# Chapter 4

# LEAP product and manufacturing design support system

Daniele Cerri[1], Sergio Terzi[1]

[1]Department of Management, Economics and Industrial Engineering, Politecnico di Milano, Piazza Leonardo da Vinci 32, 20133, Milano
daniele.cerri@polimi.it, sergio.terzi@polimi.it

**Abstract.** The aim of this chapter is to show the development of the so called "LEAP model for design support". The introduction highlights the importance to pursue a product life cycle approach during the design phase, in order to gain economic and environmental impact savings. The methodologies to pursue this aim, Life Cycle Costing (LCC) and Life Cycle Assessment (LCA) are then presented, defining them and presenting some lacks. Furthermore, Life Cycle Optimization is also presented. In the light of the gaps identified by the state of the art, the chapter presents the Life Cycle Optimization model, highlighting the development and the implementation, and a first application in the industrial context, referred to the COMAU use case. Finally, conclusion summarizes results and possible extensions of the model.

**Keywords:** Life Cycle Costing, Life Cycle Assessment, Life Cycle Optimization, Multi-Objective, Genetic Algorithm.

## 4.1 Introduction

In the modern world, companies are increasingly needed to consider product life cycle. Companies from advanced countries have to face the low-cost pressure of emerging countries, with whom they cannot compete in terms of labor costs. Policy makers (e.g., the European Commission, the UN, etc.) are also pushing for environmental life cycle considerations. In fact, in the last few years, a huge number of laws and directives have been launched. Finally, industrial companies are changing their way of thinking. They want personalized solutions, with the lower total cost of ownership possible.

In this context, manufacturing companies have to identify the best life cycle–oriented solution, in order to satisfy customers' requests and survive in the global market. In fact, being able to develop eco-friendly, energy-efficient and green products before others could give a competitive advantage in the years ahead. Designers and systems engineers are the most involved actors in life cycle consideration: in fact, in the early design phases, about two-thirds of total life cycle costs are fixed [3].

Therefore, they are the most responsible for the improvement of product life cycle.

Two methodologies are well known in literature and can support the evaluation of the costs and environmental impacts generated along the whole life cycle: they are LCC (Life Cycle Costing) and LCA (Life Cycle Assessment).

Life Cycle Costing considers "cradle-to-grave" costs summarized as an economic model of evaluating alternatives for equipment and projects. Engineering details drive LCC cost numbers for the economic calculations. The economics of alternatives drives the scenario selection process. Good engineering proposals alternatives without economic justification are often uneconomical; good engineering with good economics provide business successes. Therefore, the LCC economic model provides better assessment of long-term cost effectiveness of projects [2].

Other LCC definitions are:

- LCC is the total cost of ownership of machinery and equipment, including its cost of acquisition, operation, maintenance, conversion, and/or decommission [10];
- LCC are summations of cost estimates from inception to disposal for both equipment and projects as determined by an analytical study and estimate of total costs experienced in annual time increments during the project life with consideration for the time value of money. The objective of LCC analysis is to choose the most cost effective approach from a series of alternatives to achieve the lowest long-term cost of ownership. LCC is an economic model over the project life span. Usually the cost of operation, maintenance, and disposal costs exceed all other first costs many times over. The best balance among cost elements is achieved when the total LCC is minimized [6].

Life Cycle Costing helps to change perspective on the mere acquisition cost with an emphasis on enhancing economic competitiveness by working for the lowest long term cost of ownership, which is not an easy answer to obtain. Consider these typical problems and conflicts observed in most companies:

1. Project Engineering wants to minimize capital costs as the only criteria;
2. Maintenance Engineering wants to minimize repair hours and costs as the only criteria;
3. Production wants to maximize uptime hours as the only criteria;
4. Reliability Engineering wants to avoid failures as the only criteria;
5. Accounting wants to maximize project net present value as the only criteria;
6. Shareholders want to increase stockholder wealth as the only criteria.

Management is responsible for harmonizing these potential conflicts under the banner of operating for the lowest long term cost of ownership. LCC can be used as a management decision tool for harmonizing the never ending conflicts by focusing on facts, money, and time [2].

Life Cycle Assessment, instead, is a technique to assess environmental impacts associated with all the stages of a product's life from-cradle-to-grave. This concept considers the entire life cycle of a product [4]. “Cradle-to-grave” begins with the gathering of raw materials from the earth to create the product and ends at the point when all materials are returned to the earth. LCA evaluates all stages of a product’s life from the perspective that they are interdependent, meaning that one operation leads to the next. LCA enables the estimation of the cumulative environmental impacts resulting from all stages in the product life cycle, often including impacts not considered in more traditional analysis (e.g. raw material extraction, material transportation, etc.). By including the impacts throughout the product life cycle, LCA provides a comprehensive view of the environmental aspects of the product or process and a more accurate picture of the true environmental trade-offs in product and process selection. The term “life cycle” refers to the major activities in the course of the product’s life-span from its manufacture, use, and maintenance, to its final disposal, including the raw material acquisition that is required to manufacture the product.

Specifically, LCA is a technique to assess the environmental aspects and potential impacts associated with a product, process, or service, by:

- Compiling an inventory of relevant energy and material inputs and environmental releases
- Evaluating the potential environmental impacts associated with identified inputs and releases

- Interpreting the results to help decision-makers make a more informed decision [11].

LCA can help decision-makers to select the product or process that is in the least impact to the environment. This information can be used with other factors, such as cost and performance data to select a product or process. LCA data identifies the transfer of environmental impacts from one media to another and/or from one life cycle stage to another. If an LCA were not performed, the transfer might not be recognized and properly included in the analysis because it is outside of the typical scope or focus of product selection processes.

This ability to track and document shifts in environmental impacts can help decision makers and managers fully characterize the environmental trade-offs associated with product or process alternatives.

By performing an LCA, analysts can:

- Develop a systematic evaluation of the environmental consequences associated with a given product.
- Analyze the environmental trade-offs associated with one or more specific products/processes to help gain stakeholder (state, community, etc.) acceptance for a planned action.
- Quantify environmental releases to air, water, and land in relation to each life cycle stage and/or major contributing process.
- Assist in identifying significant shifts in environmental impacts between life cycle stages and environmental media.
- Assess the human and ecological effects of material consumption and environmental releases to the local community, region, and world.
- Compare the health and ecological impacts between two or more rival products/processes or identify the impacts of a specific product or process.
- Identify impacts to one or more specific environmental areas of concern.

However, most of the available researches are not able to guarantee reaching the optimal solution; rather, in most cases, LCC and LCA are used only for simple evaluations.

The aim of this chapter is to show the development of the so called "LEAP product and manufacturing design support system", able to support designers in the creation and identification of the optimal life cycle oriented solutions, in terms of life cycle costs and environmental impacts.

Firstly, in this Chapter Life Cycle Costing and Life Cycle Assessment methodologies are presented. Furthermore, a section is dedicated to the so called "Life Cycle Optimization".

In the next section the development of the LEAP model for the design support. Finally, the model is applied on an industrial case provided by COMAU. Last section concludes the chapter.

