Modular design for quality and cost

Bruno Agard, phd
Dep. Math. And Ind. Eng.
Ecole Polytechnique
Montréal, QC
bruno.agard@polymtl.ca

Samuel Bassetto, ing. jr. phd
Dep. Math. And Ind. Eng.
Ecole Polytechnique
Montréal, QC
samuel-jean.bassetto@polymtl.ca

Abstract— **The purpose of this article is to help managers early in design of new product families. The proposal includes a single level module design formulation that considers quality and cost simultaneously. The method for testing the proposed algorithm is based on a case study of an electro-mechanical assembly device. The performance of the algorithm is compared to that of the 0 module case. The main result is a model and an algorithm that optimizes quality and cost under the constraints of quality and cost. It shows what modules to manufacture, in what quantities, and in which products to use them. The output also provides the predicted quality and cost, based on improvements made to the modules. This research enables the joint optimization of quality and cost by defining the modules to be manufactured.**

Keywords; modularity, design for quality, design for cost, assembly, optimization

I. INTRODUCTION

Mass customization and pricing competition force companies to develop new strategies to cope with greater flexibility, while remaining competitive in terms of price and delivery time [1]. These strategies are undoubtedly key elements in gaining competitive advantage, or at least remaining competitive. However, customers require fully functional products, whatever the price. Their tolerance of product malfunctions is often very low. If a product is labeled "industrial", whether it is a low cost one (T-Shirt, computer flash drive, pen) or a low volume one (airplane, substation circuit breakers, wind turbine), the manufacturer is expected to have fully understood its characteristics. The product is supposed to work properly and faultlessly, and any variability in its functions can be considered a risk to meeting the customer's requirements.

Manufacturers put controls in place to master every level of their processes [2], and barriers are deployed throughout the manufacturing system to prevent faults from occurring [3]. While the integration of quality and quantity has been investigated in classical manufacturing lines [4], the interaction between quality and supply chain design in modular design has not, and constitutes an opportunity for investigation.

The concept of quality adopted in this paper conforms to the view of manufacturers and supply chain managers. It is the degree of conformance of products to predefined specifications and standards. This degree is measured by processes controls and inspections. Actions on quality have an impact on a global defect rate. Through this paper, the degree of conformance will be appreciated by the final failure rate of a product. Preassembled parts affect this indicator. These subsets are called modules, and they are employed to solve diversity issues, like determining an optimal threshold manufacturing quantity. The creation of modules should leads to efficiencies in terms of reduced assembly time and overall cycle time, while maintaining high potential for diversity. When modules are produced from components, resulting modules may have different quality level than its components, depending on actions that have been performed during its manufacturing. Resulting quality of a module could be increased (for instance modules could be "sort" or "test") or decreased (for instance in cases of handlings that produce scratches or default on modules). There is then a possible action on quality each time a module is created, being a positive one or a negative one.

This paper investigates module design considering quality, cost, and the product family-mix. It is structured in four parts. We presents in section 2 our model; in section 3, a case study; and in section 4, the results.

II. MODEL

A. *Description of the product*

Consider ***P*** to be a set of products to manufacture. Product P_k is made up of a set of components C_i. *r* products are considered: $k \in [1, r]$. The components are called C_i, $i \in [1, p]$. In its description, a product is represented as a vector of size *p* that expresses the components that are present (*1*) or absent (*0*) in it (see **Figure 1**). For example, $P_1=(1, 0, 1, \ldots, C_i=1, \ldots, C_p=0)$ means that product P_1 contains components $C_1, C_3, \ldots C_i, \ldots$, but not $C_2, \ldots, C_p$, and so on. Every product P_k has a failure rate $\rho(P_k)$ and a cost $Cost(P_k)$, and must be produced in a certain quantity $Q(P_k)$.

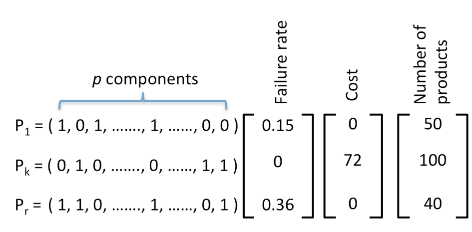

Figure 1: Product modeling

For each product P_k, if a failure rate (resp. cost) constraint is to be solved $\rho(P_k) \neq 0$; (resp. $Cost(P_k) \neq 0$) then $Cost(P_k)$ (resp. $\rho(P_k)$) indicates 0. This modeling is used for the solving branching in equations (1 and 2).

