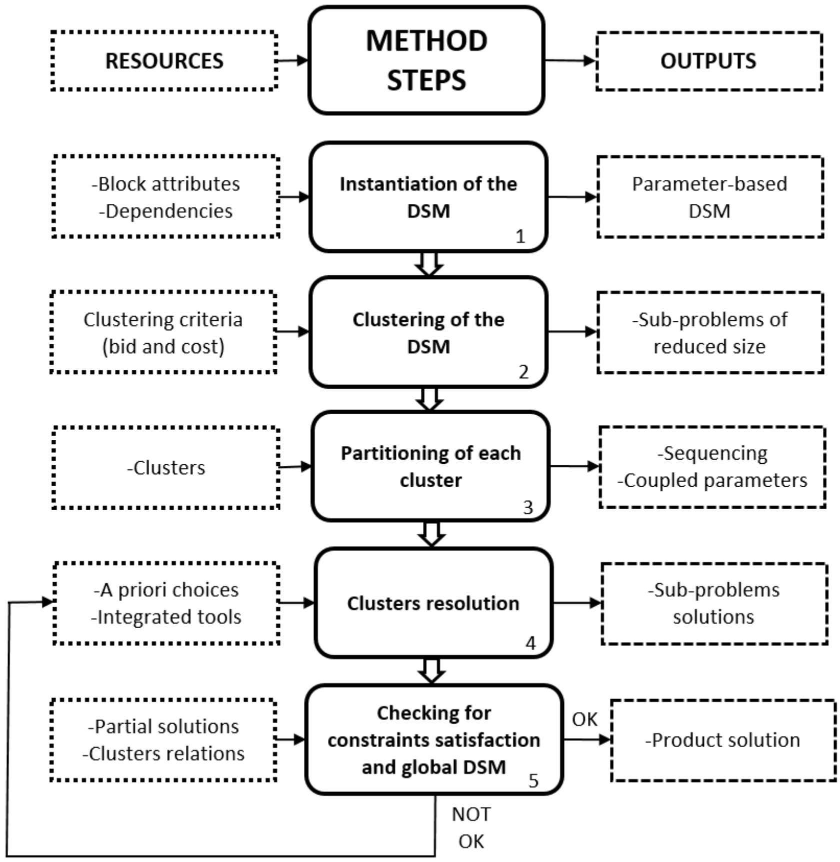

Figure 19 - Method: support towards the solution

The instantiation of the parameter-based DSM (step 1) is automatically made by the software, while the user establishes dependencies in the product architecture. Figure 20 shows the instantiation of the DSM starting from the dependencies among X, Y and Z blocks attributes given in the product architecture.

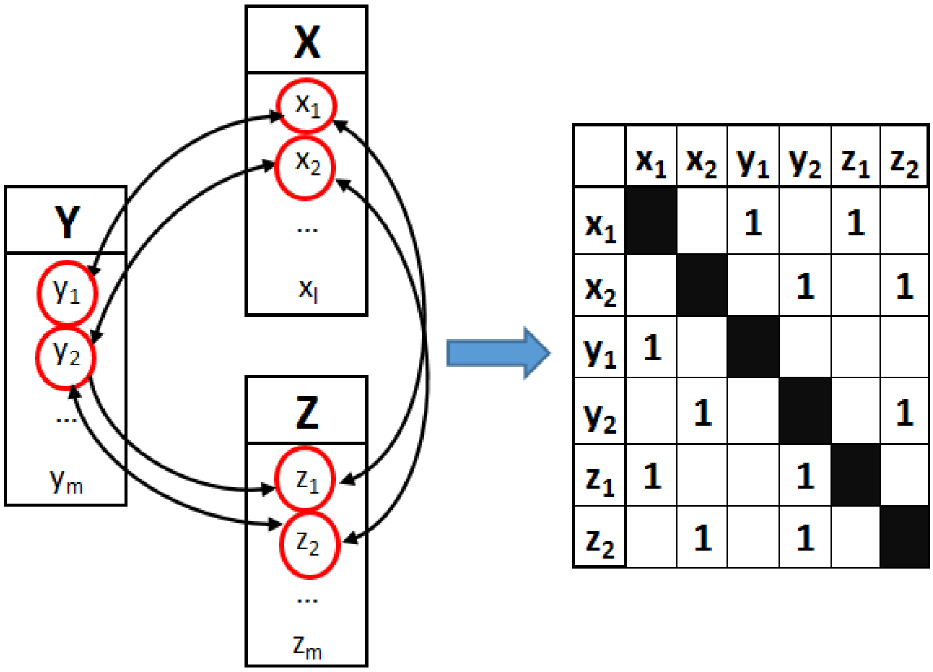

Figure 20 – Instantiation of the DSM

The introduced dependencies generally constitute an intricate and complex network with an elevate degree of coupling between elements. Such complexity can be identified as the main difficulty a designer is facing when trying to figure out a preliminary solution in the early phases. Since many parameters are mutually dependent, they must be solved together in an iterative manner, which is not feasible in the short time normally available in the conceptual phases. DSM is employed to organize design parameters dependencies because it allows simple, compact and visually effective representations of complex graph of networks.

Clustering the DSM allows to divide the problem into simpler sub-problems of reduced size, which are expected to be much more manageable. This approach is close to the designer perspective, which is used to dealing with problems of limited size. Relationships between the sub-systems as well as input and output data flow are highlighted and monitored. Figure 21 shows the second step of the method, i.e. the clustering of the DSM. Starting from the above-instantiated DSM, the clustering algorithm has led to two clusters of coupled parameters, i.e. two sub-problems to be solved separately. However, the dependency between z_1 and y_2 attributes constitutes a dependency between the clusters and need to be monitored.

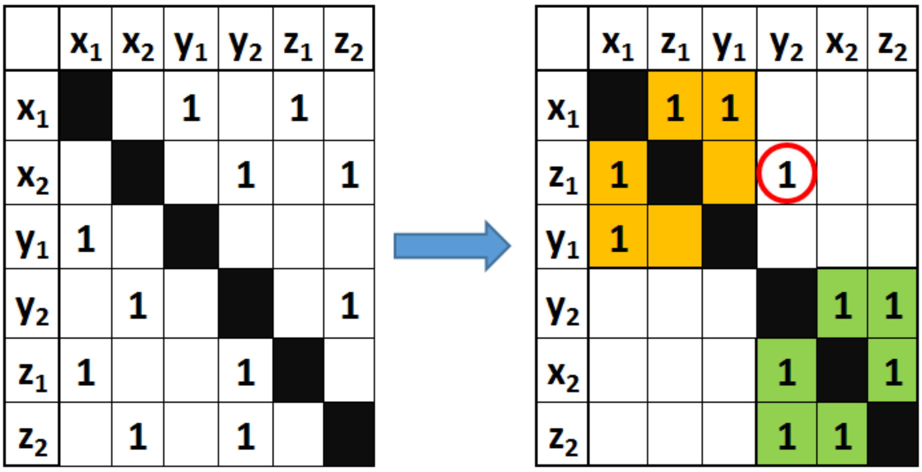

Figure 21 – Clustering of the DSM

Clusterization is based on a simulated annealing optimization procedure derived from the one proposed by Thebeau (2001). The advance of such approach lies on the capability of controlling the number, size and composition of the clusters based on custom algorithm parameters and functions. In particular, "bid" and "cost" functions are defined in order to formulate respectively an offer for a parameter to be in a cluster and to assess the optimality of a certain configuration of clusters. Such functions are formulated in order to foster the presence in the same cluster of:

- parameters linked by pure mathematical rules, which can be effectively resolved by existing CSP solvers;

- geometrical and spatial relationships, which can be effectively solved in parametric CAD environments;
- parameters which are linked by engineering tools introduced in the blocks structure such as spreadsheets or connections to external CAE software.

Once the problem has been divided into sub-problems, i.e. clusters of parameters, each of these can be solved separately (see Figure 22). However, clusters maintain couplings between them and should be solved respecting the order in which they appear in the clustered DSM.

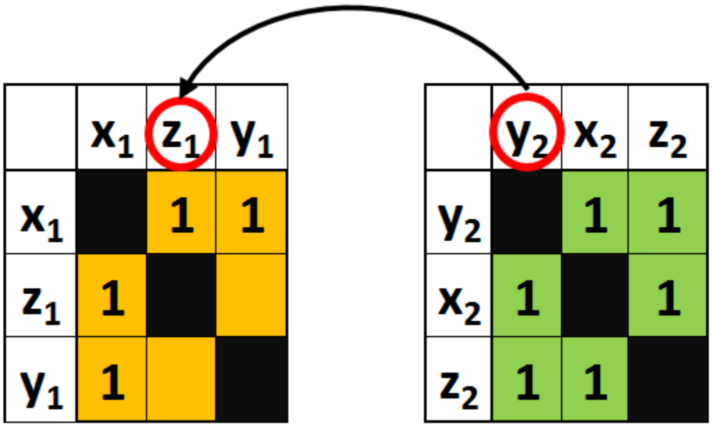

Figure 22 – The sub-problems

In order to solve each sub-problem, a partitioning algorithm is applied to each cluster (step 3). The partitioned DSM shows all the product parameters sequenced according to their dependencies, thus it is possible to know when a parameter must be defined in order to proceed with the design process. The partitioning algorithm groups around the diagonal those attributes that need to be defined in the same design step, as highly coupled. Thus, the obtained parameters sequence allows minimizing the iterations during the phase of the parameters determination.

In general, the partitioned DSM shows several blocks of coupled parameters, i.e. the parameters that are affected by mutual dependencies. In order to determine these parameters and proceed with the design process, a priori choices are needed. A priori choices are properly characteristics of the design process, e. g. the material selection of a part. Usually assumptions must be verified after an iteration. For example, a constraint on the weight of a part can be verified only after several choices have been made, i.e. the material selection.

In order to assess what parameters are the best candidates for a priori choices, an algorithm has been proposed. The algorithm aims to minimize the number of parameters to make assumptions and suggests the parameters to the user. In addition, the method serves to identify the steps in which human choices are needed.

The algorithm tries to remove dependencies from each one of the n coupled parameters and, after new partitioning, verifies if triangularization condition is satisfied. Triangularization condition of the DSM means the possibility of determining all the parameters of the block without other parameter assumptions. If only one parameter is not enough to reach the condition, algorithm searches for pairs of parameters, triplets, and so on. Figure 23 shows the first cluster before and after removing the dependencies from x_1 attribute. Since the selection of z_1 or y_1 would not lead to the triangularization condition, x_1 is the best candidate for a priori choices. Conversely, the second cluster needs two a priori choices to reach the triangularization condition. In this case, there is not best candidates for the choice. Figure 24 shows the second cluster before and after removing the dependencies from y_2 and x_2 attributes.

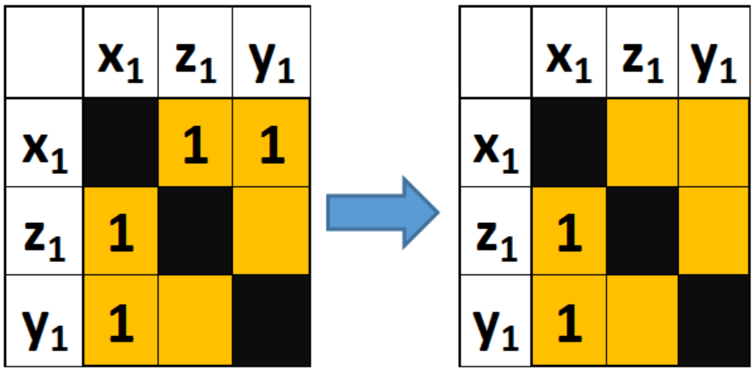

Figure 23 – Cluster 1 before and after removing the dependencies from x_1

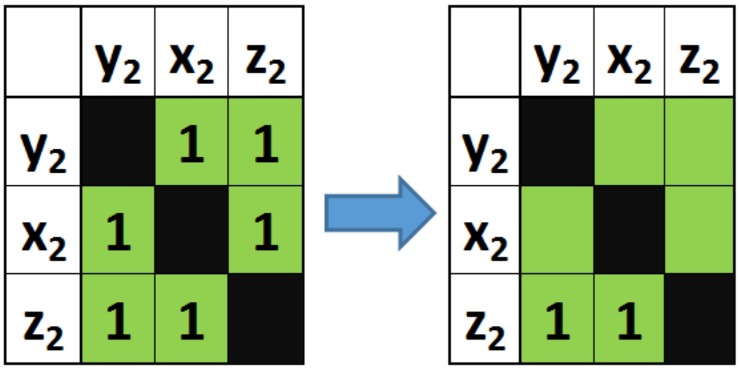

Figure 24 -Cluster 2 before and after removing the dependencies from y_2 *and* x_2

Cluster resolution, i.e. variable assignment, can be done in different ways, mostly using external tools to be integrated to the framework (e.g., CAD, CSP, spreadsheets, CAE) but the overall process is controlled and managed by the proposed framework in a semi-automatic manner.

For example, if the problem to be solved consists only of dependencies of math rules type, a CSP solver or an optimization tool can be used for a variable assignment. In this case, the user must select the parameters to be defined to begin the iteration and choose an objective function. CSP and optimization tool automatically explore predefined domains, e.g. using backtracking strategies. In order to restrain the explosion of propagation paths, the user can suggest a range for variable assignment on the basis of his/her knowledge and experience.

On the contrary, if dependencies among the coupled parameters include geometric relations, the problem cannot be solved by a CSP solver and needs the human intervention. In this case, the user should define a plan of experiments to explore the solution space and return the parameter values that the system cannot calculate. This step concerns the definition of the range of variation and the number of levels so that the number of experiments can be minimized according to the DOE theory. The exploration must continue until the iterations lead to valid assignments, i.e. compatible with the assumptions. A tool of optimization, for example RSM-based, can be used to find the best solution through the fitting of the experimental data.

In conclusion, the support towards the solution moves from the identification of sub-problems characterized by a greater interaction among the parameters of each cluster and a smaller interaction among different clusters. Then, each sub-problem can be individually addressed starting from the identification of the minimum number of parameters to be a priori defined in order to determine all other parameters of the block.

The solution search ends when a valid solution is found, i.e. variable assignment comply with all dependencies within the block. However, it is possible to proceed with the exploration of the solution space in order to find other valid solutions, with the aim of minimizing or maximizing an objective function, e.g. the product cost. In this thesis, cost optimization has been applied in order to make an offer more competitive, but other parameters can be used for optimization, e.g. the product weight, performance or environmental impact. Furthermore, in order to take a multiple-criteria decision, multi-objective optimizations can be used, which allow to find a Pareto frontier instead of a single optimum.