### 4.1.1 Product and Manufacturing Life Cycle Optimization

Analyzing the literature, only few papers have been identified in the optimization of costs and/or environmental impacts along the product life cycle. This kind of optimization is called “Life Cycle Optimization”, because it uses optimization methods with life cycle methodologies (LCC and LCA).

An exploratory analysis of the literature has been conducted, analyzing 39 papers referring to Life Cycle Costing and 40 papers referring to Life Cycle Assessment. Analyzing the results, only a few papers deal with the optimization issue. The percentage is about 20% for LCC literature, which is reduced by half in LCA literature.

The optimization methods used in the literature are: Linear Programming, Genetic Algorithms and Particle Swarm Optimization.

In LCC literature, the most used method is genetic algorithm (about 60%), while in LCA literature the most used is linear programming (about 50%).

As it is possible to see, no contribution optimizes both cost and environmental impact in the whole product life cycle. Furthermore, life cycle optimization is applied just on some sectors. One of the sectors not covered is the industrial system sector, which is instead considered within LinkedDesign project.

## 4.2 Life Cycle Optimization in the LEAP

In this paragraph a model to optimize product life-cycle costs and environmental impacts together, called Product Life Cycle Optimization (PLCO), is introduced. PLCO model is firstly developed with genetic algorithms and compared to linear programming. Finally, it is implemented using Java frameworks.

### 4.2.1 Evaluation of optimization models and algorithms

**4.2.1.1 Linear programming models**

Two linear programming models are used: the “so called” Weighted Sum Model (WSM) and the transformation of multi-objective problem into a single-objective problem, with an objective moved under the constraints, called Bi-Mono Model.

The weighted sum model (WSM) is the best known and simplest Multi-Criteria Decision Analysis (MCDA)/multi-criteria decision making method for evaluating a number of alternatives in terms of a number of decision criteria. It is very important to state here that it is applicable only when all the data are expressed in exactly the same unit. To obviate this we firstly run a single objective problem, maximizing in a run life cycle costs, and in the other one life cycle environmental impacts. Therefore we obtain the maximum values of life cycle costs, identified by LCC*, and life cycle environmental impacts, identified by LCA*. Then we run WSM, dividing the obtained life cycle costs and life cycle environmental impacts values with LCC* and LCA*, in order to have a sum of the two ratios.

In general, suppose that a given MCDA problem is defined on m alternatives and n decision criteria. Furthermore, let us assume that all the criteria are benefit criteria, that is, the higher the values are, the better it is. Next suppose that $w_j$ denotes the relative weight of importance of the criterion $C_j$ and $a_{ij}$ is the performance value of alternative Ai when it is evaluated in terms of criterion $C_j$. Then, the total (i.e., when all the criteria are considered simultaneously) importance of alternative $A_i$, denoted as

$A_i^{WSM\text{-}score}$ [12], is defined as follows (eq. 1):

$$A_i^{WSM\text{-}score}=\sum_j w_j \cdot a_{ij} \text{ for } i = 1, 2, 3, \ldots, m \qquad [1]$$

In Bi-Mono Model, firstly we perform a single-objective problem, minimizing and maximizing life cycle costs. The minimum life cycle costs value is identified by L, the maximum one is identified by H. Then, we run the other objective, referred to life cycle environmental impacts, while life cycle costs is used as constraint, varying its value between L and H, according to a specified step. If the objective in the constraints was to maximize, it will be a constraint of greater or equal, otherwise it will be a constraint of lower or equal.

#### 4.2.1.2 Genetic Algorithms

Many types of multi-objective genetic algorithms exist in literature. We decided to use Non Dominated Sorting Genetic Algorithm 2 (NSGA-2).

NSGA-2 is one of the most popular multi objective optimization algorithms with three special characteristics: fast non-dominated sorting approach, fast crowded distance estimation procedure and simple crowded comparison operator [5].

Deb et al. simulated several test problems from previous study using NSGA-II optimization techniques and it is claimed that this technique outperformed PAES and SPEA in terms of finding a diverse set of solutions.

NSGA-2 has been demonstrated as one of the most efficient algorithms for multi-objective optimization on a number of benchmark problems [7].

The complete procedure of NSGA-2 is given below to demonstrate an implementation of elitism without using a secondary external population.

The procedure of NSGA-2 is:

1. Create a random parent population $P_0$ of size N. Set t = 0.
2. Apply crossover and mutation to $P_0$ to create offspring population $Q_0$ of size N.
3. If the stopping criterion is satisfied, stop and return to Pt.
4. Set $R_t = P_t \cup Q_t$.
5. Using the fast non-dominated sorting algorithm, identify the non-dominated fronts $F_1, F_2, ..., F_k$ in $R_t$.
6. For i = 1, ..., k do following steps:
   a. Calculate crowding distance of the solutions in $F_i$.
   b. Create $\mathrm{P}_{t+1}$ as follows:
      i. Case 1: If $| P_{t+1} | + | F_i | \leq$ N, then set $P_{t+1} = P_{t+1} \cup F_i$ ;
      ii. Case 2: If $| P_{t+1} | + | F_i | >$ N, then add the least crowded N - $| P_{t+1} |$ solutions from $F_i$ to $P_{t+1}$.
7. Use binary tournament selection based on the crowding distance to select parents from $P_{t+1}$. Apply crossover and mutation to $P_{t+1}$ to create offspring population $Q_{t+1}$ of size N.
8. Set t = t+1, and go to Step 3. [5]

#### 4.2.1.3 Experimental Scenarios

Three different scenarios are created to compare the three optimization methods.

Scenario A has a unique optimal solution. Scenario B has more optimal solutions arranged on a Pareto Front. Scenario C is equal to the second with the addition of a constraint.

It is supposed to have a generic product composed of 10 subgroups. Each subgroup has two alternatives to be realized. Each alternative has data that consider all the life cycle of the product, in terms of costs and environmental impacts. Each alternative has this data input:

- Cin: initial cost;
- Cmnt: maintenance cost;
- Cen: energy cost;
- Cmdpmn: cost of manpower for maintenance;
- BOL: environmental impact in beginning of life;

- MOL: environmental impact in middle of life;
- EOL: environmental impact in end of life.

The units of measurement are a generic unit of cost for LCC and a generic unit of environmental impact for LCA.

LCC is calculated as (eq. 2):

$$LCC = \sum_i (Cin_i + Cmnt_i + Cen_i + Cmdpmn_i) \cdot x_i \qquad [2]$$

while LCA is calculated as (eq. 3):

$$LCA = \sum_i (BOL_i + MOL_i + EOL_i) \cdot x_i \qquad [3]$$

where $x_i$ is a binary variable which assumes value 1 if the subgroup i-th is used to realize the product, otherwise it assumes value 0.

The two objectives are:

Minimize the Life Cycle Cost (LCC);

Minimize the Life Cycle Assessment (LCA).