Different products P_k may contain the same components C_i. A module M_j is a set of components C_i. Various options are available in product manufacturing: Option (A): use all the necessary individual components for each product. This is the basic assembly process. Option (B): use a mix of components and modules for each product. This is the module creation process and its uses.

For instance in **Figure 2**, following option A, product P_1 would be made of raw assembly of C_1, C_2 and C_3 and P_2 would be made of C_2, C_3, C_4. The option B would generate the module M_1, made of C_2 and C_3. Then product P_1 would be made of C_1 and M_1 and C_2 would be made of C_4 and M_1. The design of M_1 enables actions on its costs and its quality.

Figure 2 shows the options for manufacturing products P_1 and P_2.

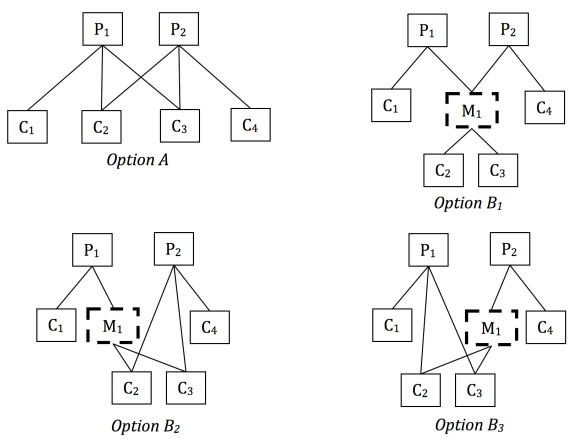

Figure 2: Problem description

A quality issue is understood as a failure that occurs in a product, a component, or a module. A failure can occur during manufacture or during an assembly operation. It can also appear later, at which point it is referred to as a reliability issue. Its main characteristic is to propagate along the supply chain, and such problems are rarely detected by classical functional tests.

In manufacturing the products in ***P***, workers select the components (or modules) needed from different lots. Each lot comes from a specific contractor who guarantees the reliability of the entire lot. The failure rate for the set of modules *Mj* is $\rho(M_j)$. For each lot, $\rho(M_j) \in [0, 1]$, $\rho(M_j) = 0$ means that all the products are reliable, $\rho(M_j)=1$ means that 100% of the modules M_j are faulty. Different scenarios of failure rates are studied below. Failure rates depend on:

- The ability of suppliers to produce reliable components
- The ability of suppliers to identify unreliable components in their processes
- The ability of shipping and incoming departments to identify quality defects

The same applies to the cost of each component and module. In option (A) (**Figure 2**), the failure rate and cost of P_1 will depend on components C_1, C_2, C_3, and in option (B_1), the failure rate and cost of P_1 will depend on C_1 and M_1, and so on.

Depending on the objective for each type of product (in terms of cost and failure), we look to answer the following questions: *Is it better to buy a module M_1 instead of two components C_2 and C_3?* and *Is it better for P_1 and/or P_2 to use it?*

B. Mathematical modeling

The notations are the following:

- C_i is a component;
- ***C*** is the set of components C_i, $C_i \in [1, p]$;
- M_j is a binary vector of size p, called module j. The vector represents the components it contains; for example, module $M_1=(1, 0, —, 0)$ contains only component C_1. A component can also be considered like a module (with only one component).
- ***M*** is the set of modules M_j, $M_j \in [1, q]$;
- P_k is a binary vector of size p, called product k. The vector represents the component it requires; for example, product $P_1=(1, 0, 1, ... , C_i=1, ..., C_p=0)$ means that product P_1 contains components C_1, C_3, ... C_i, ..., but not C_2, ..., C_p, and so on.
- ***P*** is the set of products P_k, $P_k \in [1, r]$;
- $Cost(C_i)$, $Cost(M_j)$, and $Cost(P_k)$ are the cost of C_i, M_j, and P_k respectively;
- $\rho(C_i)$, $\rho(M_j)$, and $\rho(P_k)$ are the failure rates of C_i, M_j and P_k respectively;
- $Q(P_k)$ is the quantity of products P_k to manufacture;
- x is a binary vector of size q, such that $x_j=1$ if $M_j \in$ M'. It is the decision variable.