Once the sub-problems have been solved separately, relations between parameters of different sub-problems must be checked, in order to verify the compatibility in the whole system. In case some parameters do not match, i.e. they do not comply with the assigned dependencies, a new assignment for them is needed.

Although variable assignment is mostly done manually, the framework monitors the compatibility of each assignment with respect to the dependencies, supporting the designer in a wizard-like design process.

When all the parameters have been assigned so that they comply with all the dependencies within the product architecture, a solution for the entire problem has been found, including also the product cost estimation, which will be discussed in the next section.

Since the customer intervenes heavily on the product during the offer stage, such a tool is useful to reduce the timing of technical alignment stages, allowing a rapid review of the product design and costs according to any possible change in product requirements. In fact, by means of product requirements and dependencies management, it is possible to evaluate the impact of each change.

The presented approach supports the configuration of product families according to the design constraints given in the request for proposal. The approach proposed is particularly useful when dealing with highly coupled problems, which are difficult to be treated as a single block. Moreover, the approach is especially suitable when problems consist of dependencies of various type, that are impossible to be automatically managed and solved. In these cases, human intervention is very important in order to restrict the solution space or to check some dependencies, e.g. clearances between parts in the 3D layout.

3.2.3. Cost estimation

The cost estimation methods proposed in this thesis make use of past projects data to estimate the costs of new products. Because the costs are attributes of the product, depending on blocks attributes and relationships, they are stored for a possible future reuse. For these reasons, the building of a cost database must precede the cost estimation activities. Database building consists in the acquisition and categorization of past projects data.

Since the aim of the framework is to perform an early cost estimation, a parametric method is the most suitable for this purpose, allowing a rapid evaluation that follows the configuration process. In fact, since a block is described by a set of attributes, the cost of each block can be expressed as a function of its attributes (such as weight, length, area, etc.…).

Blocks can be produced within the company or purchased by a supplier. In the former, cost structure is known, while in the latter, only the total cost is known. For this reason, cost estimation follows two main paths distinguishing purchased materials from produced parts. The proposed cost estimation method for purchased parts is based on regression models, which can be derived by fitting the costs of past purchased codes using one or more relevant parameters (e.g. the power of an electric motor and the output torque for a gearbox). Cost estimation for produced parts are deducted by more complex cost models based on cost structure, allowing a better management of the company resources involved in the manufacturing and life cycle processes.

A more detailed explanation will be given in the next two sub-sections.

3.2.3.1 Cost estimation of purchased materials

A cost estimation method for purchased material arises from the necessity to know the costs of some parts like motors, gearboxes, couplings, whose costs are often not included within the catalog. Moreover, outside processing costs to obtain a semi-finished product can be estimated by means of this method. In order to estimate the cost of a new block variant that has been instantiated during the conceptual design stage, a cost estimation model for the block is needed, which uses the known costs of the variants that have been purchased in the past.

The steps for the creation of a cost estimation model for purchased materials are the following:

1. Database building
2. Identification of the cost influencing parameters
3. Database filtering
4. Calculation of the β coefficients of the linear regression

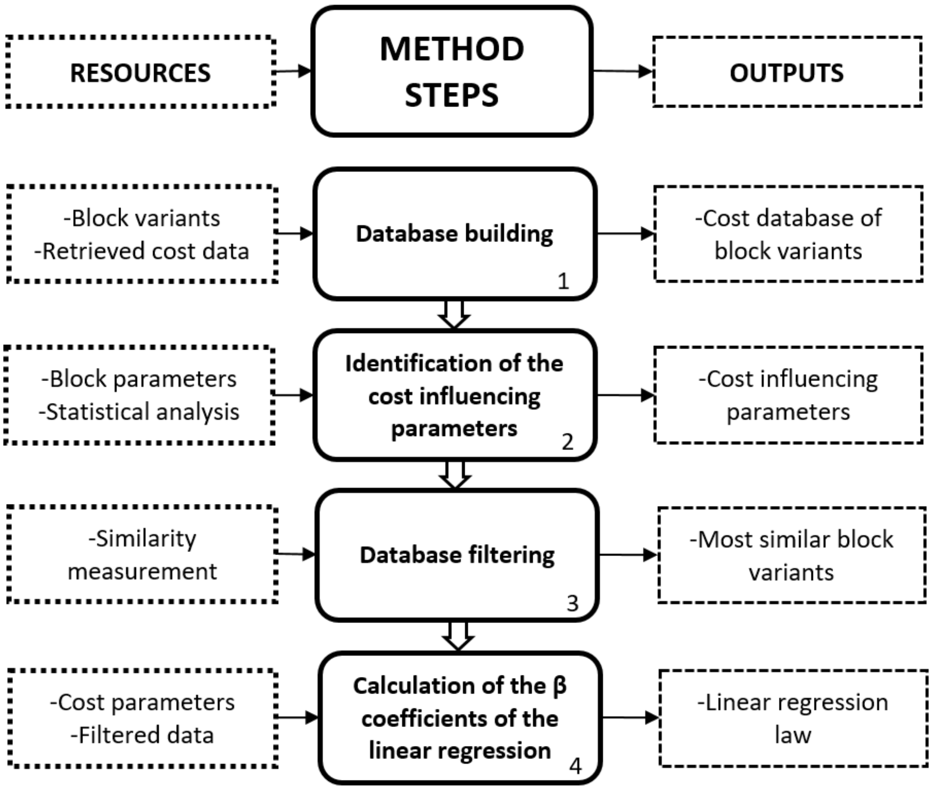

Figure 25 - Method: cost estimation of purchased material

The database building should include all the variants (codes) of the same block type, whose cost attribute is known. The framework allows an automated storage of the codes, which are the block variants assigned in the past projects. Without a sufficient number of variants, it is not possible to estimate the cost of a new part. When past costs are retrieved, it is important to consider inflation rates.

Since the block consists of different attributes, it is important to identify the ones that most affect the block cost, i.e. the influencing parameters. A single attribute is often enough to perform a good estimation, within a certain range. Functional parameters available in a catalog (e.g. the motor power or the gearbox torque) can be often related to the cost instead of geometrical attributes.

Moreover, a linear correlation is preferable, because it often gives more reliable results within a limited range.

Parameters affecting the block cost can be easily identified by means of a correlation measurement between block attributes and cost. The correlation coefficient r is an easy measure of a linear correlation between two variables X and Y. It can assume values between +1 and −1, where 1 is a perfect positive linear correlation, 0 is no linear correlation, and −1 is a perfect negative linear correlation (Pearson, 1895).

$$r = \frac{cov(x,y)}{\sigma_x \sigma_y}$$

where:

- cov is the covariance
- σ_x is the standard deviation of X
- σ_y is the standard deviation of Y

Given the cost Y, the framework suggests to the user the attribute X that maximizes the absolute value of *r*.

If it is expected that two or more attributes affect the block cost, a multiple correlation is needed. Multiple correlation is a measure of how well a given variable can be predicted using a linear function of a set of other variables. It is the correlation between the variable values and the best predictions that can be computed linearly from the predictive variables (Allison, 1998).

Given variables x, y and z, the multiple correlation coefficient is the following:

$$R_{z,xy} = \sqrt{\frac{r_{xz}^2 + r_{yz}^2 - 2r_{xz}r_{yz}r_{xy}}{1 - r_{yz}^2}}$$

where r_{xz}, r_{yz} and r_{xy} are the correlation coefficients between variables pairs.

If x and y are independent variables, meaning that there is no correlation between them, then $r_{xy}=0$ and the previous equation becomes:

$$R_{z,xy} = \sqrt{r_{xz}^2 + r_{xz}^2}$$

In general, for n variables, the square of multiple correlation coefficient can be calculated with the following:

$$R^2 = c^T R_{xx}^{-1} c$$

where c^T is the transpose of c and R_{xx}^{-1} is the inverse of the matrix

$$R_{xx} = \begin{pmatrix} r_{x_1x_1} & \cdots & r_{x_1x_N} \\ \vdots & \ddots & \vdots \\ r_{x_Nx_1} & \cdots & r_{x_Nx_N} \end{pmatrix}$$

and c is the vector of the correlation coefficients $c = \left(r_{x_1y,}\, r_{x_1y}, \dots, r_{x_1y}\right)^T$

If the n variables are all independent among them, R_{xx} becomes the identity matrix and R^2 is then the sum of the square of the correlation coefficients.

Once the cost influencing attributes have been identified, it is possible to use them for filtering the database, in order to retrieve the past codes that are the most similar to the new instantiated block. Similarity measurement can be used to identify the closest block variants and, in addition, some conditions can be imposed to the attributes in order to filter the data.

Given X the vector containing the values of the selected attributes for the cost estimation of the new product and Y the corresponding vector of a code in the database:

$$X = (x_1, x_2, \dots, x_n) \text{ and } Y = (y_1, y_2, \dots, y_n) \in \mathbb{R}^n$$

the distance between the two blocks is the following

$$\left(\sum_{i=1}^{n}|x_i - y_i|^p\right)^{\frac{1}{p}}$$

where p can be chosen as 2 (Euclidean distance).

The last step consists in determining such a cost estimation formula by means of a linear regression:

$$Cost = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p x_P$$

where β_i are the coefficients to be determined and x_i are the independent variables for the regression. x_i can include also non-linear attributes, e.g. x_i^2. The least-square method is used in order to perform the data fitting of the n codes that are filtered in the third step on the attributes chosen in the second step.

As a result, a cost estimation formula has been found for the new block, considering both the block similarity and the attributes influence in the cost estimation.

3.2.3.2 Cost estimation of purchased materials

The cost model for a new part variant to be produced within the company makes use of the cost structure of the past variants (codes) available in the database.

The steps for performing a cost estimation of a produced part are the following (see Figure 26):

1. Building of a database considering the block cost structure
2. Identification of the cost influencing parameters for each cost item
3. Database filtering
4. Identification of a cost-growth law

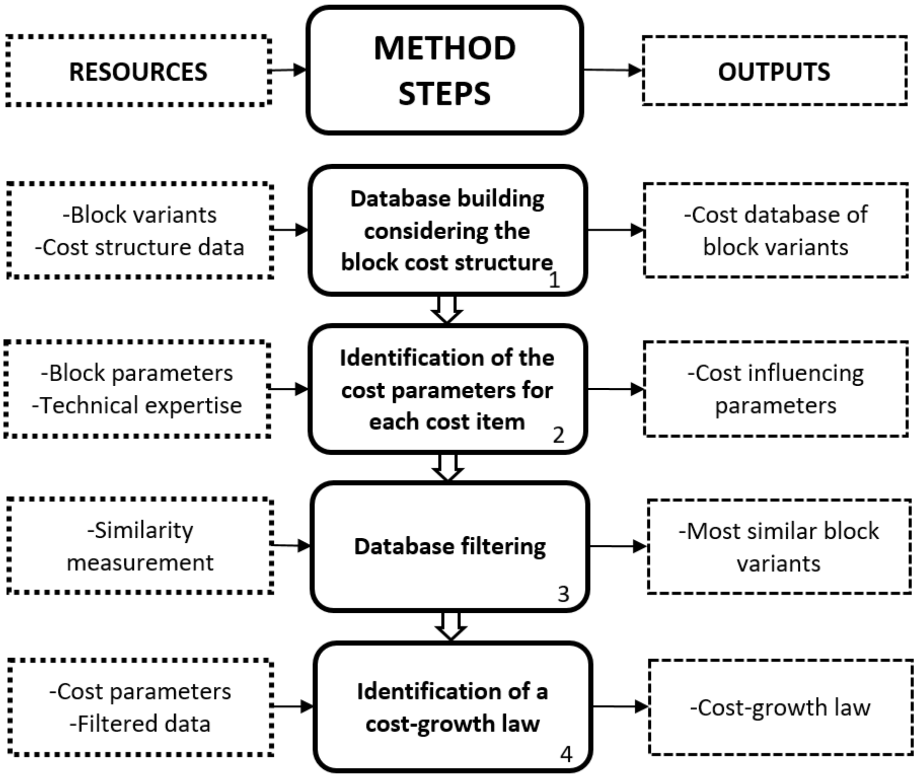

Figure 26 - Method: cost estimation of produced parts

Since even the cost estimation of produced parts needs a cost database, the first step is quite similar to that of the purchased materials. Indeed, it is important to identify the most convenient way to divide the cost into different cost items, e.g. material and manufacturing costs. Thus, the cost database shows the block variants having the total cost divided into different cost items. Cost items should also include all the life-cycle costs, e.g. power consumption, consumables, maintenance and end of life. The life-cycle cost estimation will be deepened in the next section.

Since the cost structure of produced parts is known, it is possible to relate each cost item to one or more block attributes. In section 2.6, a method for estimating costs particularly suitable for product families have been discussed, which is based on part geometric similarity. Although the method allows an easy and rapid cost estimation, it can be used only to estimate the variable part of the manufacturing costs, without considering the life-cycle costs (LCC). The method makes use of similarity relationships of parts to derive cost-growth laws that are based on influencing factors, i.e. the block attributes that most affect the cost. For example, because the material cost increases proportionately to the part volume, the latter can be used as the influencing factor of the material cost. In the same way, manufacturing cost items can be estimated through the identification of the influencing factors that affect the times for the individual operations.

Although a statistical analysis of the data could be used in order to find out correlations between cost items and influencing factors, the choice of these can be guided by the technical expertise of senior designers, probably leading to more reliable cost estimations. Once the influencing factor of each cost item has been identified, proportionality coefficients that are company specific can be derived from known costs data of past projects. The coefficients are then used to estimate the costs of different parts belonging to the same part family.

The use of a single basic design to estimate the costs of a product family (Ehrlenspiel, Kiewert, & Lindemann, 2007) does not consider the variability of costs within a company and extends the validity of a single case in a too wide domain. For these reasons, while the method reported in the state of art makes use of a single basic design in order to estimate the costs of an entire product family, the method proposed in this thesis averages the costs data of the most similar blocks. Similarity measurements, eventually powered by a filter, allow to find out the most similar blocks basing on the cost influencing factors. In this way, because the retrieved data are closer to the block to be estimated, the results are more reliable.

3.2.3.3 Life-cycle cost estimation

As mentioned above, literature background shows early cost estimation methods that are limited to the estimation of the material and manufacturing costs. However, a life-cycle cost (LCC) estimation is important since the early stages, e.g. in the offer stage. For example, when a customer is going to buy a power plant, it is interested in both the product cost (CAPEX), which is the investment, and the running cost (OPEX). The latter has a big importance in the product life-cycle and depends on attributes like the product performance.