In weighted sum model (WSM) the model is written as:

Minimize

$$w \cdot (LCC/LCC^*) + k \cdot (LCA/LCA^*) \qquad [4]$$

Subject to

$$LCC = \sum_i (Cin_i + Cmnt_i + Cen_i + Cmdpmn_i) \cdot x_i \qquad [5]$$

$$LCA = \sum_i (BOL_i + MOL_i + EOL_i) \cdot x_i \qquad [6]$$

$$x_i + x_{i+1} = 1 \qquad i = 1, 3, 5, \ldots, 19 \qquad [7]$$

$$w + k = 1 \qquad [8]$$

$$w, k \geq 0 \qquad [9]$$

$$x_1, x_2, \ldots, x_{20}\ \varepsilon\ (0, 1) \qquad [10]$$

The two objectives are dimensionally different: LCC has a cost dimension while LCA has an environmental impact dimension. If we want to add LCA and LCC we must make LCA and LCC dimensionless. So firstly we solve a single objective problem, maximizing one time LCA and one time LCC. We obtain LCA* and LCC*. We put these values in the objective function as shown above (eq. 4). So we make LCA and LCC dimensionless and we can sum them. Iteratively we change the values of w and k, respecting the constraint, to obtain the different solutions of the problem. The iteration starts from w=1 and k=0 to arrive at w=0 and k=1, passing through intermediate values as w=0.55 and k=0.45.

In Bi-Mono Model, the model is written as:

$$\min\ LCC = \sum_i (Cin_i + Cmnt_i + Cen_i + Cmdpmn_i) \cdot x_i \qquad [11]$$

Subject to

$$LCA = \sum_i (BOL_i + MOL_i + EOL_i) \cdot x_i \leq TV \qquad [12]$$

$$x_i + x_{i+1} = 1 \qquad i = 1, 3, 5, \ldots, 19 \qquad [13]$$

$$x_1, x_2, \ldots, x_{20}\ \varepsilon\ (0, 1) \qquad [14]$$

where TV is the Target Value.

In Multi-Objective Genetic Algorithm (we have used NSGA-2) there is a chromosome (which represents the generic product) composed of ten genes (which represent the subgroups). Each gene can assume only two values (for example gene 1 can be 1 or 2, gene 2 can be 3 or 4, ..., gene 10 can be 19 or 20). The genetic algorithm optimizes the two objectives simultaneously creating a curve similar to a Pareto front.

Here we have used a population size of 50, a one point crossover with a rate of 0.95 and a single mutation by gene with rate 0.05.

All above described is used for each of three scenarios.

In Test A scenario the data are arranged, so that we obtain a unique solution reported in Table 1.

**Table 1 Solution of Scenario A**

| **Product Sub-groups** | 1 | 3 | 5 | 7 | 9 | 11 | 13 | 15 | 17 | 19 |
|---|---|---|---|---|---|---|---|---|---|---|
| **Min LCC** | 589 | | | | | | | | | |
| **Min LCA** | 430 | | | | | | | | | |

Every optimization method reaches this solution.

In Test B scenario the models are equal to those of the previous scenario, while the data input changes: indeed, in Test B there is not a unique solution, but the solutions are distributed on a Pareto Curve. Therefore, it is analyzed the behavior of the three methods (WSM, Bi-Mono Model, genetic algorithm) within Test B.

A graphical comparison is reported in Figure 1.

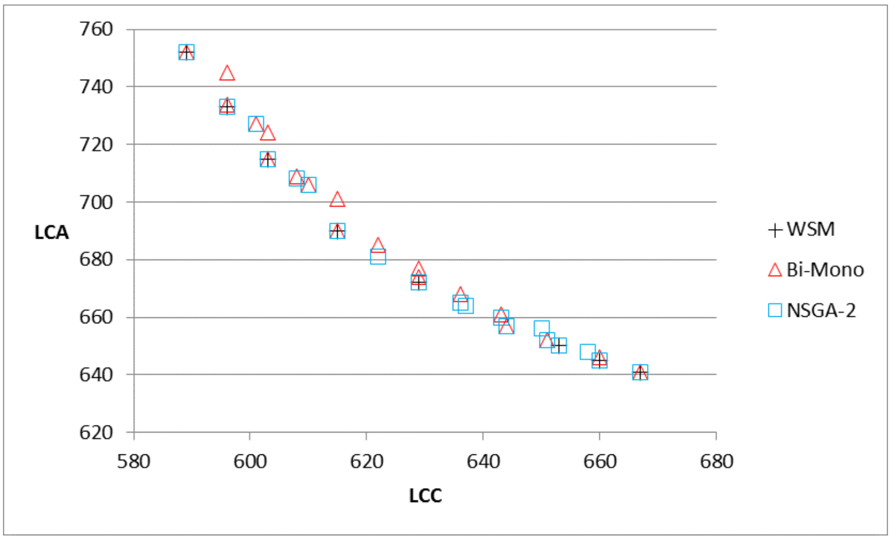


**Figure 1 Graphical comparison of the solutions (Scenario B)**

As you can see NSGA-2 provides a set of non-dominated solutions more complete than the other two optimization methods. WSM finds fewer non dominated solutions than NSGA-2. This is probably caused by the chosen step (we have used a step of 0.05). This is the major disadvantage of WSM. The advantage is that WSM found solutions are surely on a Pareto Curve. This allows a comparison between NSGA-2 solutions and WSM solutions: it is possible to observe from the graph that NSGA-2 solutions, corresponding to WSM solutions, are surely on a Pareto Curve. Bi-Mono, instead, finds a good number of solutions, but many of them are dominated.

In Test C scenario the data input and models are equal to the second scenario, with the addition of a column of values which must respect the following constraint, added in the models (eq. 15):

$$\sum_i g_i \cdot x_i \leq G \qquad [15]$$

where $g_i$ is a generic value of the i-th subgroup, while G is the threshold value.

A graphical comparison is reported in Figure 11

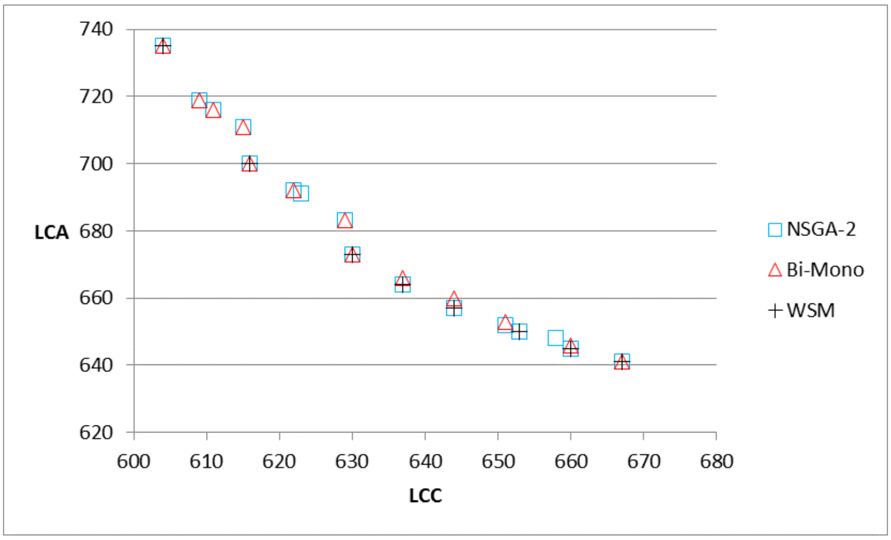


**Figure 2 Graphical comparison of the solutions (Scenario C)**

The graphical comparison is very similar to the previous at Scenario B. NSGA-2 provides more non dominated solutions than WSM. Bi-Mono, instead, provides some dominated solutions. So we can affirm that NSGA-2 is a robust and reliable optimization method.

