**INTERNATIONAL CONFERENCE ON ENGINEERING DESIGN, ICED'09**
24 - 27 AUGUST 2009, STANFORD UNIVERSITY, STANFORD, CA, USA

# APPROACH FOR MANAGING LEAN PRODUCT DESIGN

**Michele Germani, Serena Graziosi, Maura Mengoni, Roberto Raffaeli**
Polytechnic University of Marche

## ABSTRACT
Flexibility is the main keyword in order to face the rapid changing market requirements. Companies need methods and tools in order to implement flexibility over the whole product development process, from ideation to manufacturing. The proposed approach goes towards the concretization of the lean product design concept. It can be achieved if design alternatives and product modifications can be rapidly evaluated in terms of feasibility, resources, cost and time.
The approach is based on a multilevel representation of the product structure, where functions, modules, assemblies and components are strictly interrelated. The complex representation requires suitable software tools in order to model and visualize the entire structure and support the easy user navigation. On the other hand it is necessary to define rules and operators to interact with the structure in order to make product changes and evaluate the possible impact. Finally, this system has to be integrated within the product development flow for exchanging data and information with CAD, PLM and ERP tools. In this paper the general approach is defined and the preliminary software solution is described.

*Keywords: Lean Design, Change Management, Change Propagation, Modularity*

## 1 INTRODUCTION
Lean concept was derived from the Toyota Production System, which in simple terms is defined as: producing what is needed, when it is needed, in the time that is needed, with the minimum amount of resource and space. The whole objective of lean is the elimination of waste and the concretisation of maximum flexibility. This is good to improve productive efficiency within a manufacturing company but currently it is not sufficient to gain new market shares. Companies need innovative models that go beyond the lean manufacturing concept to ensure the transformation of the whole enterprise into a complete lean environment focused on the value creation. This is necessary to respond to the rapid changes of market demands by creating products based on modularity, customisation and sustainability.
The significant change in enterprise performance can come from the adoption of lean thinking through out the entire product life cycle. It requires the development of suitable methods and tools able to support the lean principles implementation. Lean product design can be considered an important part of this lean transformation, as up to 80% of the manufacturing cost is determined in the design stage. It is important to note that a complex design product cannot be easily “leaned out” in production stage. Lean product design means to achieve the configuration of new product characteristics after a rapid and robust evaluation of the impact in terms of feasibility, resources, cost and time.
In order to effectively support lean product design different research areas must be investigated: the methods in order to integrate the different levels of knowledge present within the different company departments, the methods and tools in order to reconfigure the product features in accord to the plant reconfiguration, the methods and tools in order to manage the heterogeneous design information during the product reconfiguration phase etc.
In this context an important problem is the definition of lean design approach and the related tools able to support the rapid evaluation of design alternatives and product changes by taking into consideration the impact of choices over the whole product life cycle. This requires a complete product structure representation where the heterogeneous product characteristics are properly interrelated at different levels of detail. In fact generic functions, implementation principles, specific components etc., should be collected in a unique framework where flows of information and data correlate the different levels.

The aim of the present work is to illustrate the cited framework by focusing on the early evaluation of changes impact. The approach is based on a multilevel product representation structure, as reported in section 3. After a short description of the approach a preliminary software solution is described in order to operatively implement the approach. An experimental test case serves as exemplification of the tool application.

## 2 TOOLS AND METHODOLOGIES TO IMPROVE THE FLEXIBILITY OF DESIGN PROCESS

The need of improving the efficiency of design process has lead researchers to define formal approaches and methodologies in order to better carry out analysis and synthesis design procedures both when developing new products or modifying existing ones. In particular, the design management area has attempted several responses to solve the problems mentioned above like: project management, concurrent engineering, process models, value management, new organizational forms, and information technology support [1]. In this research area the application of lean production principles the so called *lean design approach*, represents an attempt to eliminate waste and non-value adding activities in both engineering and design processes. Lean design considers three perspectives to describe the design process (*conversion, flow,* and *value generation*) each one used to conceptualize the design problem following its own specific vision [2]. Lean approach aims at defining a new design methodology more process management oriented rather than finding the best way to correctly formalize the design intent.

Another way to improve design process efficiency is based on product flexibility, which is conceived as the degree of adaptability for any future change in a product design. In their work Otto et al [3] present an empirical study foundation aiming to develop a method to evaluate flexibility of product design and derive a set of guidelines to guide product architecture to a desired state of flexibility. Indeed, the implementation of modular product architecture can increase productivity in terms of product variety and consequently market segments acquisition.