The table below shows a general cost structure, comprising the product life-cycle phases from the design to the end of life. For each phase, the most important cost drivers are reported. Cost drivers have been defined as the structural determinants of the cost of an activity, reflecting any linkages or interrelationships that affect it (Porter, 1985). In the column details, contributions examples for each cost driver have been shown.

In most cases, only the cost drivers regarding the first phase are included in an offer.

Table 3 - A general cost structure

Phases	Cost Drivers	Details
Design and Technical Purchases	Purchasing Cost	Design
		Construction
		Commercial components
		Documentation
		Installation
	Spare parts first supplying	
	Training	
	Packaging and Trasport	
Use and Production	Energy Consumption	Electricity Consumption
		Gas Consumption
		Cold water Consumption
		Hot water Consumption
		Steam Consumption
		Compressed air Consumption
	Manpower	
Management	Maintenance	Corrective Maintenance
		Preventive Maintenance
		Predictive Maintenance
		Shut Down Maintenance
		Ameliorative Maintenance
		Maintenance activities in case of extraordinary events
	Scraps	
	Strategic and Specific Spare Parts	
End of Life	Decommissioning	Disposal
		Disassembly

In order to extend the material and manufacturing cost estimation models discussed in the background, cost models usable at early stages are proposed in this thesis to estimate the cost of:

- Product design (mechanical, electrical, automation and software)
- Documentation and quality assurance
- Energy consumption

Design cost models are used for the estimation of the product engineering costs, i.e. mechanical, electrical and automation. In the offer stage of engineer to order products, these costs often suffer of a bad estimation, leading to an underestimation or an overestimation of them. This is due to the difficulty to estimate how much from the past projects is possible to be reused and consequently understanding the engineering effort for the new product. For this reason, a cost model that uses the knowledge base of the company has been introduced. Given a maximum cost (or time) for a design activity C_{max}, which is obtainable from the cost database, the actual cost C_{act} must consider the past project reuse so that:

$$Cact = k * Cmax, \quad 0 \leq k \leq 1$$

where k can be derived from the Euclidean distance above discussed between the block of the new product and a block retrieved from past cases. Attributes should be chosen as the most affecting the design effort.

As regards the documentation and quality assurance costs, the proposed model is similar to the previous, however the attributes for distance measurement must be of a more general nature and inherent to the project general specifications, i.e. standards, certifications and tests.

In order to consider the running costs of different product alternatives, energy consumption cost models are needed. For example, the choice of a more efficient motor should be evaluated basing on the motor duty cycle and the product service life given in the product specifications. In the same way, the use of a variable frequency drive to control the motor speed should be evaluated in terms of LCC.

Cost estimation is mostly made after the conceptual design stage; however, it gives most of the benefits when applied together with the conceptual design because the search for the minimal cost is simpler. In addition, the product layout designer is aware of his/her choices since the early stages, evaluating the impact of his/her choice on the product cost.

The framework alone does not allow a cost optimization, i.e. it is not possible an automated variable assignment in order to obtain a minimal cost. However, a cost reduction can be obtained through a manual assignment of product attributes, knowing the cost-growth laws of the blocks.

A target cost reasoning is also possible, considering the cost as a constraint given in the product specifications. Because the tool continuously checks for consistency during the product configuration, if the estimated product total cost exceeds the target cost, a warning is given.

3.2.4. Summary of the method

The method proposed in this thesis makes use of the knowledge acquired in the past projects in order to configure and estimate the costs of new engineer to order products. The method consists of 3 main parts: (1) the knowledge formalization and representation, (2) the support towards the solution and (3) the cost estimation.

Knowledge formalization and representation lead to the creation of a meta-model to conveniently represent a product family, which preferably has a good degree of modularity. In addition, a knowledge base that contains product variants is created and continually grows. Both the meta-model and the knowledge base have been implemented within a framework in order to be easily integrated with the most common tools used in a company.

The second part of the method shows a semi-automatic tool to support the user towards the solution starting from the product requirements, particularly suitable for taking a priori decisions and for managing variables of different types.

Finally, in the last part of the method, several methodologies for cost estimation have been given to evaluate the product LCC at early stages. The methodologies allow to build parametric cost models by exploiting the knowledge base that has been created in the first part and the product attributes that have been determined in the second part.

The method and the framework proposed in this thesis are easily applicable in the industrial field and allow a reduction in the time needed for early cost estimation, especially during the offer stage.

3.2.5. Difficulties in the practical implementation of the method

The difficulties of the method implementation within a framework to be used in a company are mainly in the first part of the method, i.e. the knowledge formalization and representation. As discussed above, the possibility of connecting data and bringing tools together allows to avoid redundancy and inconsistency of information and to easily retrieve data from the DB. However, several difficulties have been encountered in managing data that come from different tools, e.g. CAD, PLM, ERP, spreadsheets, …. This mainly depends on the company information system, i.e. how data management works within the company. Nowadays, while big enterprises often have advanced data management systems, SMEs normally use cheaper and easier to maintain systems.

In order to overcome these difficulties and make the system usable also for SMEs, the focus was in the integration with spreadsheets, usually used in all companies. Where the integration of the framework with other tools was not possible, spreadsheets have been used as a bridge for the integration with other systems.

Finally, a challenge that nowadays remains open is to convince people, who have to change something in their way of working, of the benefits in using different methods and tools and above all of the importance of properly managing data for a future reuse.

Chapter 4.

Use of the framework in a case study

4.1. Introduction

In order to prove the validity of the method discussed in chapter 3 within the industry, it has been applied to the overhead crane product family, thanks to the collaboration of the company that has co-financed the doctoral scholarship. The system has been tested by the collaboration with designers from a partner company, after a short training about the main system functionalities.

Due to the time needed for applying the method to this product family, it was not possible to extend the use of it in other industrial fields. However, the case study can be considered very significant because it covers almost all the aspects treated in the method. In addition, overhead cranes are engineer to order (ETO) products having a good degree of modularity. Overhead cranes consist of different part types, such as mechanical parts, structural parts and electrical and automation systems. Moreover, some parts are produced within the company while others are purchased.

It has been seen that in the company the offer stage requires much time and effort, thus the method and the related framework have been thought to reduce them.

4.2. Company knowledge formalization and representation

The work began with the recovery of the data available in the company regarding the overhead crane product family. For this specific product family, a set of product information was retrieved, organized and classified, i.e. customer specifications, BOM, design documents, 3D models, 2D drawings, and cost data (Raffaeli, Savoretti, & Germani, 2016). A large number of product variants has been taken into account, in order to make an exhaustive analysis. In particular, about 50 different overhead crane variants have been recovered from the company database, belonging to job orders the company has executed in the last 10 years.

Overhead cranes are complex products since they are made of numerous parts performing several functions. The several specifications and customer requirements make the design always different in some aspects. In order to consider all possible combinations of relations and customer requirements, a general functional model for the overhead crane has been defined, by combining all the customer requirements. In the functional structure the material, energy and signal flows have been reported. Functions have been grouped in order to identify the product modules, which are linked to the physical components, i.e. the blocks of the product architecture. The software Modulor (Raffaeli, Mengoni, & Germani, 2011) has been used to represent the functional and modular structures. In such a way, the company knowledge was ordered and functionally structured, so that it became easier and more efficient to visualize the product architecture.

Product functional analysis starts with the identification of the basic function of the concerned product. This function is like a "Black box" from which all the product sub-functions can be derived. For an overhead crane, the base function is "Move the load" (Figure 27).

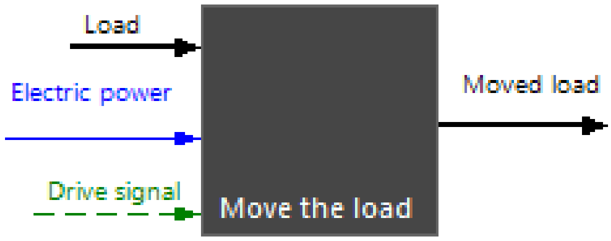

Figure 27 – Black box

Input and output flows, which describe product needs and results, have been identified in order to perform the zero-level function. Then, input and output flows have been developed consistently in the following sub-levels. Load, electric power and drive signal are the needed inputs to obtain the load movement.

In the first level of the functional decomposition, which is shown in Figure 28, four key functions have been identified:

1 **Hoist the load**
2 **Move the load along the x direction**
3 **Move the load along the y direction**
4 **Transmit the signal**

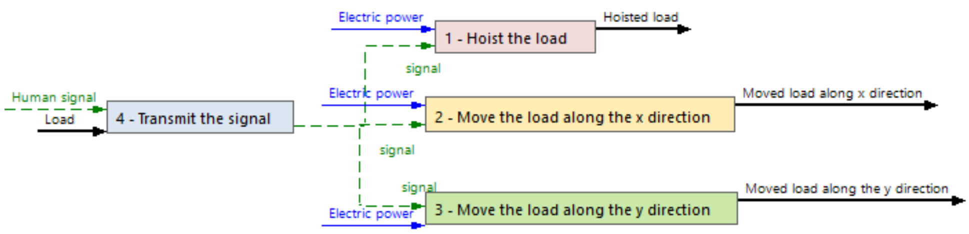

Figure 28 – First level of the functional analysis

1. **Hoist the load** allows moving the load in the vertical direction, by lifting and lowering it according to the human signal.
2. **Move the load along the x direction** allows moving the load along the transverse direction of the plant
3. **Move the load along the y direction** allows moving the load along the longitudinal direction of the plant
4. **Transmit the signal** allows to transmit the human signal to the crane unit control in order to move the load

The case study concerns a detailed analysis of the crane hoist system, identifying the functions and the building blocks and then applying the design and the cost estimation methods above discussed.

In the second functional level, 11 sub-functions of the functional block "Hoist the load" have been identified (see Figure 29).

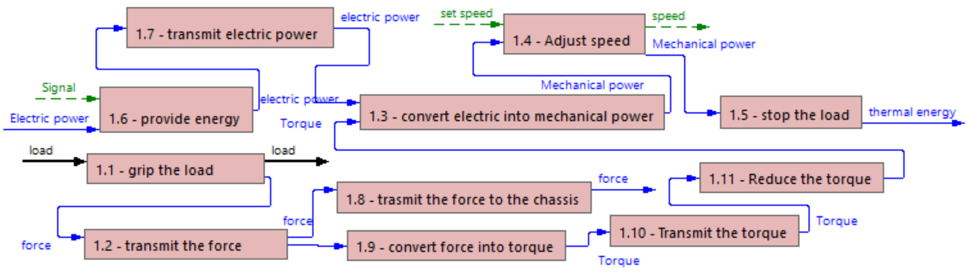

Figure 29 - "Hoist the load" - second level of the functional analysis

Hoist the load has been divided as follows:

1.1 ***Grip the load*** allows a safe handling of the load

1.2 **Transmit the force** allows transmitting the load weight to the parts that are responsible of the hoisting

1.3 **Convert electric into mechanical power** provides mechanical power, i.e. torque and speed, by converting electric power.

1.4 **Adjust speed** allows moving the crane at the correct set speed

1.5 **Stop the load** allows the load safe braking, by dissipating mechanical energy into thermal energy

1.6 **Provide energy** serves to give electric energy to the hoisting parts

1.7 **Transmit electric power** allows the electric energy transmission to the hoisting parts that need it

1.8 **Transmit the force to the chassis** allows the force transmission to the chassis of the crane, which is able to support it

1.9 **Convert force into torque** allows converting the linear into a rotational motion and the force into a torque

1.10 **Transmit the torque** allows the torque transmission along the kinematic chain

1.11 **Reduce the torque** allows converting mechanical energy by reducing torque and increasing speed

As regards the hoist functions, the following building blocks have been identified:

- **Drum** is responsible for converting the force into a torque
- **Drum support** allows the transmission of a part of the force to the trolley chassis
- **Equalizer system** is responsible for transmitting a part of the force to the trolley chassis
- **Upper block** allows both the transmission of the load force to the hoist driving parts and the transmission of a part of the force to the trolley chassis
- **Motor** is responsible for converting the electric power into mechanical power, i.e. torque and speed.

- **Brake** is responsible for dissipating the mechanical energy into thermal energy
- **Gearbox** allows reducing the needed motor torque
- **Drum-Gearbox coupling** is responsible for transmitting the torque from the drum to the gearbox
- **Inverter** allows adjusting the hoist speed at the set speed
- **Rope** is responsible for transmitting the load force to the hoist driving parts
- **Load block** allows a safe grip of the load and transmits the load force to the hoist driving parts

"Provide energy" and "transmit electric power" functions are performed by the "power supply" block which has been placed outside the hoist because it provides and transmits the electric power to all the trolley parts, e.g. the trolley drives.

It is worth pointing out that a block generally consists of several components. For instance, the load block can be subdivided into the hook, the sheaves, the sheaves pin, the structure, etc. Moreover, the functional analysis has been conducted by neglecting several functions, e.g. the load weighing, which disables the load lifting if the load weight exceeds the crane capacity. However, the aim of the crane functional analysis has been to identify the main parts of the crane family, i.e. the product blocks, which have a close connection with the product functionalities.

Figure 30 shows the general product architecture of an overhead crane, where the third level is reported only for the hoist system.

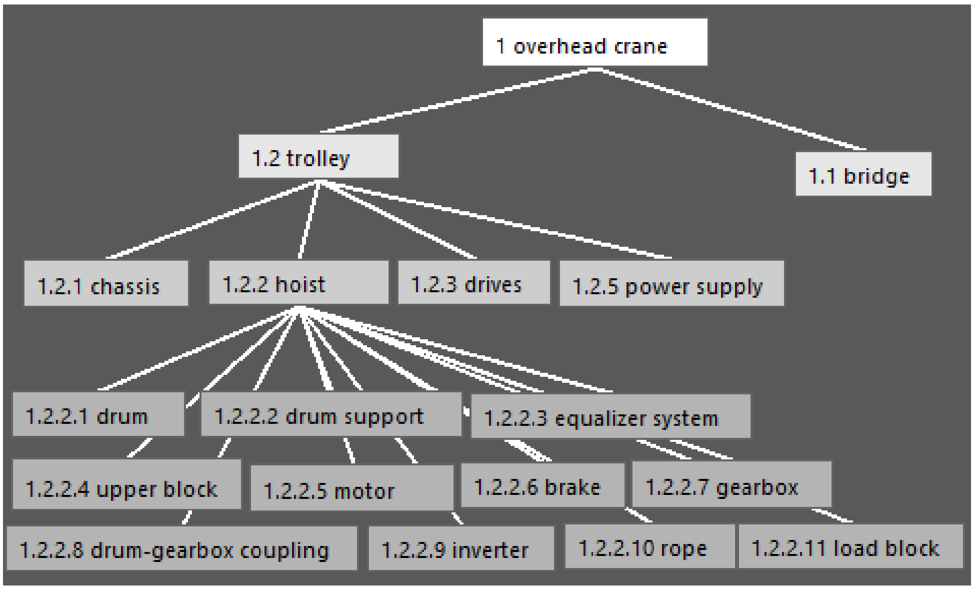

Figure 30 -Crane architecture

For each block above described, the most important attributes have been identified. Spreadsheets for dimensioning activities have been useful for identifying the most important attributes for block instantiation. Moreover, template CAD models have been built for blocks that can be geometrically represented.