The goal is to determine the subset of modules M' Є M of minimum cost, such that all products in P can be built, each product P_i respecting its own constraints. If a product P_k has no failure rate constraint, then $\rho(P_k) =0$, in which case product P_k has a maximum cost constraint. If a product P_k has no cost constraint, then $Cost(P_k) =0$, in which case product P_k has a maximum failure rate constraint.

Two kind of constraints exist in ***P***: for r_1 products, there is a maximum cost constraint, and for r_2 products, there is a maximum failure rate constraint ($r_1+r_2=r$). Each product in ***P*** may have a different constraint.

We call (**Figure 3**):

- Aeq a binary matrix of size $q.p$ formed by all products in ***P***.
- $A_{failure}$ a vector of size q that contains failure rates for all modules in ***M***.
- A_{cost} a vector of size q that contains costs for all modules in ***M***.

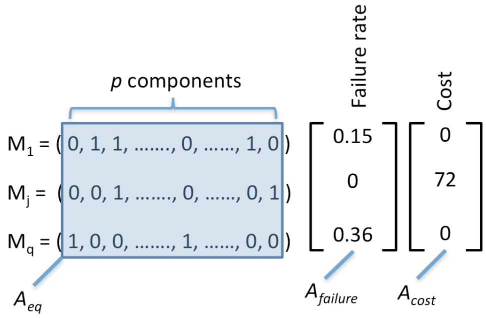

Figure 3: Module modeling.

The formulation is the following:

$$\min_{x} C(x)$$

such that

for all k in $[1, r]$,

$$\text{if } Cost(P_k) = 0,\ A_{failure} \,.\, x \le \rho(P_k) \qquad (1)$$

$$\text{if } \rho(P_k) = 0,\ A_{cost} \,.\, x \le Cost(P_k) \qquad (2)$$

$$Aeq^T .\, x = P_k \qquad (3)$$

where

$$C(x) = \sum_i \delta_{jk}\, Cost(Mj).\, Q(Pk) + \sum_i x_i \,.\, G \qquad (4)$$

$\delta_{jk}=1$, if product P_k contains module M_j

If $Cost(P_k)= 0$ a quality constraints is to be solved for product P_k (equation 1), if $\rho(P_k)=0$ a cost constraints is to be solved for product P_k (equation 2). $C(x)$ is the total cost of the whole product family, and represents the sum of the costs of all the necessary modules (based on the quantity of products, and so the number of each type of module), plus the total number of modules multiplied by a management cost G per module. The management cost has been shown to have a major impact on the number of modules in the final product solution [5]. For computation purposes, G is assigned a fixed value for all the experiments, so that the quality and cost of modules can be compared for analysis. This problem includes the Set Partitioning Problem (Equation 3), which is then NP-hard in the strong sense [6].

C. Problem solving

This optimization problem cannot be solved by standard optimization software for large instances. As explained previously, the problem is an NP-hard 0–1 optimization problem. In order to arrive at an approximate solution, we adopted a simulated annealing procedure. **Figure 4** presents the general scheme of the algorithm.

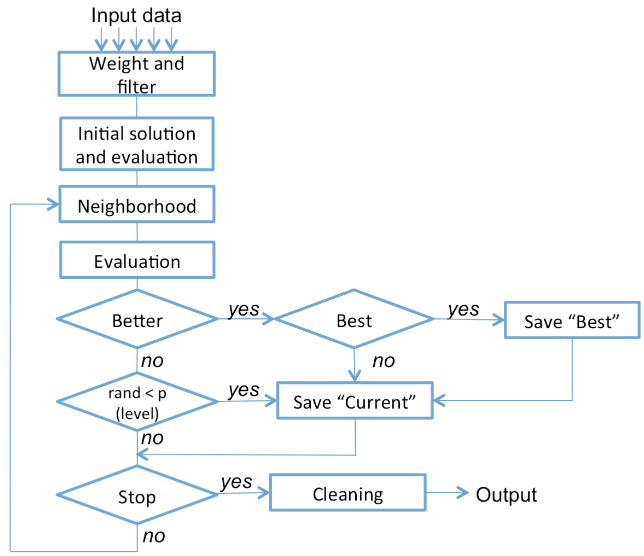

Figure 4: General scheme of the algorithm.