Design process formalization and representation has been already deeply analyzed in literature: formal models, methodologies and frameworks [4][5][6] have been developed through which design problems and solutions can be described, compared and modified from a functional and behavioral point of view.

In particular, functional modeling applied during preliminary design stages provides flexible models aiding numerous design activities [7][8]. Product function structure represents the first step to generate product architecture, that is the information about how many components the product consist of, how these components work together, how they are built and assembled, how they are used, and how they could be disassembled [9]. Product architecture is the formal representation model that better embodies the link between design and production process phases since it is strictly connected to the layout, the configuration, the topology of functions and their embodiment [10]. While BOM only lists physical components, the explicit product architecture represents a database of product information useful for fast retrieving during design activities.

The link between functional domain and architecture domain is represented by functional modules, drawn from the clustering of functions on the base of energy, material and signal flows. In Höltta et al [11] three modularity methods, already accepted and used in industry, are compared concluding that the choice of a modularity method mainly depends on the modularizing objectives.

In [12], authors affirm that it is important to employ all mentioned levels of representation during the design process. Furthermore, since product representation levels depend each other, it is also necessary to maintain the link among them in order to enable designers to analyse the whole design rationale.

The definition of concrete tools able to systematize the mapping of function into modules and then into physical models associated with product assemblies, subassemblies and components, can effectively support designers in improving design and redesign processes. In particular, for the last one, it is interesting the managing of the engineering changes and their spread across the product architecture from functional modifications.

To support the redesign process Kitamura et al [13] based their approach only on product functional structure. In other academic works [14][15][16] product change process and redesign phase have been focused on the final product structure without considering the functional aspect. Moreover, these methods assess change impact for existing products on the base of static models neglecting product design evolves.

From the point of view of available systems, modern commercial PLM software packages allow only circumscribed impact analysis because they do not take into account functional aspects and they are also unable to predict the change extended effects and their implications. In fact PLM modules neglect the product functional or modular structure and the way functions are explicitly arranged in a physical architecture. High level PLM systems can deal with some functional aspects but they are mainly bounded to process oriented functionalities which do not take into account design rationale.

In such a context, the lacking of concrete software tools and the static approach of current methodologies have lead to define a methodology and a relate software tool based on a multilevel product representation framework.

## 3 APPROACH

The research aim is to support conceptual design phase and in particular to improve and systematize the engineering change management analysis during the redesign process. The proposed methodology and its software tool enable designers to get an estimation of the engineering change impact both from the design and production point of view. Through a quantitative evaluation of the change implications a real quantification of the redesign workload is performed in terms of cost and efforts.

Product architecture and its functional structure represent the main elements of the defined multilevel framework in which graph networks represent the means through which changes can spread. Designers use this structure in evaluating and comparing different technical solutions and choosing the most effective one calculating the necessary redesign activity. In this section the design support system framework is overviewed starting from the discussion of the multilevel representation and presenting the approach elaborated to manage changes from the functional layer.

### 3.1 Analysis of change management processes

Product change processes can be various. For instance, a component can be redesigned because it needs to be stiffer. In this case the modification is bound to the specific part and its interfaces are maintained constant. On the other hand, other changes can originate from product functional structure. For example some sub-functions are removed or new requirements lead to additional new modules.

Therefore, in order to properly manage change propagation it is necessary to classify products modifications. It is worth to bound changes to impacted levels and come to suitable operators to support design changes across different analysis domains.

From the analysis of common design processes, following kinds of changes emerge:

- **Substitution, elimination or addition of a sub function.** In this case, from a functional point of view the change is restricted by a module boundary. Modular structure needs no changes. In the case of strong correspondence between functional and implementing domains – that happens for modular products - changes propagate in a limited way inside a single physical module, as long as functional interfaces, i.e. material, energy and signal flows, are maintained constant.
- **Substitution, elimination or addition of a functional module.** In this case a product is subjected to a more extended modification. Working on an entire module means at least the modification of all its implementing physical assemblies. Moreover, all functional interfaces, i.e. material, energy and signal flows to other modules, are involved. Modules that have or will have connection with the module being deleted or added should be revised. Therefore, many sub-functions need an updating.
- **Modification of functional flows in a module or between different modules.** This kind of modification can emerge from specification updating. For instance, material or energy flows need to be increased. New flows can be added as consequence of sub–functions changes. From implementation point of view, new constraints can be added for a better operation. These modifications show impacts on a single or multiple modules even if they are restricted to a sub-set of their properties.
- **Modification of relation attributes in a physical module or between different modules.** As far as it is possible, the impact is limited by affected module boundaries. That means interfaces should maintained constant finding new implementing solutions, which preserve relations to other modules.
- **Modification of physical module properties.** At implementation level, modules are

characterized by properties, which define general specifications of all the parts they are made of. Changing an attribute means the modification of all the sub parts that need new design or arrangement to fulfill the new requirement. This modification is bound inside the physical module. However these attributes are connected to other modules or component properties. That leads to similar modifications in other parts of the product.