Figure 31 shows the instantiation of the drum block, which is an important part of the hoisting unit, by means of the framework.

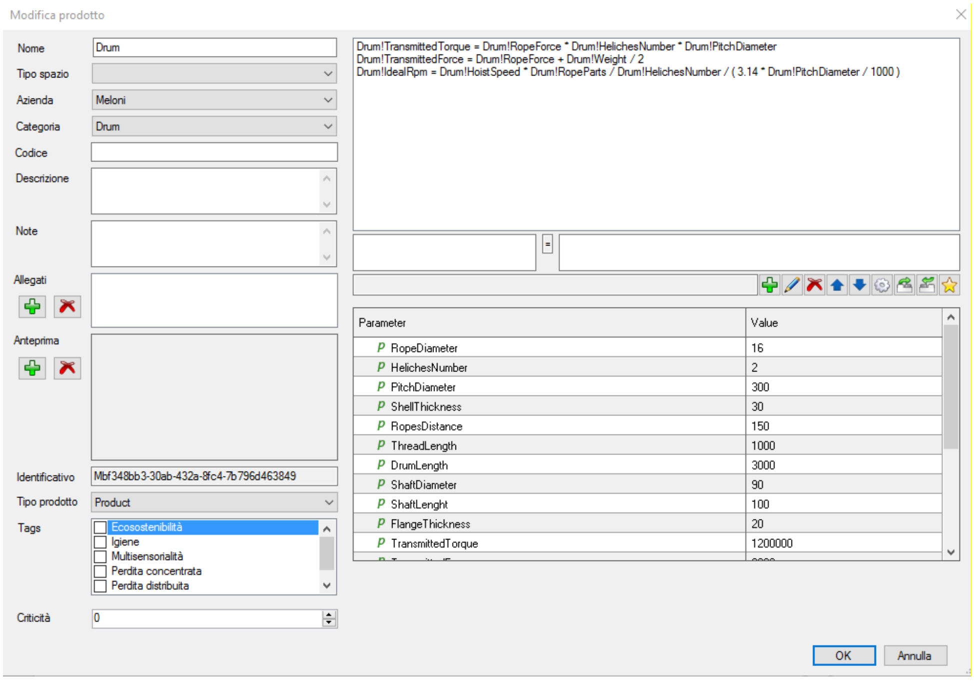

Figure 31 - Instantiation of the drum block

On the left, generic characteristics of the block are reported, i.e. the name, code, category, etc.… At the top right, dependencies between attributes of the block can be established, while at the bottom right there is the list of the block attributes and their assigned values. Block attributes have been conveniently divided into inputs and outputs, where inputs are the required data for the drum instantiation (e.g. rope diameter and force) and outputs are the resulting data from the instantiation (e.g. weight, cost). Rope diameter and force are attributes of a block rope that is connected to the drum. The connection establishes a dependency between the two blocks.

Examples of dependencies of different types are shown below:

$$n_{drum} = \frac{v_{hoist} \cdot n_{ropes}}{n_{threads} \cdot \pi \cdot D_{drum}}$$

$$D_{drum} \geq k \cdot D_{rope}$$

are *math formulas* which connect the following attributes:

- hoist speed v_{hoist} is a product requirement
- parts number n_{ropes} of the rope is a requirement of the hoist unit
- number of threads $n_{threads}$ is a free choice attribute of the drum
- drum diameter D_{drum} is a free choice attribute of the drum
- rope diameter D_{rope} is an attribute of the block rope.

The following *topology rules* have been introduced for the block drum:

- A block drum needs to be connected to at a least one rope
- A block drum needs two lateral connections, at least one must be a gearbox, the other can be a support.

A *structure traversing rule* has been used to count the number of gearboxes attached to the drum, because it affects the drum dimensioning. Structure traversing rules are also used to calculate the total cost of a parent block, by summing the costs of the children blocks.

The *geometric relation* between the drum axis and the gearbox output axis has been set within the CAD environment (see Figure 32).

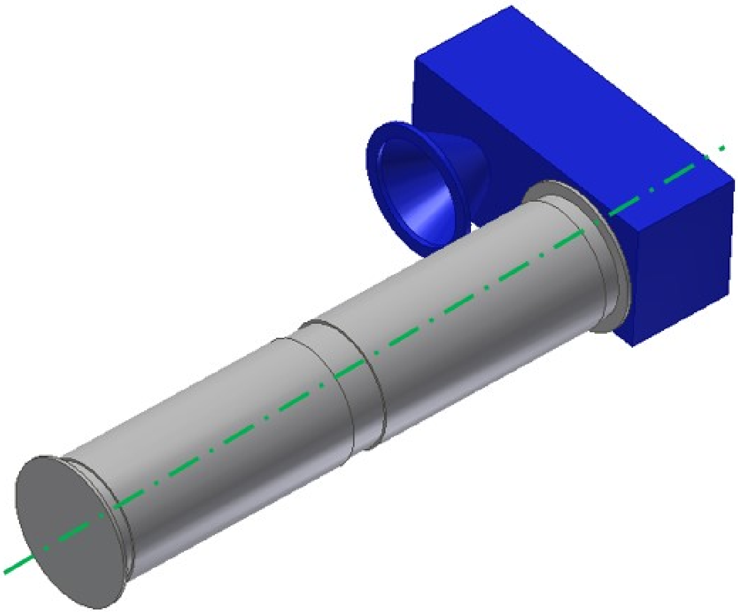

Figure 32 - Geometric relation example

A *spreadsheet relation* between many drum attributes and the spreadsheet model that is used within the company for drum dimensioning has been created (see Figure 33). A sheet input-output is useful to manage the attributes values relation between the sheets and the blocks of the product architecture.

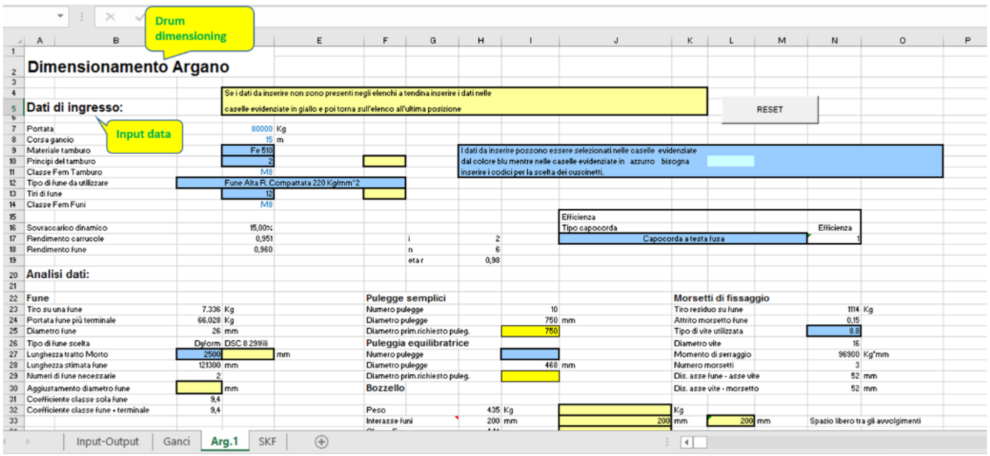

Figure 33 - Spreadsheet used within the company for drum dimensioning

Parametric CAD model templates have been created with the software Autodesk Inventor (Autodesk, 2017) and then linked to the framework by API connections. Simplified geometry of the block drum consists of the following geometric entities: pitch diameter, thickness, length and, eventually, two parameters to identify the number of threads and the distance between them. These geometric entities have been linked to the framework in order to generate and update drum 3D CAD models. Some examples of simplified 3D CAD model are given in Figure 34. The total area of the drum has been retrieved to calculate the painting surface while the minimum distance between drum and motor has been retrieved to check the clearance between the two blocks.

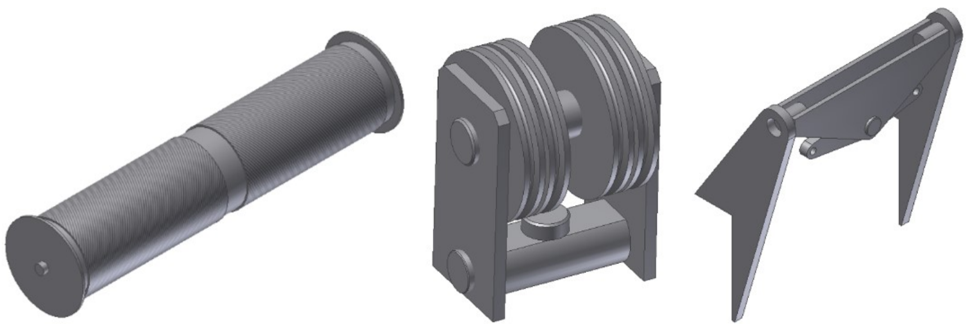

Figure 34 - Simplified CAD models of drum (left), load block (center) and equalizing system (right)

Once blocks and dependencies have been identified, product has been represented through an IDEF as a network of blocks connected through input-output attributes. The IDEF diagram has been expanded down to such a level that all the dimensioning tasks and parameters are shown and a network of dependencies between blocks are evidenced. Figure 35 shows an exemplary portion of the crane design process, i.e. the drum and block dimensioning. It can be seen that the block weight is an output of the block dimensioning, which depends on rope dimensioning, but at the same time it provides an input for the rope dimensioning, thus realizing coupled dependencies to be solved. Coupled dependencies are very common when designing and must be solved in an iterative manner.

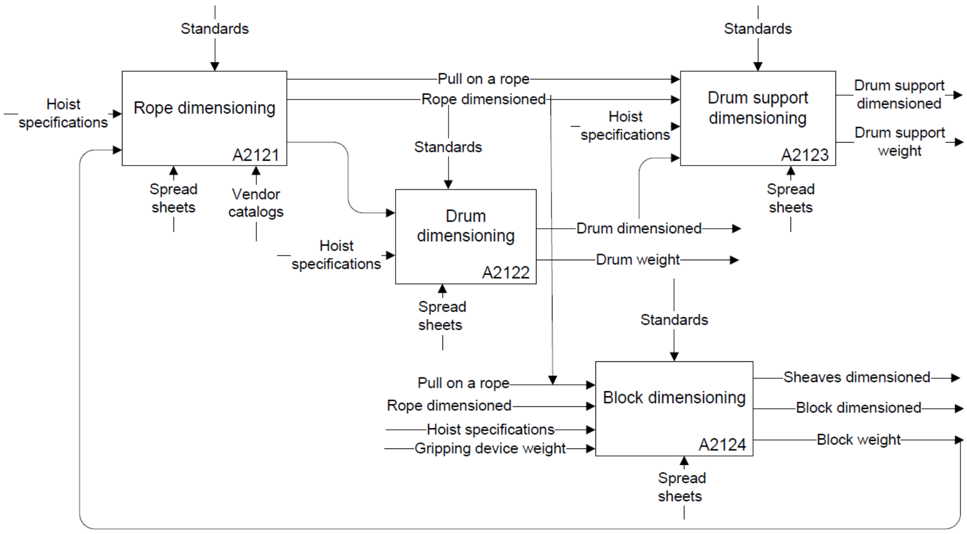

Figure 35 - A screenshot of the activity A212 regarding the drum and block dimensioning

The steps described above have led to the building of a database of about 50 overhead crane variants and to the definition of a meta-model for the overhead crane product family. This collection of data constitutes a starting database for product configuration and cost estimation which continually grows as the company will acquire new job orders. Then, blocks can be retrieved from the database and variants of these blocks can be used to design new products.

A similarity measurement based on the following product specification parameters has been performed in order to retrieve the 10 most similar product variants (see Table 4) from the database:

- Capacity: 80 tons
- Span: 30 m
- Hook lift: 16 m
- Hoist speed: 5 m/min

- Trolley speed: 20 m/min
- Bridge speed: 40 m/min

Table 4 - Most similar retrieved variants

	1	2	3	4	5	6	7	8	9	10
Capacity (tons)	75	80	110	90	90	103	125	115	97	60
Span (m)	40.2	26.8	43.1	29.8	25.1	29.8	19.4	9.5	15	27.2
Hook lift (m)	16	17	11	14	10	14	15	14.5	18	13
Hoist speed (m/min)	2	6	5	8	8	8	6	3	4	5
Trolley speed (m/min)	10	10	20	40	40	40	20	20	30	10
Bridge speed (m/min)	30	16	20	40	40	40	50	60	60	30

The crane blocks (i.e., motor, gearbox, hook, beams, etc…) have added in the software system along with a 3D geometry, identified parameters, dependencies and dimensioning rules. In particular, the requirements above have been assigned to few designers of different expertise. They have input the specifications in the system and started the configuration process by retrieving the required blocks from the system DB. The product architecture of the new crane has been designed starting from the above product variants, which have been adapted to take into account the requirements of the new product. A preliminary 3D layout of the trolley that has been obtained is shown in Figure 36.

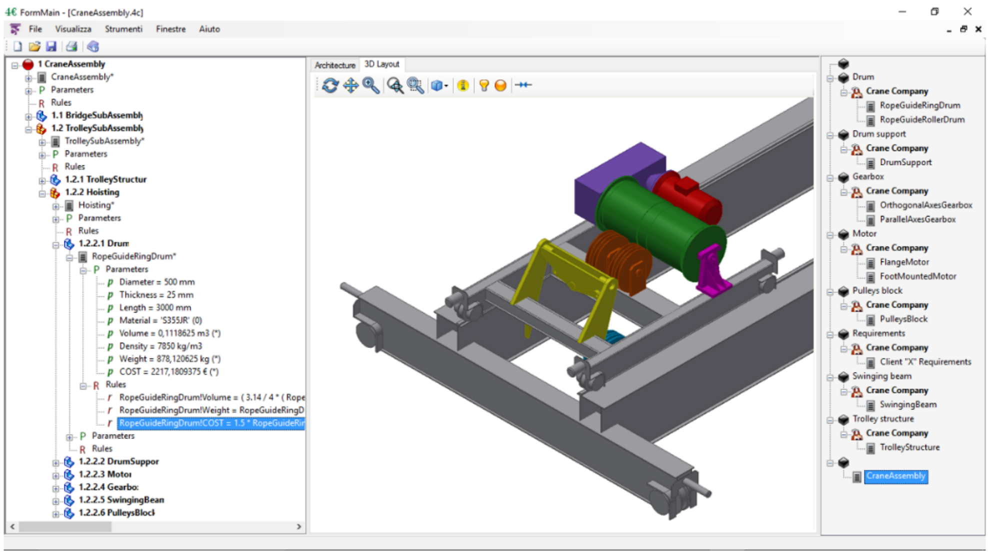

Figure 36 –preliminary 3D layout

Once the crane structure has been defined, dependencies between product attributes have been analyzed and collected by the framework in a parameter-based DSM. The convention used to represent dependencies in this thesis is the IC/FBD convention, i.e. DSM feedback marks with inputs shown in columns and outputs in rows.