After the input data have been read (a description of the products to manufacture, a list of possible modules, and a parameter solution are generated), the first step, called "weight and filter", follows. For each module, the number of use cases is evaluated (a simple comparison of M_j and P_k, where $P_k\ (i)$ should always be higher than or equal to $M_j(i)$). The use case value of each module represents its weight. A weight of 2, means that the module could potentially be used in 2 different products. All modules with a weight equal to 0 are deleted from the search space. This is the filter operation. An initial solution is selected and evaluated. The initial solution (x) is constructed, such that it contains nothing but modules with only one component. This means that only components are considered at the start of the process. Modules from x are added/removed to improve the solution, as follows: With an initial value x, $C(x)$ (Equation 4) is evaluated, which is the initial temperature. For every constraint, Equations (1) to (3), that is not respected, a penalty is added. $Best(x) \leftarrow C(x)$, $x^* \leftarrow x$, $Level \leftarrow 0$ and $Iteration \leftarrow 0$.

A neighbourhood of x is constructed, and two alternatives are considered:

- If x does not permit the manufacture of all products, respecting all constraints, a module is added to x, and we obtain x'.
- If x permits the manufacture of all products, a random process decides whether to add or remove a module from x, and we obtain x'.

The module to be added or removed is randomly selected, the random process being weighted with the use case number of each module. Modules with a large (small) weight are more likely to be selected to be added (removed).

1. $C(x')$ is evaluated in a similar way to that in step 2. $Iteration \leftarrow Iteration + 1$.
2. If $C(x') \le C(x)$ (with respective penalties), then the neighbor is accepted: $x \leftarrow x'$ and $Level \leftarrow 0$, otherwise go

to step 7; if $C(x') \leq Best(x)$, then the best solution is recorded: $Best(x) \leftarrow C(x')$ and $x^* \leftarrow x'$.

3. A random number α is compared to *p(Level)*, if $\alpha \leq p(Level)$, $x \leftarrow x'$, otherwise x' is rejected.

4. If $Level \geq Max\_Level$, *p(Level)* is updated; if $Iteration \geq Max\_Iteration$, the optimization process is stopped.

5. All modules in x^* that do not appear in any product P_k are removed, $C(x^*)$ is updated. x^*, $C(x^*)$ is the list of non feasible products, and the evaluations of all *Pk* are given.

6. A random number α is compared to *p(Level)*, if $\alpha \leq p(Level)$, $x \leftarrow x'$, otherwise x' is rejected.

7. If $Level \geq Max\_Level$, *p(Level)* is updated; if $Iteration \geq Max\_Iteration$, the optimization process is stopped.

8. All modules in x^* that do not appear in any product P_k are removed, $C(x^*)$ is updated. x^*, $C(x^*)$ is the list of non feasible products, and the evaluations of all P_k are given.

III. CASE STUDY

A. *Description of the product*

This case study is structured around the modular design of headlamp devices.

Figure 5: Illustration of the head-lamp device.

The device presented in **Figure 5** produces a maximum of 700 Lumens to keep the cost of the components low (under \$70). It weighs less than 250g. It has 10 functional parts, among them a lamp, a battery pack, a microcontroller, and a switch. It is made up of 56 components. Many options can be accommodated on this device. The case study is made of 8 options, 15 functions, 7 constraints, 11 products, 1 supplier per function and 1 quality grade per function. It is presented in Table 1. Every component performs a specific function, and is defined with a cost and a quality rating (evaluated based on its failure rate). It is possible to preassemble these modules. The failure rate and the cost of a module both depend on the components it contains. The following has been adapted for computation purposes:

The failure rate of a module or is the sum of the failure rate of the components it contains minus *d*. It is a positive value.

$$\rho(M_i) = Max\left(\sum_{i \in M_i} \rho(C_i) - d ; 0\right) \quad (5)$$

The failure rate of a product is the sum of the failure rate of the modules it contains minus *d*. It is a positive value.

$$\rho(P_k) = Max\left(\sum_{i \in P_k} \rho(M_i) - d ; 0\right) \quad (5bis)$$

Function	Name	Details
F1	LED (mounted on a star PCB, with 2 connections per pole)	Warm
F2		White
F3	Batteries with PCB voltage, driver, and charger	PCB, 3,7V, 5-level driver (on, off, low, middle, high)
F4		PCB, 3,7V, 4-levels driver (on, off, low, high)
F5	Switcher	2 positions; waterproof
F6		2 positions; strong; waterproof (to 100m and gas resistant)
F7	Battery case	Case for helmets with short cables; waterproof
F8		Case to be carried manually with long cables; waterproof
F9	Battery case Straps for helmet	For Camp
F10		For Petzl Écrin-roc
F11		For Petzl Elios
F12	Battery output charger	
F13	Battery waterproof reinforcement	
F14	Color / camouflage	Navy blue
F15		Militarian

Table 1: Details of the 15 functions selected for the test.