Summarizing, results of this analysis are:

- NSGA-2 provides a larger number of non-dominated solutions than the other two optimization methods;
- NSGA-2 is a robust and reliable optimization method (it provides optimal solutions: we can say this by comparing NSGA-2 solutions with WSM solutions);
- WSM finds optimal solutions, but fewer than NSGA-2;
- Bi-Mono finds a good number of solutions, but some of these are dominated.

### 4.2.2 Life Cycle Optimization Model Implementation in the LEAP

In this paragraph, the model previously presented is implemented in a Java application. In detail, the so-called “Front-End” (user interface) is developed by ZK [1], that is an open source Java web framework. The so-called “Back-End” is created into an object-oriented Java based framework, JMetal [8]. JMetal is a library to develop, experiment and study meta-heuristic for solving multi-objective problems. Figure 3 shows the model framework.

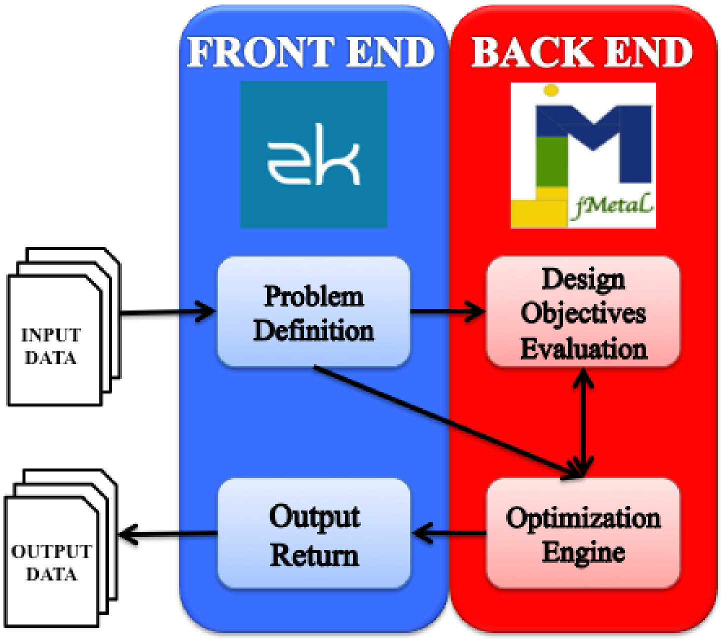


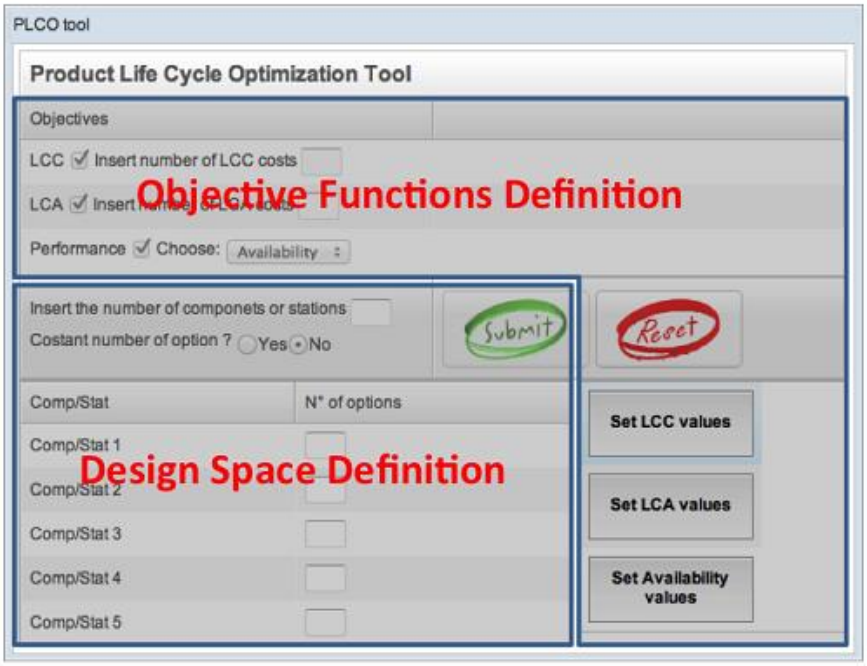
**Figure 3 Model implementation framework**

### 4.2.2.1 Front End

The front-end side of the model includes the Problem Definition module and the Output Return module. The problem to be defined requires various information, which comes out from the customer requests and from the designer needs. Furthermore, the output of the algorithm should be clearly and understandable visualized. Thus, it is indispensable to build a user inter-face (UI) that allows and helps the designer to set in a simple way all the parameters required and permits to display the achieved output.

ZK, an open-source Java framework, is used due to its easiness and rapidity to create a user interface, thanks also to ZK components. They can be configured to fit the developer desires.

The first objective of the developing the UI is to design an index window with specified buttons and forms, in order to set the parameters of the problem. The PLCO (Product Life Cycle Optimization) index user interface is divided into 2 main areas: Objective Functions Definition and Design Space Definition. (see Figure 4)