In the next section, the step of product configuration is discussed, which consists in the variables assignment and in the layout assessment in order to satisfy the product requirements and the dependencies that have been introduced in the product architecture. In particular, the supporting tools within the framework are applied to solve different design problems, showing the benefits of such a framework.

4.3. Support towards the solution

In this section, different examples are discussed, showing the tool support in the solution search phase. Once the parameter-based DSM has been instantiated according to the dependencies coming from the product architecture of the crane, the matrix has been reordered by means of clustering algorithms, in order to divide the problem into more sub-problems of reduced size. In fact, because of the highly-coupled problem nature, a partitioning of the entire DSM would lead to just one big cluster. The resulting clustered DSM in Figure 37 shows several blocks of highly coupled parameters, but the dependencies among clusters are much lower.

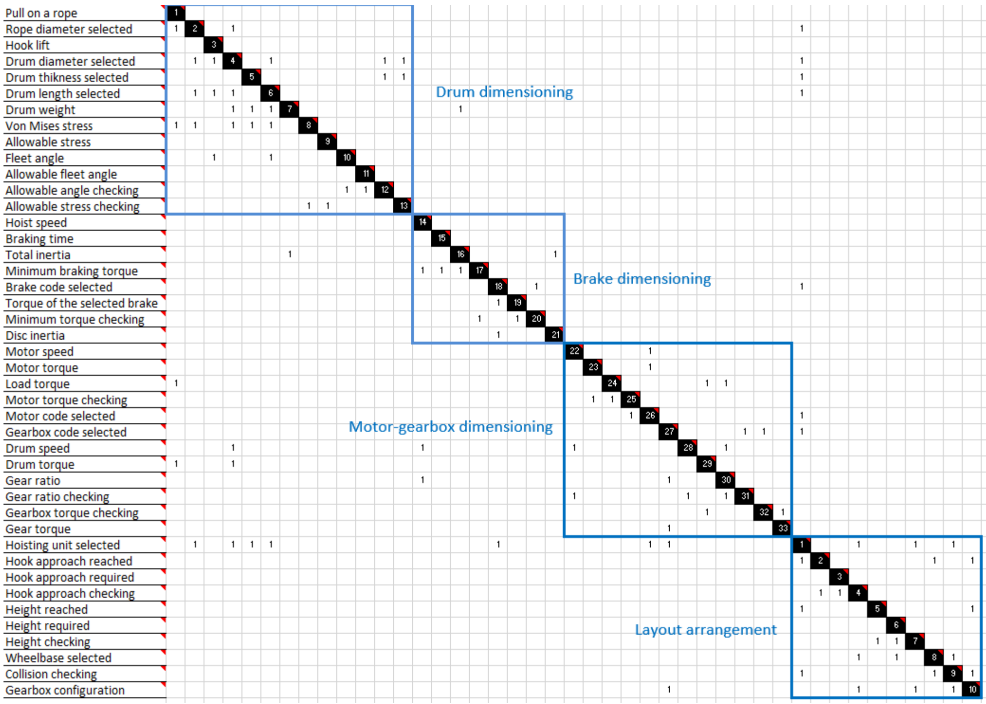

Figure 37 - Clustered DSM

Then, each matrix cluster has been treated as an individual problem and has been solved independently from the others. The dependencies between different clusters have been checked after the resolution of all the clusters. Three exemplary clusters of coupled parameters that have been identified in the DSM are here reported with more details.

The first example regards the resolution of a cluster of 8 parameters that results from the clustered DSM above and refers to the brake dimensioning process. Figure 38 shows a typical layout of a disc brake which is used in the crane hoist system. A brief description of the parameters belonging to this cluster follows:

- *Hoist speed* is the maximum allowed speed for lifting/lowering the load, which is 5 m/min as per product specifications
- *Braking time* is the maximum allowed time for stopping the load which is moving at the hoist speed, i.e. one second as per product specifications
- *Total inertia* is the hoist system inertia that includes the disc inertia [kgm^2]
- *Disc inertia* is the inertia of the selected disc brake [kgm^2]
- *Brake code selected* is the code of the disc brake that has been selected from the supplier catalog
- *Minimum torque checking* is a Boolean parameter (True/False) that verifies the inequality constraint

$$\text{Torque of the selected brake} \geq \text{Minimum braking torque}$$

- *Minimum braking torque* is the minimum torque that allows a safe braking in the required time [Nm]
- *Torque of the selected brake* is the torque of the selected brake [Nm]

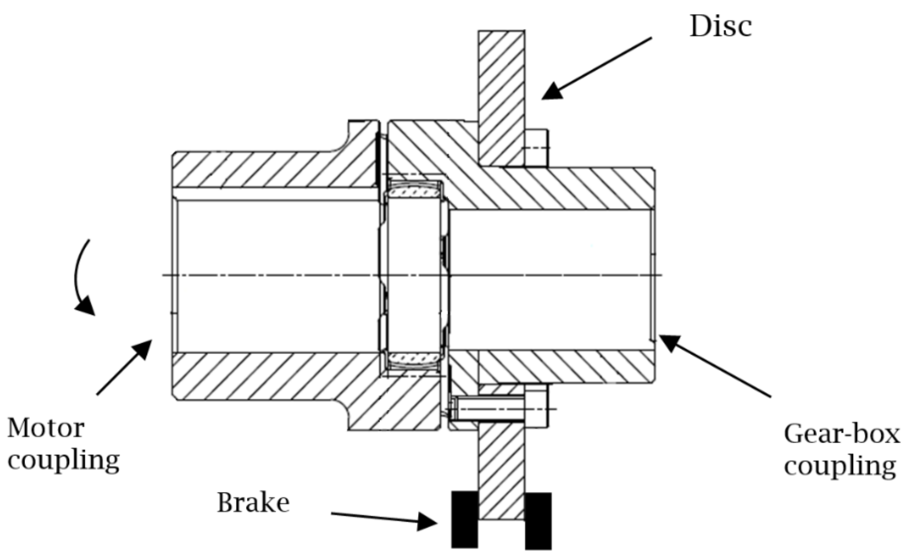

Figure 38 - Disc brake layout

Building a DSM with the 8 parameters above cited and then partitioning it leads to the partitioned DSM shown in Figure 39(a). The ones above the diagonal identify the parameters which are affected by mutual dependencies and need to be solved in an iterative manner by a backtracking approach.

The brake selection is a coupled problem since the determination of the minimum braking torque depends on the disc inertia that will be known only after the selection of the brake. It can be verified that the a priori choice of *brake code selected* allows to determine all the other parameters of the cluster. In fact, removing the dependencies from *brake code selected* leads to the triangularized DSM shown in Figure 39(b).

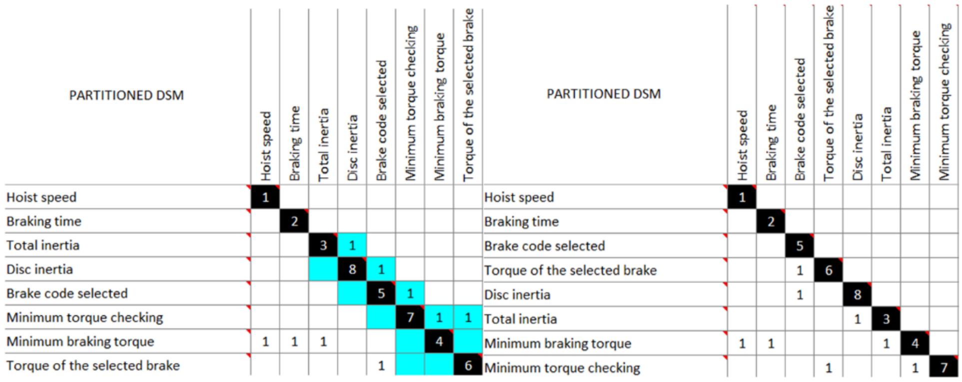

Figure 39 Partitioned DSM of the brake before (a) and after (b) tearing

Assuming to have a brake supplier catalog with the information of Table 5, the brake can be manually or automatically selected.

Table 5 – Brake disc catalogue

Brake disc	Torque [Nm]	Disc inertia [kgm²]	Total inertia [kgm2]	Min braking torque [Nm]	Min torque checking
Ø 200x20	325	0,02	0,17	586	False
Ø 250x20	450	0,06	0,21	592	False
Ø 315x20	525	0,15	0,30	606	False
Ø 400x20	685	0,39	0,54	644	True
Ø 500x30	940	1,44	1,59	809	True
…	…	…			

After removing the dependencies from *brake code selected*, problem resolution proceeds as follows. From an a priori choice of *brake code selected*, it is possible to derive *torque* and *disc inertia* of the selected brake (Table 5). Then, *total inertia* can be calculated by summing the *disc inertia* to the inertia of the other blocks of the hoist kinematic chain, e.g. the load, the drum and the motor. From *hoist speed*, *braking time* and *total inertia,* it is possible to derive *minimum braking torque* that is required for a safe braking.

Finally, the last parameter of the DSM, i.e. *minimum torque checking*, can be determined. After deriving all the parameters, a check of all the dependencies of the original DSM in Figure 39(a) must be done. In particular, the dependency between *brake code selected* and *minimum torque checking* must be restored and checked. According to this dependency, if *minimum torque checking* is true, then *brake code selected* is a valid assignment, otherwise a new assignment for *brake code selected* is required. Because several assignments are valid, a function to decide which is the best choice is needed.

An objective function to optimize the problem solution can be as follows:

$$Min\ (braking\ torque)$$

This condition, together with the dependencies given above, allows to select the brake with the minimum size which enables a safe braking (highlighted in green in Table 5).

The second example regards the resolution of a cluster of 12 coupled parameters resulting from the clustered DSM and is related to the dimensioning of the drum, which is a part of the crane hoist system which is responsible of rope winding (see Figure 40).

The parameters of the block include dimensions, weight and stress of the drum due to the pull on the rope. Moreover, they also comprise relations with other clusters, such as the crane requirements: i.e. the rope diameter must ensure the support of the maximum load, and the drum length must be enough to ensure the hook lift. Checking parameters (True/False values) include the verification of the following inequality constraint:

$$\text{equivalent drum stress} \leq \text{allowable stress}$$

$$\text{fleet angle} \leq \text{allowable fleet angle}$$

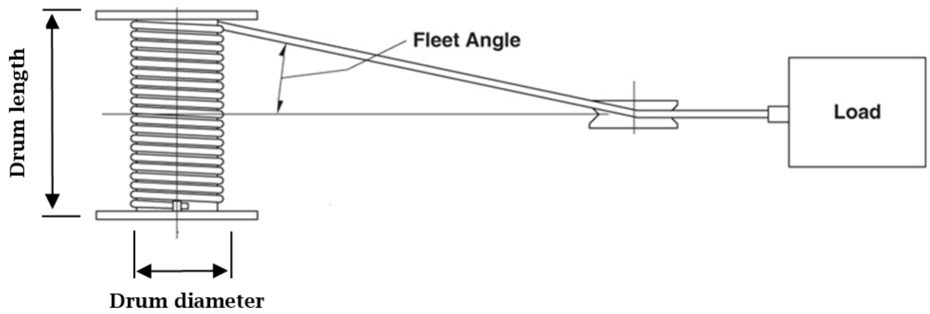

Figure 40 - The hoisting drum

Stresses are calculated according to the applicable standards and *fleet angle* results from a simple trigonometric formula. The partitioning of the coupled block has led to the DSM in Figure 41(a). Although the parameters assignment could be made by a CSP integrated-optimization tool, the need of two a priori choices has been demonstrated. In fact, by removing the dependencies only for one parameter at a time, the condition of triangularization is not yet satisfied. Instead, searching for pairs of parameters, several alternatives have been found. Discarding the pairs containing verification parameters, the solving iteration has given the *Drum Diameter* and the *Drum Thickness* parameters as a possible solution for a priori choices. The Figure 41(b) shows the DSM partitioned after removing dependencies by this specific pair of parameters.

(a)

PARTITIONED DSM	Pull on a rope	Rope diameter	Hook lift	Allowable stress	Fleet angle	Drum length selected	Drum diameter selected	Allowable angle checking	Allowable stress checking	Von Mises stress	Drum thikness selected	Drum weight
Pull on a rope	1											
Rope diameter	1	2										
Hook lift			3									
Allowable stress				9								
Fleet angle			1		10	1						
Drum length selected		1	1			6	1					
Drum diameter selected		1					4	1	1			
Allowable angle checking					1			11				
Allowable stress checking				1					12	1		
Von Mises stress	1	1				1	1			8	1	
Drum thikness selected								1	1		5	
Drum weight						1	1				1	7

(b)

PARTITIONED DSM	Pull on a rope	Rope diameter	Hook lift	Drum diameter selected	Drum thikness selected	Drum length selected	Drum weight	Von Mises stress	Allowable stress	Fleet angle	Allowable angle checking	Allowable stress checking
Pull on a rope	1											
Rope diameter	1	2										
Hook lift			3									
Drum diameter selected				4								
Drum thikness selected					5							
Drum length selected		1	1	1		6						
Drum weight				1	1	1	7					
Von Mises stress	1	1		1	1	1		8				
Allowable stress									9			
Fleet angle			1			1				10		
Allowable angle checking										1	11	
Allowable stress checking								1	1			12

Figure 41 - Partitioned DSM of the hoist drum before (a) and after (b) tearing

In order to solve the problem in a manual manner, i.e. without a CSP solver, a DOE plan has been assigned, which provides the drum diameter and thickness variation in 5 levels. Consequently, the DOE plan consists of $5^2 = 25$ experiments. The company spreadsheet in Figure 33 has been used to calculate the other parameters of the DSM and check the dependencies.