Failure rate

It is assumed by this choice that chosen modules act as key elements of the product. By the way the failure of one of them generates a failure of the product and impacts its quality. This assumption is acceptable for core part of a product. Even if inside such core components redundancy is organized, globally the component will be perceived as an entity with improved quality characteristics. The action on quality is modeled by the quantity d. It is also assumed that d will not change over time. This assumption is a strong limitation, as every enterprise has continuous improvement programs. Nevertheless we decide to keep this variable as constant to manage a tractable model.

- If $d < 0$, the module is of poorer overall quality than the components it contains. The assembly operation increases the risk of failure.
- If $d > 0$, a sort operation is performed after the module has been assembled. The failure rate of the module is reduced.

Cost

To calculate cost, we suppose that the cost of a module depends on that of its components.

$$Cost(M_i) = (1 - a)\left(\sum_{i \in M_i} Cost(C_i)\right) \quad (6)$$

- If $a > 0$, the module is less expensive than the sum of its components (this could change, if the contractor profits from the effect of volume sales),
- If $a < 0$, the module is more expensive than the sum of its components.

So, the cost and failure rate of a finished product are directly linked to the modules and the components selected for its manufacture.

B. *Technical constraints of this scenario and numeric values*

Not all combinations of components result in a technically or commercially feasible product. The following constraints are observed: F_1 or F_2: contain only one type of LED / F_3 or F_4: a 4- levels or a 5-levels PCB device / F_5 or F_6: switch is reinforced or non reinforced / F_7 or F_8: battery cases made for helmets or to be carried manually / If F_8, then not F_9, F_{10}, F_{11}: if the batteries are carried manually, then there are no helmet straps / F_9 or F_{10} or F_{11}: helmet straps depend on the helmet / F_{14} or F_{15}: one of two colors available in each product / F_1 or F_2 means that a feasible product must contain either / F_1 or F_2, but not both. Also, if a final product contains F_8, it will not contain F_9, F_{10}, or F_{11}, and so on.

This set of functions and the related constraints return a possible 2,930 different modules. Our study proposes to select the set of modules that will permit the manufacture of the following set of final products.

Function	Cost (in $)	Failure rate (.10⁻⁶)	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11
F1	23	1	0	1	0	0	0	0	1	1	0	0	0
F2	30	1	1	0	1	1	1	1	0	0	1	1	1
F3	60	1	1	1	1	1	1	1	0	0	1	1	1
F4	27	30	0	0	0	0	0	0	1	1	0	0	0
F5	3	1	1	1	1	1	0	0	1	0	0	0	0
F6	4	2	0	0	0	0	1	1	0	1	1	1	1
F7	35	1	1	1	1	1	1	0	0	0	1	1	1
F8	30	20	0	0	0	0	0	1	1	1	0	0	0
F9	4	3	0	0	0	1	0	0	0	0	0	0	1
F10	1	10	1	1	1	0	1	0	0	0	1	1	0
F11	4	5	0	0	0	0	0	0	0	0	0	0	0
F12	2.5	3	1	1	1	1	1	1	0	0	1	1	1
F13	20	5	0	0	0	0	1	1	0	1	1	1	1
F14	2	1	0	0	0	0	0	0	0	1	1	1	1
F15	3	1	0	0	0	0	1	1	0	0	0	0	0
Resulting failure rates with the simple assembly of row components			17	17	17	10	24	33	52	59	24	24	17
Resulting costs (in $) with the simple assembly of raw components			131 .5	124 .5	131.5	134.5	155.5	149.5	83	106	154.5	154.5	157.5

Table 2. Functions and products

Table 2 contains different models of head lamps for manufacture. For example, P_1 is a lamp for cavers. It must be reliable (an expected failure rate of 15.10^{-6}), and we would like to provide it at the lowest possible cost. This lamp contains functions F_2, F_3, F_5, F_7, F_{10}, and F_{12}. Based on a simple assembly of raw components, the resulting product, P_1, will have failure rate of 15.10^{-6}, and a final cost of $131.50.