Objectives

LCC ☑ Insert number of LCC costs

LCA ☑ Insert number of LCA costs

Performance ☑ Choose: Availability

Insert the number of componets or stations

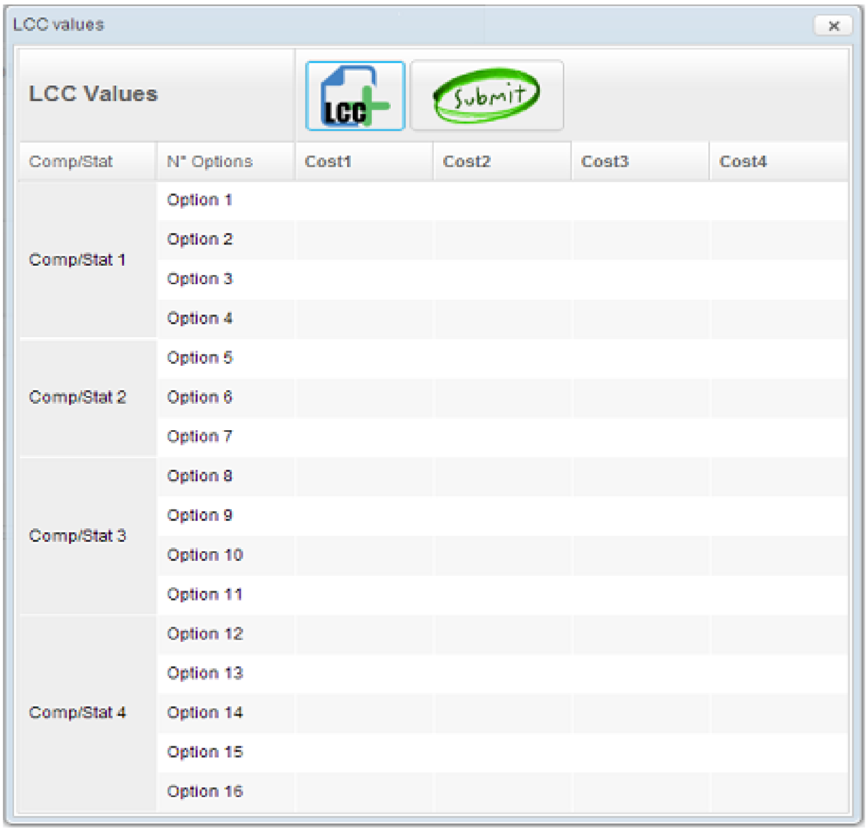
Costant number of option ? ○ Yes ● No

| Comp/Stat | N° of options |
|---|---|
| Comp/Stat 1 | |
| Comp/Stat 2 | |
| Comp/Stat 3 | |
| Comp/Stat 4 | |
| Comp/Stat 5 | |

**Figure 4 Division of the index windows based on the model framework**

In the Objective Functions Definition area, coherently with the model framework, it is possible to:

- Choose the objectives to evaluate in the optimization model ticking the checkbox;
- Define how many components compose the objective functions filling the textbox;
- Insert the data matrix containing the LCC, LCA and Performances values clicking the dedicating button and selecting the file (Figure 5 displays the data insert windows).
- In the other area, the Design Space Definition, it is possible:
- Define the number of variables filling the textbox (components or stations);
- Define the number of options for each variable filling the textbox.

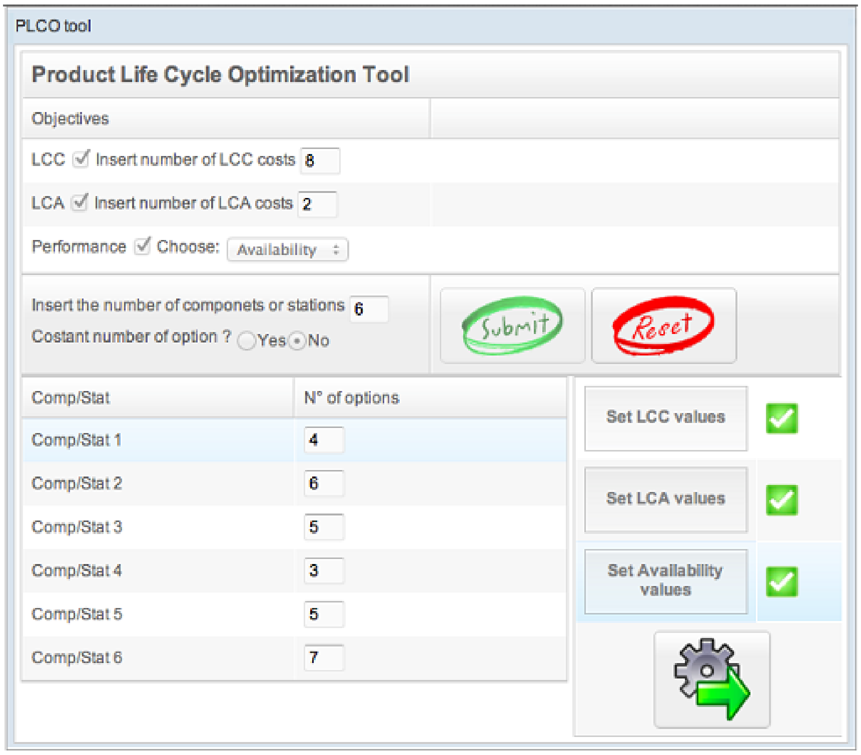


| Comp/Stat | N° Options | Cost1 | Cost2 | Cost3 | Cost4 |
|---|---|---|---|---|---|
| Comp/Stat 1 | Option 1 | | | | |
| | Option 2 | | | | |
| | Option 3 | | | | |
| | Option 4 | | | | |
| Comp/Stat 2 | Option 5 | | | | |
| | Option 6 | | | | |
| | Option 7 | | | | |
| Comp/Stat 3 | Option 8 | | | | |
| | Option 9 | | | | |
| | Option 10 | | | | |
| | Option 11 | | | | |
| Comp/Stat 4 | Option 12 | | | | |
| | Option 13 | | | | |
| | Option 14 | | | | |
| | Option 15 | | | | |
| | Option 16 | | | | |

**Figure 5 Table for the LCC data matrix insert**

All the parameters and the data matrix inserted are checked and validated in order to avoid errors. If all the data are correct the interface shows a hidden button that allow the algorithm execution (Figure 6).

The button invokes a method in the back-end side.



PLCO tool

Product Life Cycle Optimization Tool

Objectives

LCC ☑ Insert number of LCC costs 8

LCA ☑ Insert number of LCA costs 2

Performance ☑ Choose: Availability

Insert the number of componets or stations 6

Costant number of option ? ○ Yes ● No

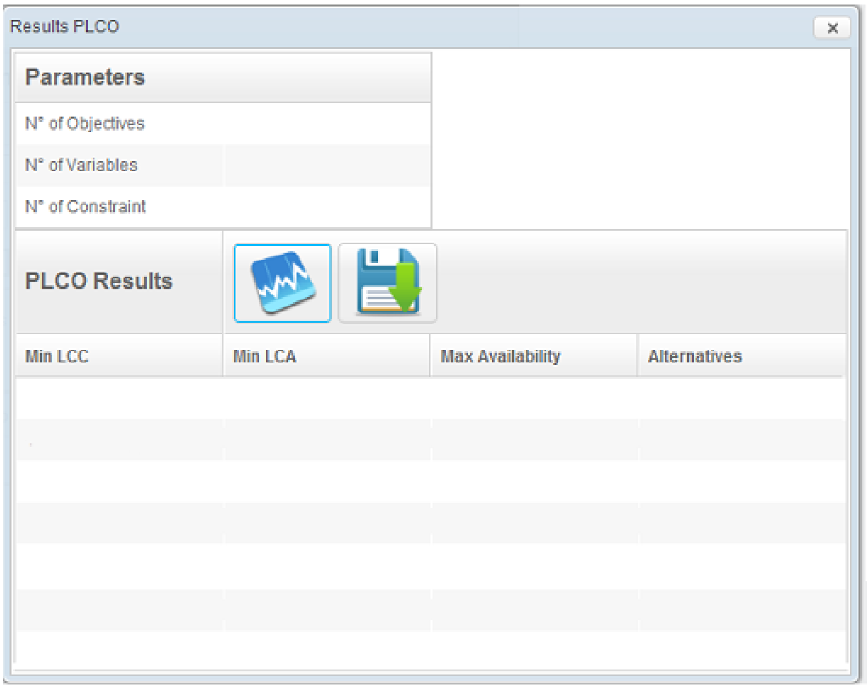
| Comp/Stat | N° of options |
|---|---|
| Comp/Stat 1 | 4 |
| Comp/Stat 2 | 6 |
| Comp/Stat 3 | 5 |
| Comp/Stat 4 | 3 |
| Comp/Stat 5 | 5 |
| Comp/Stat 6 | 7 |

**Figure 6 The execute button showed in the index windows**

The second objective of the front-end side is to implement the Output Return module. Thus, after the algorithm execution it is necessary display in the browser the achieved results. Figure 7 shows the Output Return screen, which reports:

- values of the main parameters;
- values of the objective functions;
- alternative solutions (under development a Multi Criteria Decision Making Approach).