Since it must be

$$Drum\ diameter \geq k * Rope\ diameter$$

where k is a coefficient given in the FEM standards which depends on the hoist FEM class, the minimum drum diameter follows the rope diameter choice. A *rope diameter* of 24 mm descends from the *pull on a rope* value. Because k is 20 according to M6 FEM class, the *drum diameter* must be at least 480 mm. By choosing an arbitrary maximum for the drum diameter of 800 mm and a range for the drum thickness from 30 to 70, the DOE plan, which reports also the results for the other parameters of the cluster, is shown in Table 6. While *hook lift* is 16 m as per customer specifications, *allowable fleet angle* is 2° and *allowable stress* is 110 MPa according to the FEM standards. Although the weight can be retrieved from the CAD environment, the formula that follows gives a good approximation of it:

$$weight = \rho\pi L \frac{D^2 - (D - 2t)^2}{4}$$

where D, t and L are respectively the drum diameter, thickness and length and ρ is the density of the drum material.

Table 6 – DOE plan for drum dimensioning

Nr	Drum diameter	Drum thickness	Drum length	Drum weight	VM stress	Fleet angle	Angle checking	Stress checking
1	480	30	2860	952	197	2,7	False	False
2	480	40	2860	1241	135	2,7	False	False
3	480	50	2860	1516	104	2,7	False	False
4	480	60	2860	1777	84	2,7	False	True
5	480	70	2860	2024	71	2,7	False	True
6	560	30	2530	992	190	2,3	False	False
7	560	40	2530	1298	130	2,3	False	False
8	560	50	2530	1591	99	2,3	False	True
9	560	60	2530	1872	80	2,3	False	True
10	560	70	2530	2140	67	2,3	False	True
11	640	30	2320	1047	186	2,1	False	False
12	640	40	2320	1373	127	2,1	False	False
13	640	50	2320	1688	96	2,1	False	True
14	640	60	2320	1991	78	2,1	False	True
15	640	70	2320	2283	65	2,1	False	True
16	720	30	2100	1072	183	1,8	True	False
17	720	40	2100	1409	125	1,8	True	False
18	720	50	2100	1735	95	1,8	True	True
19	720	60	2100	2051	76	1,8	True	True
20	720	70	2100	2356	64	1,8	True	True
21	800	30	1940	1105	181	1,6	True	False
22	800	40	1940	1454	123	1,6	True	False
23	800	50	1940	1794	94	1,6	True	True
24	800	60	1940	2124	75	1,6	True	True
25	800	70	1940	2445	63	1,6	True	True

Table 6 shows in green the six valid solutions that have been found. Due to the presence of several solutions that satisfy the dependencies of the DSM, an objective function to decide the best alternative is needed. The objective function can be represented as follows:

$$f\ obj\ =\ Min\ (Weight)$$

This condition allows to select one of the best alternatives in terms of cost among the space of solutions, because it was found that the drum cost is approximately dependent from the weight. However, a more detailed cost estimation of the drum is reported in the next section.

According to the objective function, experiment 8 is the best alternative. However, a new DOE plan can be performed for further investigations within the solutions domain that has been found, with the aim to find other valid solutions and eventually a new optimum.

As stated before, many design problems cannot be formulated in the form of simple mathematical formulas and resolved by a CSP solver as shown in the previous example. Sometimes the problem is largely dependent on geometrical constraints and simplified parametric layouts are necessary to represent the problem. For instance, the trolley layout design (Figure 42) includes the arrangement of components, such as the hoist units, the traversing units and the trolley chassis and it depends on constraints given by customer specifications that regard the overall dimensions and the working area reachable by the hook.

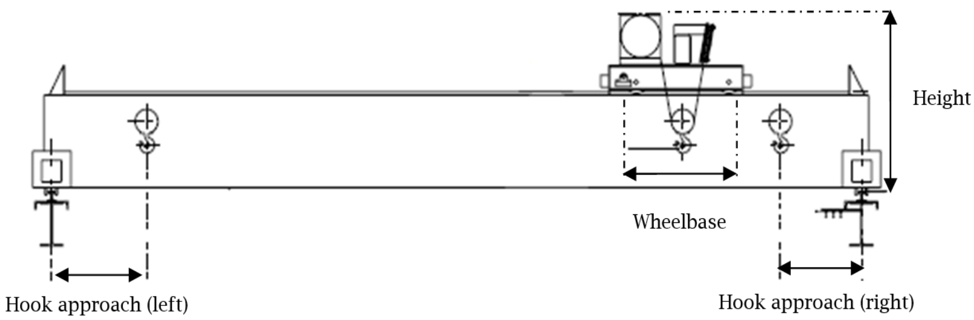

Figure 42 - Typical crane layout

A cluster of 10 coupled parameters has been identified within the clustered DSM of Figure 37, but the partitioned DSM in Figure 43(a) shows that only 8 parameters are affected by mutual dependencies.

Applying the method, it has been found that 3 parameters need assumptions: *hoist unit selected*, *wheelbase selected* and *gearbox configuration*. The first is related to the resolution of the drum dimensioning cluster above discussed, which led to several valid alternatives. *Wheelbase selected* is the distance between the wheel groups and affects the trolley length. Finally, *gearbox configuration* regards the type of arrangement between the hoist motor and the gearbox, i.e. parallel or orthogonal axes. Figure 43(b) shows the partitioned DSM after removing the dependencies from the parameters above, where coupled parameters have been eliminated and the matrix has been triangularized. The choice of the parameters above must comply with two constraints given in the product specifications, i.e. the maximum *hook approach* and *height*. While the former relates to the crane working area, the latter depends on the distance from the top of the crane rails to the roof of the building in order to avoid collisions. These parameters can be checked only after a layout arrangement within the CAD environment, together with the absence of collision between the parts. *Hook approach required* is 1600 mm for both left and right side while *height required* is 2600 mm. Furthermore, if some dimensions arising from the blocks arrangement do not satisfy the constraints, probably the block dimensioning needs a review: this iteration reflects what actually happens during the design process.

PARTITIONED DSM	Hook approach required	Height required	Height checking	Height reached	Hoisting unit selected	Hook approach checking	Hook approach reached	Wheelbase selected	Collision checking	Gearbox configuration
Hook approach required	3									
Height required		6								
Height checking		1	7	1						
Height reached				5	1					1
Hoisting unit selected			1		1	1			1	
Hook approach checking	1					4	1			
Hook approach reached					1		2	1		1
Wheelbase selected		1				1		8	1	
Collision checking					1			1	9	1
Gearbox configuration			1			1			1	10

PARTITIONED DSM	Hoisting unit selected	Hook approach required	Height required	Wheelbase selected	Gearbox configuration	Hook approach reached	Hook approach checking	Height reached	Height checking	Collision checking
Hoisting unit selected	1									
Hook approach required		3								
Height required			6							
Wheelbase selected				8						
Gearbox configuration					10					
Hook approach reached	1			1	1	2				
Hook approach checking		1				1	4			
Height reached	1				1			5		
Height checking			1					1	7	
Collision checking	1			1	1					9

Figure 43 - Partitioned DSM of the trolley layout arrangement before (a) and after (b) tearing

Since the designer expertise plays a determinant role in the layout assessment, the problem cannot be solved by an optimization tool but requires the human intervention. Thus, Table 8 contains the solutions to explore and evaluate the response for each experiment within the CAD environment. Hoist 1 and Hoist 2 derive from 18 and 23 drum dimensioning experiments, which were the best alternatives considering the drum weight optimization. Because no alternative from 1 to 12 meets the requirements above, a new exploration of the drum dimensioning solutions has been performed, which led to the Hoist 3 alternatives from 13 to 18. In particular, the explored solutions space has regarded the drum diameter variation between 640 and 720 mm and the drum thickness variation between 40 and 60 (see Table 7). The optimal valid solution has been highlighted.

Table 7 – Further solution exploration for drum dimensioning

Nr	Drum diameter	Drum thickness	Drum length	Drum weight	VM stress	Fleet angle	Angle checking	Stress checking
1	660	40	2210	1352	126	1,9	True	False
2	660	50	2210	1662	96	1,9	True	True
3	660	60	2210	1962	77	1,9	True	True
4	680	40	2150	1357	125	1,9	True	False
5	680	50	2150	1670	95	1,9	True	True
6	680	60	2150	1972	77	1,9	True	True
7	700	40	2100	1367	125	1,8	True	False
8	700	50	2100	1683	95	1,8	True	True
9	700	60	2100	1989	76	1,8	True	True

Table 8 - DOE plan for trolley layout arrangement

Nr	Hoist unit	Wheelbase	Gearbox Configuration	Hook approach	Height	Hook approach checking	Height checking	Collision checking
1	Hoist 1	2000	parallel	1130	2580	True	True	False
2	Hoist 1	2000	orthogonal	1050	2580	True	True	False
3	Hoist 1	3000	parallel	1630	2580	False	True	True
4	Hoist 1	3000	orthogonal	1550	2580	True	True	False
5	Hoist 1	4000	parallel	2130	2580	False	True	True
6	Hoist 1	4000	orthogonal	2050	2580	False	True	True
7	Hoist 2	2000	parallel	1210	2660	True	False	False
8	Hoist 2	2000	orthogonal	1130	2660	True	False	False
9	Hoist 2	3000	parallel	1710	2660	False	False	False
10	Hoist 2	3000	orthogonal	1630	2660	False	False	False
11	Hoist 2	4000	parallel	2210	2660	False	False	True
12	Hoist 2	4000	orthogonal	2130	2660	False	False	True
13	Hoist 3	2000	parallel	1070	2520	True	True	False
14	Hoist 3	2000	orthogonal	990	2520	True	True	False
15	Hoist 3	3000	parallel	1570	2520	True	True	True
16	Hoist 3	3000	orthogonal	1490	2510	True	True	False
17	Hoist 3	4000	parallel	2070	2520	False	True	True
18	Hoist 3	4000	orthogonal	1990	2520	False	True	True

Table 8 shows that only experiment 15 (highlighted in green) complies with all the given requirements.

Once all the clusters have been resolved, the supporting tool has checked for inconsistencies between parameters of different clusters. If the system detected some inconsistencies, the values of the related parameters have been modified until all the dependencies of the original DSM have been respected. The presented case of study has required a new exploration of the drum solutions space to meet the required hook approach. In particular, drum diameter has been decreased to obtain a trolley layout arrangement that allows the crane to work within the required area.

In order to solve the design problem, parameters have been varied according to the rules in the sequence suggested by the system. Thus, different designers have been guided to complete a preliminary design (Figure 44). Surprisingly, the final solutions were very similar and cost-effective, confirming the capability of the approach of supporting also novices.

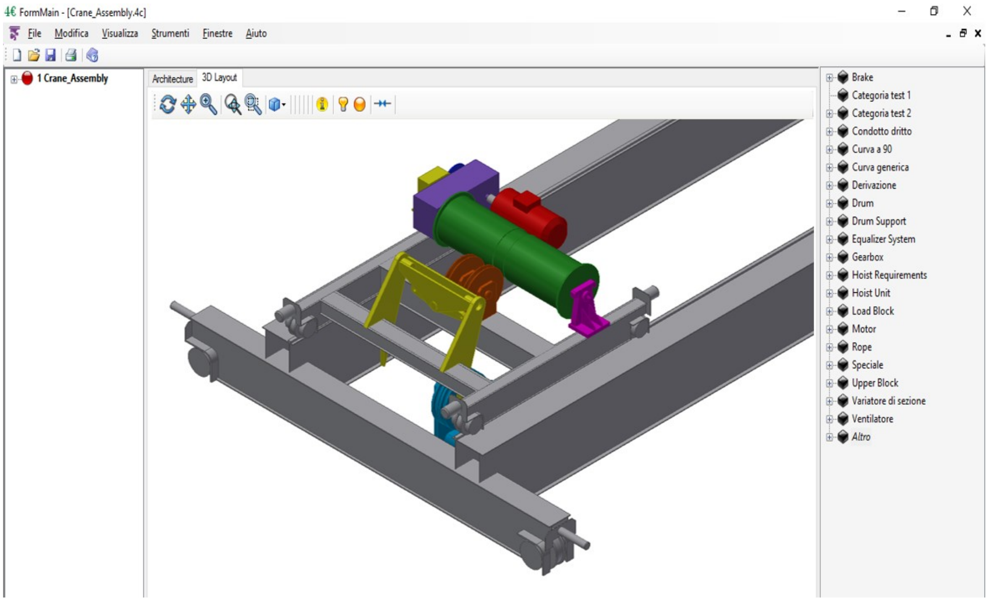

Figure 44 - Simplified 3D layout result of the configuration process

4.4. Early cost estimation

The method and the tools regarding the cost estimation discussed in the previous chapter have been applied to the hoisting unit of a crane. Four examples regarding two purchased parts and two produced parts follow.

As discussed above, cost estimation of purchased material is mostly used to know the costs of some parts like motors, gearboxes, couplings, whose costs are often not included within the catalogs. For instance, although squirrel cage electrical motors are almost standard parts, many suppliers do not show the cost in their catalogs. In order to avoid waste of time by asking for offers of these purchased parts during the cost estimation stage for a new product, a rapid cost estimation is required.

Apart from the purchased cost of the motor, it is important to evaluate the running costs of it. In fact, since an electrical motor involves a power consumption, efficiency parameters and the possible introduction of a variable frequency driver (VFD) should be considered. For these reasons, the motor cost model includes both the regression model for purchased cost and the running costs due to the power consumption.

From the motor and gearbox dimensioning cluster above, the motor code reported in Table 9 has been selected because it has the minimum power that allows a safe hoist according to the product specifications.

Table 9 - Motor selected code

Attribute	**Value**	**Unit**
Motor type	B3	B3/B5
Size	250	mm
Shaft diameter	65	mm
Shaft length	140	mm
Total height	580	mm
Total length	905	mm
Brake motor	N	Y/N
Power	55	kW
Nominal RPM	1475	rpm
Inertia	0,55	Kgm^2
Nominal torque	356	Nm
Weight	315	kg

In order to identify the cost influencing parameters, a correlation measure has been performed for both size and power, resulting in the following coefficients of correlation.

$r_{size} = 0,93$

$r_{power} = 0, 98$

Although both size and power have a good relationship with the motor cost, it results $r_{power} > r_{size}$, consequently the power has been chosen as the parameter for the regression formula.

Using the similarity measurement for power and size and filtering the 4-pole motors of IE2 efficiency class, the 5 closest motor variants have been retrieved from the database and are shown in Table 10.