- **Modification of physical components arrangement.** New implementing strategies, such as the employment of new production or assembly technologies, may require a different components structure in terms of assembly and sub-assembly arrangement. For instance, a subassembly can be moved from an assembly to another. This kind of modification is bound to architecture level, since no functional aspects are involved. On the contrary, implementing relations must be revised, added or removed. The modification can spread to different physical modules parts.

In conclusion the analysis of modification has shown two main kinds of situations. The first three points above describe the first type. Modifications originate from the functional layer in consequence of functions or flows or modules updating. The rearrangement of the functional layer leads to modifications at components level. The second type is limited to architecture level. No functional changes are present and modification depends on adopted solution principles and design choices.

### 3.2 Functional structure and product architecture: the multilevel representation and correlation

The proposed framework is based on the first three stages of a multilevel approach described in [17]. They consist in the functional structure, the modular configuration and the product architecture.
The functional structure domain firstly generates a functional representation according to the ontology of functional concepts. Next, the modular analysis generates the product modular configuration starting from the most detailed level of the functional structure. Then, the product architecture domain proposes how these modules and their functions can be implemented.
The way of reading and modifying the multilevel product representation framework can be either top-down or bottom-up, depending on which analysis the designer needs to carry on. Moving downward is useful to highlight how a function or sub-functions is realized by physical components while, moving upward the design intent is elicited in terms of parts functions and product requirements.
The link between the first level and the second one, i.e. the modular configuration, is basically a content-container relationship. On the contrary the step from the second level to product architecture is more complex. The linkages have been found in Pahl and Beitz [18] solution principles, which give concrete form to a function. Parts, in particular components or assemblies, are mapped to modules on the base of the functions they implements. These relations are *many to many*. That means a functional module can be implemented by *many* components while a component can accomplish *many* functions and then be part of different modules. In an ideal modular structure functional modules are perfectly overlapped to physical modules. Normally, as figure 1 shows, also for actual modular structures this is not completely true. Parts that refer to the same functional modules may belong to different product assemblies.

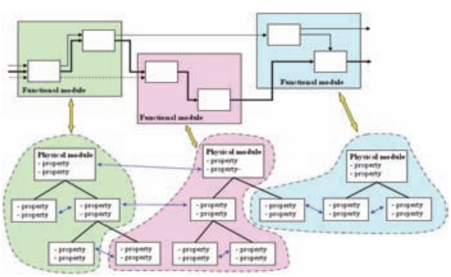


*Figure 1. Linkage between modular structure and architecture.*

### 3.3 Approach to change management from the functional layer

A component tree, whose nodes are the assemblies, sub assemblies down to all significant components, represents the product architecture [12]. Screws, bolts, fixtures, wirings are neglected and

their functions represented by a sub set of the relations between parts. Each block, i.e. each tree node, has a number of properties that characterize it. All block properties form the nodes of the graph structure, which is superimposed to the tree structure. Implementing relations connects these nodes.
The change propagation analysis starts from a property or a relation to be updated. The aim is to browse the structure in order to find all the properties, which are connected to the modified property. The propagation algorithm comes out with the list of the components and assemblies to be redesigned and the affected aspects.
In this paper, the task of a modification originated at functional level is mainly addressed. This kind of modification is more interesting since it matches common design and innovation activities. A new product design often moves from an existing product, by means of some variations of its functional structure. An approach and the relative tools to support this kind of process must accomplish at list with two relevant aspects. Firstly, the functional structure must be captured, represented and conveniently linked with product architecture. Secondly, in the common situation of addition or substitution of functions, relative implementation is still unknown. In this case designer should be supported in the estimation of different implementing strategies in order to pursuit the optimal one.
To cope with this task, the following approach is proposed:

- Existing product is represented in terms of both functions and implementing architecture;
- Functional structure is updated, adding, removing or substituting some sub-functions. Flows are modified as a consequence. Modular structure is rebuilt from the application of modularization algorithms. The system comes out with a list of changed sub-functions;
- Removed sub-functions were mapped to physical blocks that are activated as modification source. New added sub-functions do not have a correspondence to architecture. Therefore designer inputs a new part of the structure to implement what he needs in terms of blocks, properties and relations. This new parts do not still require geometric modeling, simulations or prototyping, but only an abstract definition in term of relations to the old part;
- The impact of the removed parts and the addition of the new elements are measured in terms of change indices in order to have an estimation of workload, costs and design effort. Change propagation is triggered by new added properties;
- Different implementing scenarios based on alternative solution principles can be evaluated in order to come to most promising solution selected as the one with lower impact indices.

### 3.4 Measuring change impact

The above-described method requires the definition of suitable impact indices in order to compute the estimation of design efforts and cost for modifications. On the base of the described approach:

- a modification is introduced in the functional structure and mapped to existing or new physical components. The network of implementing relations between components is updated as a consequence;
- the properties of new components and assembly along with modified properties of existing ones are the source of modifications. Each property triggers a modification sub-graph;
- the whole modification is obtained as the sum of such single propagation path.

In a formal way the total impact can be assessed as:

$$I = \sum_{b} \sum_{p} Ip \qquad (1)$$

Where:

- $I$ is the total impact of the modification. It can express for instance the cost or the effort of the modification;
- $Ip$ is the impact of the property $p$ of the block $b$
- $b$ is the index of the block and $p$ the index of the property

In the following section some $Ip$ indices are presented. Currently, the definition of how $Ip$ indices sum up for each modified property of the block and for each affected block is still an open issue. In fact, barely summing *Ip* indices it will come out in an over estimation of the total impact, since the impact of some property can already be taken in to account in the propagation of another one.

## 4 LEAN PRODUCT DESIGN SOFTWARE SYSTEM

The approach described in the previous section is currently under implementation into a software system. The system structure is quite complex and organized in a modular way as shown in the figure 2. Four main parts can be recognized in the system: the core data structures and relative algorithms which store the information on product functional structure, modular structure ant its implementation, a user interfaces to input and visualize product functional view, its modular structure and its architecture, data import modules from other company IT systems, such as CAD, ERP, PLM, etc., an output module to graphically visualize the outputs of the propagation analysis and the computation of impact indices.

In the following paragraph these modules are deeper described.

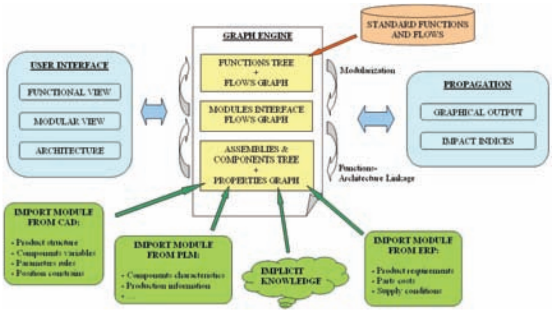


*Figure 2. Software system architecture framework*

### 4.1 Graph engine

This is the core of the system, since it stores the information on the product. Tree main connected levels, each one characterized by different data structures and different sub-levels, represent artifacts.

The first main level is the product functional decomposition. Multiple levels of detail, starting from a “black box” model, represent it. Therefore, function and sub-functions form a tree.

For each level of detail – usually three – sub functions are connected by flows of material, energy and signal. In particular, flows can come from or be directed to the external environment. From data structure point of view, that means flows form a graph with the following constrains, which are fulfilled by the software system:

- a virtual component is added to the structure to represent the external environment;
- flows can connect only nodes of the same tree level;
- the first level is represented only by one function, which represents the root of the tree. First level flows come and go only to the external environment;
- for each level, the system must ensure the coherency of the flows towards the previous level. For instance, an incoming material flow of a second level function must necessarily be present as incoming flow of one of its sub function of the third level.

During instantiation designer is helped by a database of standard functions and flows build on the base of the literature and the specific product needs.

The second main level is the modular structure, which is drawn from the last sub-level of the functional analysis. A Design Structure Matrix approach is followed to group functions in modules. DSM rows and columns represent last level sub-function, while intersection cells contain the flows, which connect them. The matrix is easily drawn by the flow graph. Then, it is transformed in a band matrix using clustering algorithms, which aim to form blocks of functions that minimize the interaction between them and maximize the inner flows. Such blocks identify the products modules. Interface flows between them form a graph.