Moreover, it is possible to store the results in an excel file clicking the save button and to display the Pareto front graphs clicking the graph button.



Results PLCO

| Parameters | |
|---|---|
| N° of Objectives | |
| N° of Variables | |
| N° of Constraint | |

PLCO Results

| Min LCC | Min LCA | Max Availability | Alternatives |
|---|---|---|---|
| | | | |

**Figure 7 PLCO output visualization**

The last step is to make the user interface responded to users. The approach introduced is to control the user interface component directly. This approach can be classified to Model-View-Controller (MVC) design pattern. This pattern divides an application into three parts:

- The View means user interface. The page (a XML-formatted language), which contains ZK components, represents this part. A user's interaction with components triggers events to be sent to controllers.
- The Controller plays the role of coordinator between View and Model. It receives events from View to update Model and retrieve data from Model to change View's presentation.
- The Model consists of application data and business rules. In the proposed model it corresponds to the logic inside the back-end side.

When a user interacts with a component (e.g. click a button) on the page, the user's action triggers an event. This event is sent to the controller and invokes corresponding event listener method. The event listener method usually executes business logic or accesses data, then, it manipulates ZK components. A component's state change in an event listener is reflected in its corresponding user interface (UI).

This ZK pattern allows to control user interface via a controller at the server-side; the controller is there-fore the main extension point to integrate any Java library or framework. To integrate ZK with other frameworks (JMetal), it is necessary to write a controller code, in order to use classes of back-end systems.

#### 4.2.2.2 Back End

The back-end side of the model includes the Design Object Evaluation module and the Optimization Engine Module of the Product Life Cycle Optimization (PLCO) problem, by using the NSGA-2 algorithm in JMetal. JMetal [23] provides a rich set of classes which can be used as the building blocks of multi-objective techniques; furthermore it contains a number of state-of-the-art algorithms (included NSGA-2) and a set of quality indicators that allows not only newcomers application to study the basic principles of multi-objective optimization with meta-heuristics but their application to solve real-world problems like PLCO. JMetal is chosen to develop the algorithm because it is simple and easy to use, portable, flexible and extensible.

The Unified Modelling Language (UML) describes the architecture and components of JMetal, used for the PLCO. A UML class diagram, showing the main components and their relationships is depicted in Figure 8. The PLCO problem is built on the basic architecture of JMetal, which relies in an Algorithm solves a Problem, using one SolutionSet and a set of Operator objects.

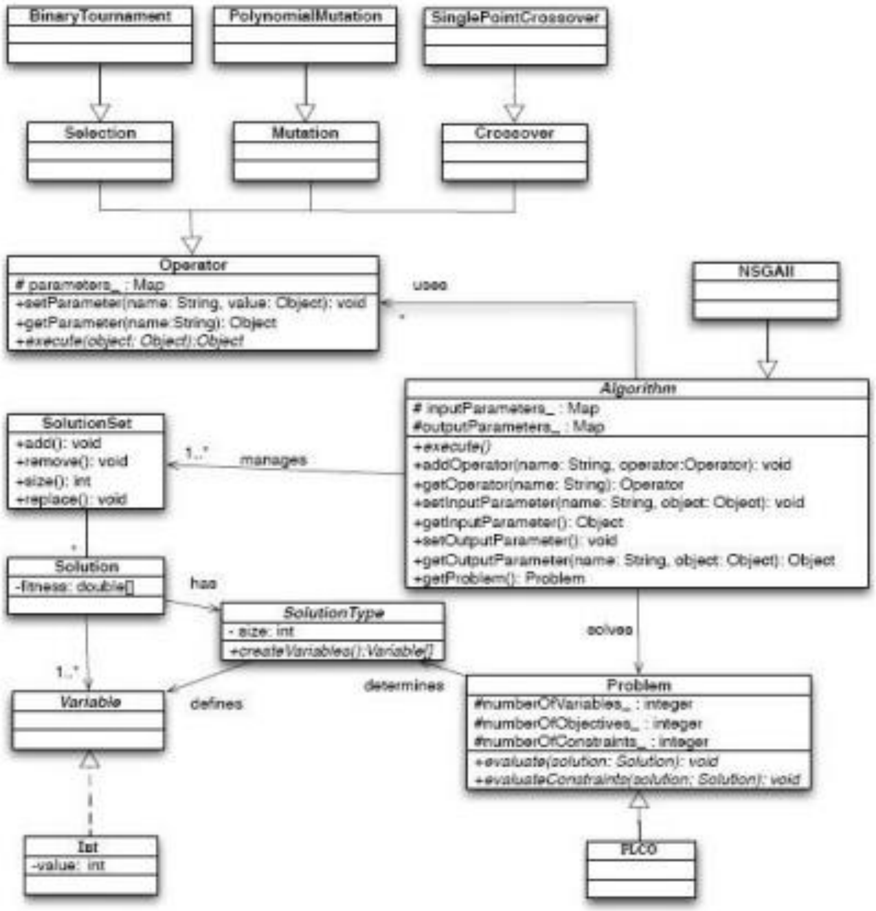

BinaryTournament
PolynomialMutation
SinglePointCrossover
Selection
Mutation
Crossover
Operator
# parameters_ : Map
+setParameter(name: String, value: Object): void
+getParameter(name:String): Object
+execute(object: Object):Object
uses
*
NSGAII
Algorithm
# inputParameters_ : Map
#outputParameters_ : Map
+execute()
+addOperator(name: String, operator:Operator): void
+getOperator(name: String): Operator
+setInputParameter(name: String, object: Object): void
+getInputParameter(): Object
+setOutputParameter(): void
+getOutputParameter(name: String, object: Object): Object
+getProblem(): Problem
SolutionSet
+add(): void
+remove(): void
+size(): int
+replace(): void
1..*
manages
*
Solution
-fitness: double[]
has
SolutionType
- size: int
+createVariables():Variable[]
solves
1..*
Variable
defines
determines
Problem
#numberOfVariables_ : integer
#numberOfObjectives_ : integer
#numberOfConstraints_ : integer
+evaluate(solution: Solution): void
+evaluateConstraints(solution: Solution): void
Int
-value: int
PLCO

**Figure 8 JMetal class diagram of PLCO**

As shown in the previous class diagram (Figure 8), the basic components of JMetal framework are:

- solution encodings
- operators
- problems
- algorithms

Establishing the proper set of those components, it is essential to ensure the correct implementation of the PLCO problem.