The discounted costs have been calculated considering an average inflation rate of 3% per year. The inflation rate can be easily estimated if the same motor code has been purchased several times in the past. Some incongruences in the motors cost database may result from supplier discount rates due to commercial negotiations.

Table 10 - 4-pole squirrel cage motor closest variants cost data

Purchased year	**Power [kW]**	**Size [mm]**	**Cost [€]**	**Discounted cost [€]**
2011	30	200	1250	1493
2013	37	180	1780	2003
2013	45	225	2230	2510
2014	90	280	3150	3442
2015	132	315	5500	5835

Considering P the motor power, the purchased cost can be estimated with the following regression formula as a linear function of P (see Figure 45):

$$Cost = 38,8\,P + 468$$

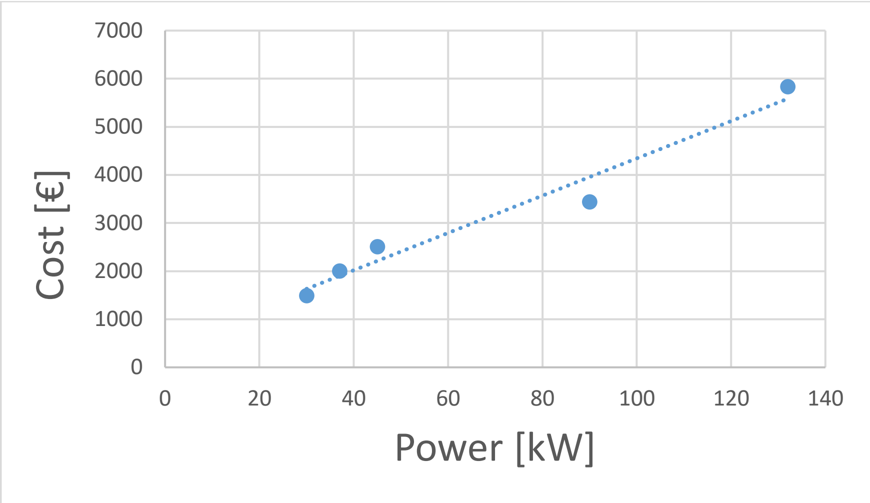

Figure 45 - Regression of motors cost data

Thus, the resulting estimated purchased cost for the motor above of 55 kW is 2602 €, which is actually very close to the prices that have been found in the offers of some suppliers. Since the price of purchased parts is strongly dependent on the relation with the supplier, which could affect the discount rates, it is recommended to filter the database by dividing into suppliers or supplier classes in order to avoid big mistakes in the estimated costs.

In order to estimate the power consumption of a motor, its service data are needed. Table 11 reports the service data of the hoist unit given in the product specification.

Table 11 - Hoist unit service data

Attribute	Value	Unit
FEM class duration T	25000	Hours
FEM class of use L	L1	
Duty cycle	40	%

FEM standard for mechanisms considers both duration and use spectrum. In addition, duty cycle is the ratio between the time the motor is power on and the total time of a cycle.

IEC/EN 60034-30-1:2014 standard considers four efficiency classes for AC electrical motors, i.e. IE1, IE2, IE3 and IE4, by fixing a minimum value for the motor efficiency to be respected.

Table 12 shows the minimum efficiency values for the 4-pole squirrel-cage motor of 55 kW.

Table 12 – Minimum efficiency values of 4-pole 55 kW motors

Efficiency class	Efficiency value	Unit
IE1	92,1	%
IE2	93,5	%
IE3	94,6	%
IE4	95,7	%

It is worth pointing out that the above efficiency values consider the power consumption when the load is in nominal condition. When the load is lower, power consumption does not decrease proportionally, thus the efficiency decreases with the load. However, the use of a variable frequency driver (VFD) allows to keep almost constant the efficiency value. For these reasons, the use of a VFD is to take into account together with the efficiency value.

IE4 class has been neglected in this analysis because it has been introduced few years ago and a full range of IE4 motors is not available for many suppliers. Table 13 shows the efficiency values and the power consumption for IE1, IE2 and IE3 motors with and without VFD.

Table 13 - Efficiency and power consumption of different design alternatives

IE1	IE1 nominal efficiency	93	%
	IE1 efficiency without VFD	65,1	%
	IE1 power consumption with VFD	369624	kWh
	IE1 power consumption without VFD	528034	kWh
IE2	IE2 nominal efficiency	93,5	%
	IE2 efficiency without VFD	65,5	%
	IE2 power consumption with VFD	367647	kWh
	IE2 power consumption without VFD	525210	kWh
IE3	IE3 nominal efficiency	94,6	%
	IE3 efficiency without VFD	66,2	%
	IE3 power consumption with VFD	363372	kWh
	IE3 power consumption without VFD	519103	kWh

From Table 13, the efficiency decrease and the power consumption increase without the use of a VFD are evident.

The cost of the VFD has been estimated using the power attribute as the parameter for the linear regression. Motor costs for IE1 and IE3 efficiency classes have been estimated by filtering the database for the corresponding motor class. Then, by considering the electricity unit cost for the company, it is possible to evaluate the total power consumption costs. By summing all these costs, it has been possible to decide the best alternative. It has been found that the cheapest alternative is to use a IE2 motor together with a VFD.

The same method has been applied to all the purchased components of the hoisting unit, i.e. the gearbox, the brake, etc. For instance, a multiple linear regression has been used for the gearbox, considering the output torque and the number of stages as the independent variables.

Table 14 shows the most important attributes of the gearbox code which has been selected.

Table 14 – Gearbox selected code

Attribute	Value	Unit
Configuration type	Parallel	Parallel/Orthogonal
Size	355	mm
Axes distance	865	mm
Input shaft diameter	50	mm
Output shaft diameter	180	mm
Transmission ratio	166	
Number of reduction stages	4	
Output torque	86800	Nm

Table 15 reports the 5 closest variants that have been retrieved from the gearbox database by considering the number of reduction stages and the output torque as the parameters for the similarity measure. Then, a filter has been applied in order to consider only the gearboxes with a parallel-axes configuration. In this case, discounted costs have been estimated considering an average inflation rate of 2,5 % per year.

Table 15 - Gearbox closest variants cost data

Purchased year	Reduction stages	Output torque [Nm]	Cost [€]	Discounted cost [€]
2014	3	29900	5500	5923
2013	4	42700	8200	9051
2009	4	63200	9500	11575
2011	3	80000	17000	19715
2010	4	119000	19000	22585

Given N the number of reduction stages and T the output torque, the purchased cost of the gearbox can be estimated with the following regression formula as a linear function of N and T:

$$\text{Cost} = 9128 - 2588\,N + 0{,}21\,T$$

It is worth pointing out that a cost model requiring a multi-variable regression should consider a greater number of variants.

It is shown below the application of the method discussed in the previous chapter for estimating the cost of a part produced within the company, i.e. the drum. In the previous section, it has been seen the drum dimensioning process, which has led to the following variable assignment compatible with the product dependencies.

- Diameter = 660 mm
- Thickness =50 mm
- Length = 2210 mm

The cost estimation of the drum has followed the dimensioning process of it. Since the drum is a produced part, the cost structure has been considered. The following cost structure includes material, manufacturing, design and documentation costs. The parameters that most influence the costs below are the geometric attributes, i.e. the diameter D, the thickness t and the length L. Basing on the expertise of company senior designers and on the data retrieved, the following relations have been introduced in the cost model.

- $Material\ cost \propto volume \propto D * t * L$
- $Sheets\ cutting \propto D^2$
- $Lathe\ machining \propto D * L$
- $Welding \propto D * t$

- $Assembly \propto D * L$
- $Sandblasting\ and\ painting \propto surface \propto D * L$
- *Design cost*

The method for the estimation of the design cost will be discussed later. The distance has been calculated as follows:

$$distance = \sqrt{(1 - ø_D)^2 + (1 - ø_t)^2 + (1 - ø_L)^2}$$

where $ø_D$, $ø_t$ and $ø_L$ are the ratios between the attribute value of the block whose costs must be estimated and the corresponding attribute value of the retrieved block. The cost data for the retrieved blocks which minimize the distance values are shown in Table 16.

Table 16 – Drum closest variants retrieved

Drum code	**709152**	**712367**	**711491**	**709268**	**710390**
Diameter [mm]	710	710	710	900	710
Thickness [mm]	50	30	30	30	25
Length [mm]	3600	2900	3660	4800	2600
Material cost [€]	9000	7000	8000	12000	4500
Sheets cutting [h]	6	6	6	2	4
Lathe machining [h]	12	12	12	12	4
Welding [h]	48	16	16	24	24
Assembly [h]	16	6	6	16	12
Sandblasting and painting [h]	5	4	4	9	5
Cost [€]	11536	8210	9210	13752	5912
$ø_D$	0,93	0,93	0,93	0,73	0,92
$ø_t$	1	1,67	1,67	1,67	2
$ø_L$	0,61	0,76	0,60	0,46	0,85
Distance	0,39	0,71	0,78	0,90	1,01

From this data, it has been possible to obtain the cost structure of the new drum.

- $Material\ cost = \frac{1}{n}\sum_{i=1}^{n}(Material\ cost * \text{ø}_D * \text{ø}_t * \text{ø}_L)_i = 6990\ €$
- $Sheets\ cutting = \frac{1}{n}\sum_{i=1}^{n}(Sheets\ cutting * \text{ø}_D{}^2)_i = 4\ h$
- $Lathe\ machining = \frac{1}{n}\sum_{i=1}^{n}(Lathe\ machining * \text{ø}_D * \text{ø}_L)_i = 6\ h$
- $Welding = \frac{1}{n}\sum_{i=1}^{n}(Welding * \text{ø}_D * \text{ø}_t)_i = 34\ h$
- $Assembly = \frac{1}{n}\sum_{i=1}^{n}(Assembly * \text{ø}_D * \text{ø}_L)_i = 6\ h$
- $Sandblasting\ and\ painting =$
  $\frac{1}{n}\sum_{i=1}^{n}(Sandblasting\ and\ painting * \text{ø}_D * \text{ø}_L)_i = 3\ h$

Manufacturing costs have been obtained by multiplying the above estimated hours by the labor cost unit of the corresponding company cost center.

In order to estimate the drum design cost, the empirical method that has been proposed in the previous chapter has been applied. Since the actual effort in designing a new product consists in creating something new, the method is based on the estimation of the part of the job that is retrievable from the past projects. By retrieving the design cost C_{max} of a similar drum from past projects, i.e. the maximum needed effort to design a new drum variant, the actual cost can be estimated as follows:

$$Cact = k * Cmax, \quad 0 \leq k \leq 1$$

where k has been calculated from the distance d above as follows:

$$k = \frac{d}{d+1}$$

When $d = 0 \Rightarrow k = 0$ and it is possible to recover the same block from a past project without any additional effort. On the contrary, when $d \rightarrow \infty \Rightarrow k \rightarrow 1$ and the designer effort is as much as designing a new block.
Considering the drum code 712012, from the distance above of 0,49 it results k=0,33; then it is possible to reduce the effort of about 2/3 by recovering that drum code from the past projects.
The cost estimation of the other crane hoist blocks produced within the company followed the same method. It is here presented the *load block* case because it has required more effort with respect to the cost estimation of the other crane hoist blocks. The load block is responsible for handling the load, generally by a hook and for distributing the load force, through a pulley system, to the other hoist blocks that are fixed in the crane trolley, i.e. the drum, the upper block and the equalizer system.
The load block dimensioning has led to the attribute assignment reported in Table 17 which considers only the cost influencing parameters.

Table 17 – Load block selected

Attribute	Symbol	Value	Unit
Hook diameter	D_h	190	mm
Sheaves diameter	D_s	550	mm
Sheaves width	W_s	80	mm
Sheaves number	N	4	
Pin diameter	D_p	220	mm
Structure thickness	T_s	35	mm

The hook is actually a standard part the company purchases. The hook number derives from the crane capacity, the hoist FEM class and the material resistance class. While the capacity and the hoist FEM class are given in the product specifications and are respectively 80 tons and M6, the hook material is a free choice. By choosing P as the material resistance class, it descends a hook number of 63. Although the hook is a purchased part, it needs some mechanical machining whose cost must be estimated. The hook diameter, which is a geometrical parameter related to the chosen hook, has been selected to estimate the purchased cost of the hook and the required labor cost.

The expertise of company senior designers and an analysis of the past project data have led to the following relations between the block cost structure and its attributes:

- $Material\ cost\ \propto\ volume$
- $Hook\ purchased\ cost\ \propto\ D_h^3$
- $Hook\ machining \propto D_h$
- $Sheets\ cutting \propto D_s^2$
- $Lathe\ machining\ \propto\ D_s * W_s$
- $Welding\ \propto\ D_p * T_s$
- $Assembly \propto\ D_s$
- $Sandblasting\ and\ painting\ \propto\ surface$
- $Design\ cost$

where the load block volume and surface values are retrieved from the CAD environment.

In order to retrieve the 5 closest variants, the load blocks with the same sheaves number have been filtered and the other attributes of Table 17 have been used for a similarity measurement.

Table 18 – Load block closest variants retrieved

Load block code	712143	712385	715466	708238	710791
Hook diameter [mm]	212	170	190	150	170
Sheaves diameter [mm]	600	550	450	500	500
Sheaves width [mm]	90	70	100	80	80
Pin diameter [mm]	230	200	200	180	150
Structure thickness [mm]	40	30	30	35	25
Volume [m³]	0,25	0,151	0,173	0,134	0,135
Surface [m²]	2,8	2,2	1,8	1,9	1,7
Material cost [€]	4875	2950	3375	2625	2650
Hook purchased cost [€]	6100	3800	4500	3350	3650
Hook machining [h]	12	9	10	8	10
Sheets cutting [h]	8	7	5	7	6
Lathe machining [h]	51	35	48	41	43
Welding [h]	19	12	10	12	6
Assembly [h]	36	31	24	29	24
Sandblasting and painting [h]	13	11	8	9	7
Cost [€]	14863	9653	10879	8980	9074
$Ø_{Dh}$	0,90	1,12	1,00	1,27	1,12
$Ø_{Ds}$	0,92	1,00	1,22	1,10	1,10
$Ø_{Ws}$	0,89	1,14	0,80	1,00	1,00
$Ø_{Dp}$	0,96	1,10	1,10	1,22	1,47
$Ø_{Ts}$	0,88	1,17	1,17	1,00	1,40
$Ø_{Volume}$	1,09	1,25	1,39	1,41	0,76
$Ø_{Surface}$	1,32	1,08	1,4	1,25	0,85
Distance	0,22	0,27	0,36	0,36	0,63

From the above retrieved data, the cost structure of the new load block follows.