In order to test the algorithm, we decided to consider a function (F_{11}) that is not necessary in any product. **Table 3** presents the quality (in number of failures per 10^6 products) and cost (in $) expected for each product to be manufactured, as well as the quantities of products to manufacture (in thousands). For instance, for product P_1, the constraint is to obtain an overall failure rate lower than 16.10^{-6} at the lowest possible cost. The quantity produced is to be 50. Comparing these numbers with those in the last two lines of **Table 2**: for product P_1, if each component is assembled individually, the final cost will be $131.50; however, the overall quality will not meet the failure rate requirement (17 > 16). In some cases, raw material assembly is an acceptable solution for the market, but the product could still be improved from a cost (or quality) perspective. For instance, for product P11, the required level for the failure rate is 20.10^{-6}, and, with raw material assembly, it is possible to achieve 17.10^{-6}.In **Table 3** the products that can be made from raw material assembly and meet the market demand are identified in italics (7 products cannot).

Product	P1	P2	*P3*	P4	P5	P6	P7	*P8*	*P9*	P10	*P11*
Target cost per product	X	120	X	X	X	X	80	X	**160**	150	X
Target failure rate per product	16	X	***20***	9	20	30	X	**60**	X	X	**20**
Quantity per product	50	50	50	50	100	50	70	70	100	100	100

Table 3. Constraints, objectives, and quantity per product

In our case study, we noted the quantities produced by a lamp manufacturer: 790. 10^3 products were ordered and split into 11 product types. The results from this case study are presented in the following section.

IV. RESULTS

We consider here the above-defined problem. Modules are manufactured and assembled under cheaper conditions than raw materials, and a sorting operation makes it possible to discard a few of the problematic modules. For computational purposes, a=0.05 and d=1 for Equations 5 and 6. As explained previously, G (the management cost of a module) has a major impact on the number of modules. In the following, G=300. The penalty cost for each non-feasible product is $1,000. For the simulated annealing procedure, the parameters are the following: Max_Iteration = 500 and number of levels = 3, with a *p(Level)* of 0.4, 0.2, and 0.1 respectively; also *Max_Level* = 100 iterations.

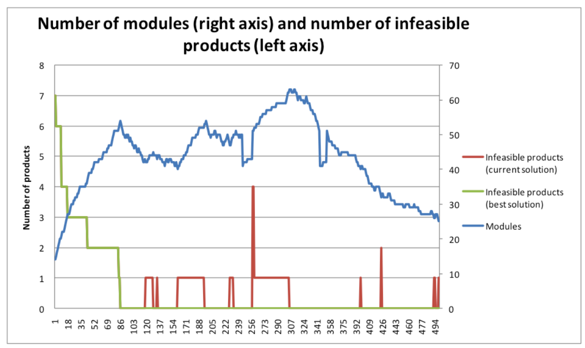

Figure 6. Number of modules (left axis) and non feasible products (right axis).

Figure 6 shows that, starting with 7 non-feasible products, modules are added to the current solution until all the products are feasible. This point is reached after 85 iterations. The algorithm seeks to improve the objective function by removing modules until some products become non feasible, adding to modules for feasibility and removing them for improvement. The end of the process had selected 28 modules. The final solution is 5 times cheaper that the solution that requires raw material assembly. The 28 modules selected make it possible to produce all the required products (respecting both constraints), and it is the best solution found up to now in terms of cost C(x). Some modules are removed from the best solution and 18 are proved to be sufficient to solve the problem.

	Cost	Failure rate	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14	F15
M1	23	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
M2	30	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
M3	4	3	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
M4	20	5	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
M5	61,75	1	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0
M6	65,55	3	0	1	0	0	0	1	1	0	0	0	0	0	0	0	0
M7	57	50	0	0	0	1	1	0	0	1	0	0	0	0	0	0	0
M8	32,3	21	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0
M9	59,375	3	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0
M10	95,475	5	0	0	1	0	1	0	1	0	0	0	0	1	0	0	0
M11	3,325	12	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0
M12	96,425	15	0	0	1	0	1	0	1	0	0	1	0	1	0	0	0
M13	22,8	6	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0
M14	49,4	31	1	0	0	1	0	0	0	0	0	0	0	0	0	1	0
M15	21,85	15	0	0	0	0	0	0	0	0	0	1	0	0	1	1	0
M16	59,85	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1
M17	59,85	21	0	1	0	0	0	0	0	1	0	0	0	0	0	0	1
M18	9,5	5	0	0	0	0	0	1	0	0	1	0	0	0	0	1	0

Table 4. Module composition matrix.