*Encoding of Solution*

One of the first decisions, which has to be taken when meta-heuristics methods are used, is to define how to encode or represent the tentative solution of the problem to solve. Representation strongly depends on the problem and determines the operation that can be applied. Thus, to select a specific representation has a great impact on the behavior of meta-heuristics and, consequently, on the obtained results. A Solution is composed of a set of Variable objects, which can be of different types, plus an array to store the fitness value. Effectively, the variable and solution in PLCO problem stand for sets of suitable subgroups of machine, hence must be integer.

*Operators*

Meta-heuristic techniques are based on the modification or on the generation of new solutions from existing ones, by means of the applications of different operators. The NSGA-2 makes use of crossover, mutation and selection operators for modifying solutions. The framework include a number of different operators, thus it is relevant to select the adequate techniques in order to reduce the time complexity of the algorithm. About operators, below the chosen type are listed:

- Crossover: Single-point crossover
- Mutation: Polynomial Mutation
- Selection: Turnover Selection

*Problems*

In JMetal, all the problems inherit from class Problem. This class contains two basic methods: evaluate() and evaluateConstraint(). Both methods receive a Solution, representing a candidate solution to the problem; the first one evaluates it, the second one determines the overall constraint violation of the solution. All the problems have to define evaluate() method, while only problems having constraints have to define evaluateConstraint(). In JMetal, the problem defines the allowed solutions types that are suitable to solve it.

*Algorithm*

The last core class is Algorithm, a conceptual class included in the framework, which must be inherited by the meta-heuristic. The algorithm chosen to solve the PLCO problem is the NSGA-2. As every meta-heuristic developed in JMetal, the NSGA-2 extends the class Algorithm and inherits from it the conceptual method execute() (Figure 8), that is called to run the algorithm. In this class there is the logic of the Engine Optimization module.

In order to ensure the correct execution of the algorithm, a rigorous setting of the parameters of the framework is mandatory. Given the structure of JMetal, the classes to set up the problem are PLCO_3obj and NSGAII_PLCO_run.java. Within the problem, all the features regarding it have to be defined:

- Number of variables: number of decision variables which represent different operations, stations or machines that compose the system
- Number of objectives: number of functions to optimize
- Number of constraint: number of conditions to respect
- Variables upper and lower limit: bounds of the variables
- Data matrix containing Costs, Environmental Impacts and Performance values

Those information are passed to the problem through a java object (object4Plco.java) filled in the controller of the UI.

All the typical parameter values, necessary for running the NSGA-2, can be set up in the class NSGA_PLCO_run.java. One of the main difficulties, which a user faces when he applies an evolutionary algorithm, is to decide an appropriate set of these parameter values:

- Parameters related to the generation of the problem instance: the problem constructor needs to be created from two values. As above-mentioned, it is necessary to configure the class problem, inserting a sting containing the solution type and a java object containing all the indispensable parameters;
- Population size (N): population of candidate solutions. It depends on the definition of the problem;
- Number of generation (T): number of maximum iterations;
- Parameters related to crossover: the crossover operator and the crossover probability must be set up;
- Parameters related to mutations: the mutation operator and the mutation probability must be set up;
- Parameters related to selection the selection operator must be set up.

## 4.3 Life Cycle Optimization in the COMAU case

The previous model, tested in order to check its own soundness, is then applied on a real industrial case. COMAU, a leading manufacturing company active in the automotive sector, provided the case. The company designs in detail and realizes production and assembly equipment for car components (e.g., engine assembly line, body production shops, etc.).

The model has been applied in detail to the real case of a fraction of an assembly line for a small car diesel engine. The layout is reported in Figure 9. The fraction of line is comprised of five stations, which realize the following operations:

- OP180: silicon coating is applied
- OP190: base is assembled
- OP200: 10 screws are filled in
- OP210: 10 screw are filled in and pallets are rotated of 180°
- OP220: screwing in under base is done

All of these locations can have several alternatives: automatic (a); semi-automatic (saut); or manual (m) stations.

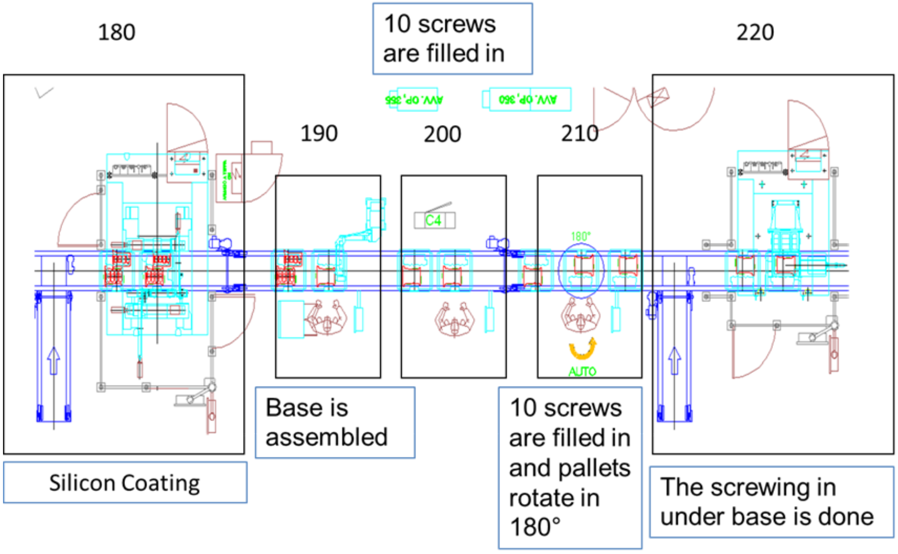

180
10 screws are filled in
220
190
200
210
C4
180°
AUTO
Base is assembled
10 screws are filled in and pallets rotate in 180°
Silicon Coating
The screwing in under base is done

**Figure 9 Reference layout of the line**

Costs and environmental impact related to the stations, and used for LCC and LCA analysis, are:

- Cin = initial cost, the acquisition cost of the station
- Ce = electric energy cost
- Cric = spare parts cost
- Cop = labor cost (the number of workers depends from the type of station: 1 worker for 1 manual or semi-automatic station, 0.2 worker for 1 automatic station)
- Ccon = consumables cost (e.g., oil and grease)
- Cair = air cost
- Cmo = preventive maintenance cost
- Cmorip = corrective maintenance cost
- EIst = environmental impact of the station
- EIel = environmental impact of electric energy
- A = availability of the station

The analysis of life-cycle environmental impacts is really limited, because this is the very first application of this study within the company. Until now, the only data that have been retrievable are: (i) the environmental impact of the station, in terms of environmental impacts related to materials used in the building of the station, and (ii) the environmental impact of the electric energy consumed by the station. Clearly, these data are not sufficient to conduct and perform an LCA analysis, even in a simplified form. However, this is an effort to introduce to the company the concept of Life Cycle Assessment and environmental impact.