- $Material\ cost = \frac{1}{n}\sum_{i=1}^{n}(Material\ cost * ø_{Volume})_i = 1476\ €$

- $Hook\ purchased\ cost = \frac{1}{n}\sum_{i=1}^{n}(Material\ cost * ø_{Dh}^{3})_i = \ 5220\ €$
- $Hook\ machining = \frac{1}{n}\sum_{i=1}^{n}(Material\ cost * ø_{Dh})_i = 10\ h$
- $Sheets\ cutting = \frac{1}{n}\sum_{i=1}^{n}(Sheets\ cutting * ø_{Ds}^{2})_i = 7\ h$
- $Lathe\ machining = \frac{1}{n}\sum_{i=1}^{n}(Lathe\ machining * ø_{Ds} * ø_{Ws})_i = 44\ h$
- $Welding = \frac{1}{n}\sum_{i=1}^{n}(Welding * ø_{Dp} * ø_{Ts})_i = 14\ h$
- $Assembly = \frac{1}{n}\sum_{i=1}^{n}(Assembly * ø_{Ds})_i = 30\ h$
- $Sandblasting\ and\ painting =$
  $\frac{1}{n}\sum_{i=1}^{n}\left(Sandblasting\ and\ painting * ø_{Surface}\right)_i = 11\ h$

From the distance of the closest variant, it descends a design effort of about 20% of the needed effort for designing a completely different load block variant. It is worth pointing that the hook cost estimation could follow the method for the purchased blocks, leading to very similar results. In this case, the hook has been considered a part of the load block and thus it becomes an item of its cost structure.

It has been seen that although the individual cost items have led to some discrepancies, by comparing the total estimated cost of each block of the hoist system with the actual cost incurred by the company in this project, a difference lower than 10 % has been found for all the hoist blocks.

Chapter 5.

Concluding remarks

5.1. Conclusions

The resulting benefits of using knowledge-based methodologies are well known and the research in this topic is continuously progressing. Different approaches are presented in literature, such as modularity, knowledge-based engineering, configuration, design automation and computational design synthesis. However, the use of this methodologies within the industrial world is often limited, probably due to the high investment costs.

In an engineer-to-order context, offer stage has a very critical role because the cost estimation must be performed on products that have still to be designed. From these motivations, the research work has been developed by applying knowledge-based tools to the context above. Through an extensive review of the state of art and an analysis of the industrial context, some methodologies and a framework that support the offer stage for new products have been developed. The methodologies and the framework have been thought to be easily usable at industry level.

The methodology begins with the company knowledge formalization and representation from the available data within the company, which can be provided by different sources. The retrieving of the information, which is supported by a strong integration with the company tools, leads to the definition of a product family meta-model and to the building of a product family database. This preparatory phase, which is also the most onerous, is necessary for the implementation of the system within the company and to proceed with the conceptual design and the cost estimation of new products. The results have demonstrated that such a methodology allows a better management of the company data and it makes them reusable for the future jobs.

In order to give a support to the user in the definition of a new product, a methodology has been presented. The results have shown the benefits of the method in managing complex design problem. In particular, it has been seen that the method and the tools presented above allow the user to face sub-problems of reduced size and help him/her in taking a priori choices. The aim of this methodology is to support the user during the conceptual design stage towards a solution of the design problem. The methods for cost estimation are then applied to the solution that has been previously found.

The cost estimation methods proposed in this thesis allow a rapid parameter-based estimation, descending from the attributes that have been calculated in the conceptual design stage. In addition, the proposed methods extend the cost estimation to the product life-cycle. The application of the methods has shown a good estimation of the actual cost both for purchased parts and for parts produced within the company.

The preliminary results have shown that the methodologies give a valid support to the offer stage of engineer to order products such as overhead cranes. Although the methodologies have been well defined in detail and experimented, the implementation in the framework is still ongoing. In addition, the system needs to be improved with a simpler user interface. The experimentation of the framework should be extended to other product families, in order to validate it in different case studies and eventually to discover different company needs.

Future research can be oriented to include solver tools for variable assignment within the framework that allow to find design solutions in an automatic manner. Moreover, the use of computational design synthesis tools could support the exploration of new product architectures.

References

Allison, P. D. (1998). *Multiple Regression: A Primer.* London: Sage Publications.

Antonsson, E. K., & Cagan, J. (2005). *Formal Engineering Design Synthesis.* Cambridge, UK: Cambridge University Press.

Autodesk. (2017). *Inventor.* Retrieved from https://www.autodesk.it/products/inventor/overview

Bohm, M. R., Vucovich, J. P., & Stone, R. B. (2008). Using a Design Repository to Drive Concept Generation. *Journal of Computing and Information Science in Engineering, 8(1)*, 014502.

Boothroyd, G., Dewhurst, P., & Knight, W. (1994). *Product Design for Manufacture and Assembly.* New York: Marcel Dekker.

Browning, T. (2001). Applying the Design Structure Matrix to System Decompositon and Integration Problems: A Review and New Directions. *IEEE Transactions on Engineering Management, 48(3)*, pp. 292-306.

Cagan, J., Campbell, M. I., Finger, S., & Tomiyama, T. (2005). A Framework for Computational Design Synthesis: Model and Applications. *Journal of Computing and Computing and Information Science in Engineering, 5 (3)*, 171-181.

Chakrabarti, A., & Bligh, T. P. (1996). An approach to functional synthesis of mechanical design concepts: theory, applications, and emerging research issues. *Artificial Intelligence for Engineering, Design, Analysis and Manufacturing, 10 (04)*, 313-331.

Chakrabarti, A., Shea, K., Stone, R., Cagan, J., Campbell, M., Hernandez, N. V., & Wood, K. L. (2011). Computer-Based Design Synthesis Research: An Overview. *Journal of Computing and Information Science in Engineering, 11(2)*, 021003.

Chandrasegaran, S. K., Ramani, K., Sriram, R. D., Horváth, I., & Bernard, A. (2013). The evolution, challenges, and future of knowledge representation in product design systems. *Computer-Aided Design, 45 (2)*, 204–228.

Cicconi, P., Germani, M., Bondi, S., Zuliani, A., & Cagnacci, E. (2016). A design methodology to support the optimization of steel structures. *Procedia CIRP, 50*, 58-64.

Coad, P., & Yourdon, E. (1991). *Object Oriented Analysis – Second edition.* Englewood Cliffs, New Jersey, USA: Prentice Hall, A Division of Simon & Schuster.

Duverlie, P., & Castelain, J. (1999). Cost estimation during design step: Parametric method versus case based reasoning method. *International Journal of Advanced Manufacturing Technology, 15 (12)*, 895–906.

Ehrlenspiel, K., Kiewert, A., & Lindemann, U. (2007). *Cost-Efficient Design.* ASME Press.

Eppinger, S. D., Whitney, D. E., Smith, R. P., & Gebala, D. A. (1991). Organizing the tasks in complex design projects. *In: Sriram D., Logcher R., Fukuda S. (eds) Computer-Aided Cooperative Product Development. WCACPD 1989. Lecture Notes in Computer Science* (pp. 229-252). Berlin: Springer.

Ericsson, A., & Erixon, G. (1999). *Controlling design variants: Modular product platforms.* New York: ASME Press.

Farrel, R. S., & Simpson, T. W. (2003). Product platform design to improve commonality in custom product. *Journal of Intelligent Manufacturing, 14*, 541-556.

Felferning, A. (2007). Standardized configuration knowledge representations as technological foundation for mass customization. *IEEE Transaction on Engineering Management, 54(1)*, 41-56.

Felferning, A., Hotz, L., Bagley, C., & Tiihonen, J. (2014). *Knowledge-Based Configuration: from research to business cases.* Morgan Kaufmann.

Frank, G., Entner, D., Prante, T., Khachatouri, V., & Schwarz, M. (2014). Towards a generic framework of engineering design automation for creating complex CAD models. *International Journal on Advances in Systems and Measurements, 7 (1,2)*, 179-192.

Holtta, K., & Salonen, M. (2003). Comparing three modularity methods. *Proc. of ASME Design Engineering Technical Conferences.* September 2-6, Chicago, IL.

Hölttä-Otto, K. (2005). Modular product platform design - Doctoral Dissertation. Helsinki University of Technology, Laboratory of Machine Design.

Kolodner, J. L. (1993). *Case-Based Reasoning.* California: Morgan Kaufmann.

L.S. Wierda, L. (1991). Linking Design, Process Planning and Cost Information by Feature-based Modelling. *Journal of Engineering Design ,2 (1)*, 3-19.

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

Mannisto, T., Peltonen, H., & Sulonen, R. (1996). View to Product Configuration Knowledge Modelling and Evolution. *Configuration Papers from the 1996 Fall Symp., Tech. Report FS-96-03* (pp. 111-118). Menlo Park, Calif.: AAAI Press.

McDermott, J. (1982). R1: a rule-based configurer of computer systems. *Artificial Intelligence, 19 (1)*, 39-88.

McMahon, C., Lowe, A., & Culley, S. (2004). Knowledge management in engineering design: personalization and codification. *Journal of Engineering Design, 15*, 307–325.

Moeller, J., Andersen, H., & Hulgaard, H. (2001). Product configuration over the internet. *6th INFORMS Conference on Information Systems and Technology.* Miami Beach, Florida.

Montgomery, D. C. (2008). *Design and analysis of experiments.* John Wiley & Sons,.

Mullineux, G., Hicks, B., & Medland, T. (2005). Constraint-aided product design. *Acta Polytechnica, 45 (3)*, 31-36.

Münzer, C. H. (2014). *Constraint-Based Methods for Automated Computational Design Synthesis of Solution Spaces, Ph.D. Thesis.* Technische Universität München, Zurich.

Negnevitsky, M. (2005). *Artificial Intelligence – A Guide to Intelligent System.* Harlow, England: Addison-Wesley.

Norris, G. A. (2001). Integrating Life Cycle Cost Analysis and LCA. *The International Journal of Life Cycle Assessment, Jg. 6, H. 2,*, 118–120.

Orsvärn, K., & Axling, T. (1999). The Tacton view of configuration tasks and engines. *AAAI'99 Workshop on Configuration*, (pp. 127-130). Orlando, Florida.

Otto, K., Hölttä-Otto, K., Simpson, T. W., Krauser, D., Ripperda, S., & Moon, S. K. (2016). Global views on Modular Design Research: Linking Alternative Methods to Support Modular Product Family Concept Development. *Journal of Mechanical Design, 138 (7)*, 071101.

Owen, R., & Horvát, I. (2002). Towards product-related knowledge asset ware housing in enterprises. *Proceeding of the 4th international symposium on tools and methods of competitive engineering*, (pp. 155-70).

Pahl, G., Beitz, W., Feldhusen, J., & Grote, K. H. (2007). *Engineering Design: a systematic approach.* Springer.

Pearson, K. (1895). Notes on regression and inheritance in the case of two parents. *Proceedings of the Royal Society of London, 58*, (pp. 240–242). London.

Pimmler, T., & Eppinger, S. (1994). Integration analysis of product decompositions. *ASME Design Theory and Methodology Conference.* September, Minneapolis, MN.

Porter, M. (1985). *Competitive Advantage.* New York: NY: Free Press.

Raffaeli, R., Mengoni, M., & Germani, M. (2011). An early-stage tool to evaluate the product redesign impact. *Proceedings of the ASME 2011 International Design Engineering Technical Conferences, DETC2011*, (pp. DTM-47625). Washington, DC, USA.

Raffaeli, R., Savoretti, A., & Germani, M. (2016). Design knowledge formalization to shorten the time to generate offers for Engineer To Order products. *Advances on MechanicsDesign Engineering and Manufacturing: Proceedings of the International Joint Conference on Mechanics, Design Engineering & Advanced Manufacturing (JCM 2016)* (pp. 1107-1114). Catania: Springer.

Ramesh, M., Yip-Hoi, D., & Dutta, D. (2001). Feature-based Shape Similarity Measurement for Retrieval of Mechanical Parts. *Journal of Computing and Information Science in Engineering, 1 (3)*, 245-256.

Sabin, D., & Weigel, R. (1998). Product configuration frameworks - a survey. *IEEE Intelligent Systems, 13 (4)*, 42-49.

Savoretti, A., Mandolini, M., Raffaeli, R., & Germani, M. (2017). Analysis of the requirements of an early life-cycle cost estimation tool: an industrial survey. *Proceedings of 27th International Conference on Flexible Automation and Intelligent Manufacturing, FAIM2017.* Modena: Elsevier.

Shea, K., Cagan, J., & Fenves, S. J. (1997). A shape annealing approach to optimal truss design with dynamic grouping of members. *Journal of Mechanical Design, 119*, 388-394.

Soloway, E., Bachant, J., & Jensen, K. (1987). Assessing the maintainability of XCON-in-RIME:copying with the problem of very large rule-bases. *Prooceedings of the Sixth National Conference on Artificial Intelligence (AAAI-87)*, (pp. 824-829). Seattle, Washington.

Steinbichler, G. (2008). *Methods and procedures for optimizing the development of parts for injection-molding production processes PhD Thesis.* University Erlangen-Nuernberg.

Stokes, M. (2001). *Managing Engineering Knowledge–MOKA: Methodology for Knowledge Based Engineering Applications.* London: Professional Engineering Publishing Limited.

Stone, R., Wood, K., & Crawford, R. (2000). A heuristic method for identifying modules for product architectures. *Design Studies, 21*(1), 5-31.

Thebeau, R. E. (2001). *Knowledge management of system interface and interactions for product development processes.* PhD Thesis, Massachusetts Institute of Technology.

Ulrich, K. T., & Eppinger, S. (2016). *Product design and Development.* McGraw-Hill.

Ulrich, K., & Eppinger, S. (2012). *Product Design and Development (Fifth Edition).* New York: McGraw-Hill.

Ulrich, K., & Tung, K. (1991). Fundamentals of Product Modularity. *Proc. of ASME Winter Annual Meeting Symposium on Design and Manufacturing Integration.*, (pp. 73-79). Atlanta, GA.

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

Wang , L., Shen, W., Xie , H., Neelamkavil , J., & Pardasani , A. (2002). Collaborative conceptual design - state of art and future trends. *Computer-Aided Design,34(13)*, 981-996.

Warfield, J. N. (1973). Binary matrices in system modeling, 3(5). *IEEE Transactions on Systems, Man and Cybernetics*, 441-449.

Wyatt, D. F., Wynn, D. C., Jarrett, J. P., & Clarkson, P. J. (2012). Supporting product architecture design using computational design synthesis with network structure constraints. *Research in Engineering Design, 23(1)*, 17-52.