Table 4 shows the solution made up of 18 modules (M_1 to M_{18}), obtained from the assembly of several functions. For instance, Module 5 has a cost of \$61.50 and a failure rate of 1.10^{-6}. It is a package made up of F_2 and F_7. The use case of every module is presented **Table 5**. For instance, Product 4 contains M_2, M_3, and M_{10}.

	M1	M2	M3	M4	M5	M6	M7	M8	M9	M1	M11	M12	M13	M14	M15	M16	M17	M18
P1		1										1						
P2	1											1						
P3		1										1						
P4		1	1							1								
P5					1						1		1			1		
P6									1				1				1	
P7	1						1											
P8				1				1						1				
P9						1			1						1			
P1						1			1						1			
P11				1	1				1									1

Table 5. Product x Module Matrix

Results are provided **Table 6**.

	Quantity (10^3)	Raw results		Legend: *Requirements* OBTAINED **Improved** **Non feasible with raw component assembly: ***	
		Quality	Cost	Quality	Cost
P1	50	17	131.5	*16**	126.425
P2	50	17	124.5	16	*120**
P3	50	17	131.5	*20*	126.425
P4	50	10	134.5	*9**	129.475
P5	100	24	155.5	*20**	147.725
P6	50	33	149.5	*30**	142.025
P7	70	52	83	51	*80**
P8	70	59	106	*60*	101.7
P9	100	24	154.5	21	*160*
P10	100	24	154.5	21	*150**
P11	100	17	157.5	*20*	150.625

Table 6. Results

For instance, product P_1, ordered in a quantity of 50,000, will have a final cost of \$126,425 and a failure rate of 16.10^{-6} . Note that the raw assembly solution (noted in the remainder, C^0) can be made up of two parts (quality and cost): for example, 17.10^{-6} and \$131.50. This method retrieves a better solution for each parameter (quality and cost). Globally, the algorithm outperforms C^0. This example proposes modular design as a solution to cope jointly with quality and cost constraints. Instead of performing a raw component assembly, the modules have to be defined. During preassembly operations, a quality assessment can be carried out, so that the modular design, combined with quality and cost control, becomes part of the continuous improvement cycle of the manufacturing system. The example proves that it is possible to address a particular market by simultaneously considering modularity, quality, and cost control.

V. CONCLUSION

This paper is about modular design. It takes into account the actions taken on cost and quality every time a module is used, which enables the production of a particular product family, each product of which is constrained by limits on one of these two parameters. These results are really encouraging, as they constitute the initial solution for an industrial team wishing to reduce the discrepancy between marketing needs and manufacturing system abilities. This gap is filled by joint action on the modules, that is, action in terms of quality and costs. With their partners' quality management and cost management skills, this team can try to achieve better performance in terms of market coverage. This research opens up opportunities for further study. The first concerns the influence of quantity on the stability of the module. The second concerns the introduction of a list of suppliers, along with their relative performances. Taking this information into account could lead to an optimum manufacturing solution, or to a robust one, which might be more costly but more resilient to disruptions. Finally, a 1-level module design has been proposed here, and a multilevel modeling should be considered as well.

VI. REFERENCES

[1] Da Cunha, C., Agard, B., and Kusiak, A. (2010). Selection of modules for mass customisation. International Journal of Production Research 48, 5, 1439-1454.

[2] Baud-Lavigne, B., Bassetto, S., and Penz, B. (2010). A broader view of the economic design of the X-bar chart in the semiconductor industry. International Journal of Production Research 48, 19, 5843-5857.

[3] Hollnagel, E. (2008). Risk + barriers = safety? Safety Science 46, 2, 221-229.

[4] Colledani, M., and Tolio, T. (2011). Integrated analysis of quality and production logistics performance in manufacturing lines. International Journal of Production Research. 48, 2, 485-518

[5] Agard, B., da Cunha, C., and Cheung, B. (2009). Composition of module stock for final assembly using an enhanced genetic algorithm. International Journal of Production Research 47, 20, 5829-5842.

[6] Garey, M. R., and Johnson, D. S. (1979). Computers and Intractability: A Guide to the Theory of NP-Completeness First Edition. (W. H. Freeman).