The time horizon is 10 years, while the Discount Rate (or Bank Rate) is 1.5%. The model has two objective functions, one that minimizes the product life-cycle costs (eq. 16) and one that minimizes the environmental impact during the whole life cycle (eq. 17). The model has two types of constraints: the availability of the fraction of the assembly line must be greater than 0.95 (eq. 18), and all the locations must have only one station, whether it is automatic, semi-automatic or manual (eq. 19, 20, 21, 22 and 23). Below, the model is written in analytical form:

$$\min \sum_{i=1}^{30} (\mathrm{Cin}_i + \mathrm{Ce}_i + \mathrm{Cric}_i + \mathrm{Cop}_i + \mathrm{Ccon}_i + \mathrm{Cair}_i + \mathrm{Cmo}_i + \mathrm{Cmorip}_i) \cdot x_i \qquad [16]$$

$$\min \sum_{i=1}^{30} (\mathrm{EIst}_i + \mathrm{EIel}_i) \cdot x_i \qquad [17]$$

Subject to

$$\sum_{i=1}^{6} A_i x_i \cdot \sum_{i=7}^{12} A_i x_i \cdot \sum_{i=13}^{18} A_i x_i \cdot \sum_{i=19}^{24} A_i x_i \cdot \sum_{i=25}^{30} A_i x_i \geq 0.95 \qquad [18]$$

$$\sum_{i=1}^{6} x_i = 1 \qquad [19]$$

$$\sum_{i=7}^{12} x_i = 1 \qquad [20]$$
$$\sum_{i=13}^{18} x_i = 1 \qquad [21]$$
$$\sum_{i=19}^{24} x_i = 1 \qquad [22]$$
$$\sum_{i=25}^{30} x_i = 1 \qquad [23]$$
$$x_i \in \{0,1\} \; i = 1,2,\ldots,30$$

where the various costs, environmental impact and availabilities are described above and $x_i$ is a binary variable.

The algorithm returns a series of information, related to each point brought up in the Figure 10 (A, B, C and D), and reported in Table 2. The information includes: the non-dominated solution, related to costs and environmental impacts generated along the whole life cycle, and the sequence of stations that permits the achievement of the previous values.

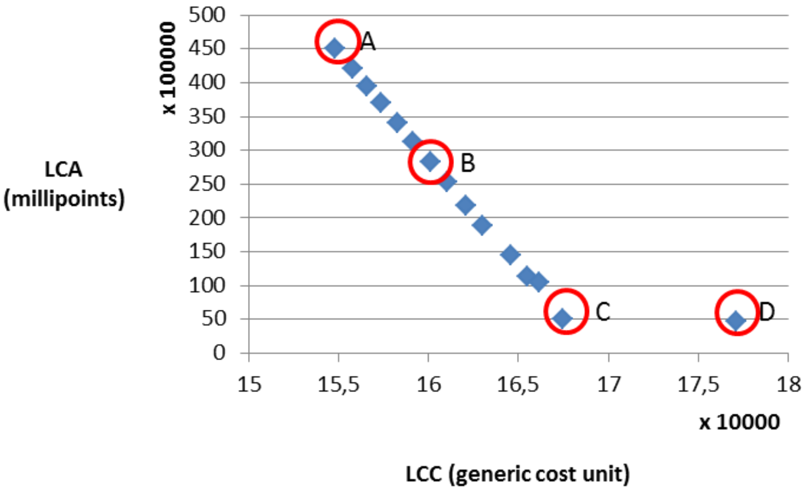


**Figure 10 Graphical results of the COMAU case**

**Table 2 Results of the COMAU case**

| | **Min LCC (unit cost)** | **Min LCA (mill points)** | **Stations** |
|---|---|---|---|
| **A** | 154823 | 45048190 | 2(aut) 7(aut) 13(aut) 20(aut) 26(aut) |
| **B** | 160104.7 | 28230730 | 2(aut) 9(aut) 18(m) 20(aut) 30(m) |
| **C** | 167493.8 | 5062750 | 6(m) 12(m) 18(m) 24(m) 30(m) |
| **D** | 177139.6 | 4627975 | 6(m) 12(m) 18(m) 23(saut) 30(m) |

Data coming from the analysis performed with the PLCO model are therefore evaluated by some experts of COMAU, in order to understand if the values returned by the tool can have sense. Experts compared results with some ones of their previous analyses, concluding that algorithm returned optimal solutions

## 4.4 Conclusions

In this chapter the aim was to show the development and implementation of the so called "LEAP model for design support". Starting from the analysis of the context where companies operate and from the analysis of the literature, it has been demonstrated how product life cycle could be a key leverage in order to compete in the global market. Some methodologies about the evaluation of costs and environmental impacts along the whole life cycle of a product, which are respectively Life Cycle Costing and Life Cycle Assessment, are well known in literature by the 60s. However, a gap is identified about the combination of life cycle methodologies with the optimization methods, which could enable the creation and identification of optimal life cycle oriented solutions. Within LinkedDesign project, a model for the so called "Life Cycle Optimization", was developed and implemented. LEAP model for design support is based on genetic algorithm, which was the optimization method that better suit with the problem faced in the project. LEAP model for design support was developed in 2 components, using Java Web Frameworks: the front end, using ZK, and the back end, using JMetal.

Finally, the model was applied to a case provided by COMAU.

The model covers the identified gap in literature. Furthermore, it is able to cover the needs of COMAU.

The model has been evaluated via questionnaire by COMAU personnel. It is able to hit some strategic values that COMAU asks, like the capability to reduce costs and environmental impacts along the whole life cycle, besides to analyze different alternatives. Some criticalities are instead identified about the simplicity to use the model.

More information about the questionnaire is reported in chapter 8, under the paragraph about the COMAU case.

## References

1. http://www.zkoss.org
2. Barringer H. P., (2003) A Life Cycle Cost Summary, ICOMS 2003
3. Blanchard, B.S. (1991) 1991 Tutorial Notes Annual Reliability and Maintainability Symposium, available from Evans Associates 804 Vickers Avenue, Durham, NC 27701
4. Curran, M.A. (1996), Environmental Life Cycle Assessment. ISBN 0-07-015063-X, McGraw-Hill.
5. Deb, K., Pratap, A., Agarwal, S. and Meyarivan, T. (2002) A fast and elitist multiobjective genetic algorithm: NSGA-II. Evolutionary Computation, IEEE Transactions on, Vol. 6, No. 2, 182-197
6. Landers, Richard R., (1996) Product Assurance Dictionary, Marlton Publishers, 169 Vista Drive, Marlton, NJ 08053
7. Murugan P., Kannan S., Baskar S., (2009) NSGA-II algorithm for multi-objective generation expansion planning problem, Electric Power Systems Research, Vol. 79, 622-628
8. Nebro, A. J., Durillo, J. J., (2013) jMetal 4.3 User Manual. Computer Science Department of the University of Malaga.
9. Okasha N. M., Frangopol D. M., (2009) Lifetime-oriented multi-objective optimization of structural maintenance considering system reliability, redundancy and life-cycle cost using GA, Structural Safety, Vol. 31, 460-474
10. Society of Automotive Engineers (1999), Reliability and Maintainability Guideline for Manufacturing Machinery and Equipment, M-110.2, Warrendale, PA
11. Scientific Applications International Corporation (2006), Life Cycle Assessment: principles and practice, 80 (Technical Report)
12. Triantaphyllou, E. (2000). Multi-Criteria Decision Making: A Comparative Study. Dordrecht, The Netherlands: Kluwer Academic Publishers.