Finally, the third level represents the physical structure through a blocks tree. Each block lists a number of design properties that are connected by relations. Properties and relations respectively form nodes and linkages of such a graph.
Each block is linked to a module in order to realize the function – architecture connection. Change propagation algorithms use these connections to jump from the functional representation to the physical one and vice versa.

### 4.2 Graphical user interface

System graphical user interface is shown in figure 3. The main part of the interface is made of a tabbed area on the right in which the user can alternate three different views of the product: a functional view (*Function graph*), a modular view (*Modules graph*) and an architectural view (*Architecture graph*). In this area blocks and linkages represent nodes by lines. Double-clicking on the block or on the lines the user can access respectively to function, blocks or relation data for editing purposes. In the first tab, functional analysis sub – levels are shown in the same area and are differentiated by different color brightness. Different arrows and labels distinguish the three types of flows. In the second tab an analog representation shows the modular structure but it is limited to one sub level. Finally, in the third tab, blocks properties can be defined and relations between them added to the structure. Color graduation indicates different levels of granularity, as shown by the pictures in the test case section. Black lines show linkages of parent-child type between an assembly and its component. Blue lines represent a bundle of all the relations among the properties of a couple of blocks.

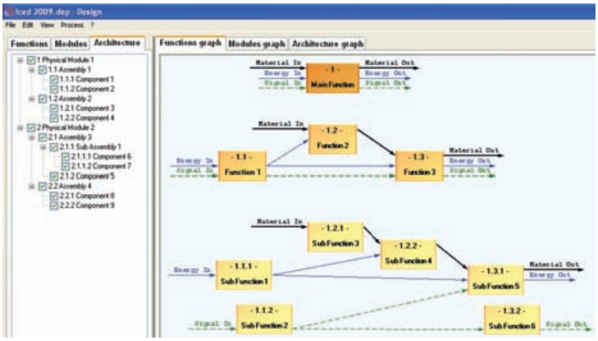


*Figure 3. Software main user interface. Three levels of the functional structure are shown on the right while the tree on the left represents the components structure.*

On the left hand side, the user can choose between three additional trees representing the same data structures. In this case the information is limited but it is useful for browsing and comparing purpose between alternative views. For instance, linkages between a block and a function can be easily instantiated doing “Drag & Drop” of a node of the tree on the left over a function on the right.

### 4.3 Data import modules and propagation output module

Product architecture definition requires inputting in the system large quantity of information. As knowledge on product increases, results are more precise and useful. The definition of the architecture requires three steps:

- Creating the structure in terms of assemblies, sub-assemblies and components;
- Adding for each block main significant properties, such as data on requirements (components technical parameters, layout, test to be fulfilled, etc…), implementation (geometry, orientation, aesthetics, etc…), design process (parts dimensioning techniques, virtual prototyping tools, engaged designers, etc…), production (technologies, suppliers, cost, production lead time, etc…);
- Adding relations between properties, such as physical (fixtures, welding, etc…) or conceptual linkages (relative positions, design decisions, purchase constrain, production constrains, etc…);

Filling in this information can be quite time consuming. On the other hand, part of it is already stored in company IT systems. It is then useful to import this data as mush as possible in order to save time and make the system efficient in its use.

Three software modules are under development for interfacing the software tool with other company systems:

- **CAD Import module:** this module browse an existing product CAD assembly model in order to explore all parts it is formed. The same structure is recreated in the system in terms of blocks. Additional information is represented by design parameters. Using recognizable names for parameters, it is possible to automatically load their values as geometrical block properties from common parametric CAD system models. Also parametric rules, if used, can be recognized and loaded. Assembly position constrains can be read, but require the user to sort the meaningful ones from the constraints used only for modeling purposes. Finally, non-geometrical data, such as material, density, designer, etc… can be read from CAD models.
- **PLM Import module:** the aim of this module is to interact with company PLM system in order to retrieve useful information about the product. In particular, in this context parts data are interesting as well as information on the product as a whole. For example, technical parameters, as well as data on revisions, specifications, etc… can be found together with information about non geometric components such as oil, glue, package, labels, etc…
- **ERP Import module:** from ERP main retrievable information is on parts cost, supplier, and time for delivering. It also gives some generic specification on parts. This information is important because economic and supplying conditions often drive design choices.

### 4.4 Propagation output module

The propagation output module aims to give the designer a feedback on the impact of the introduction of a modification into the product. In figure 4 a prototypal example of such output is presented for the impact of a single modified block property.

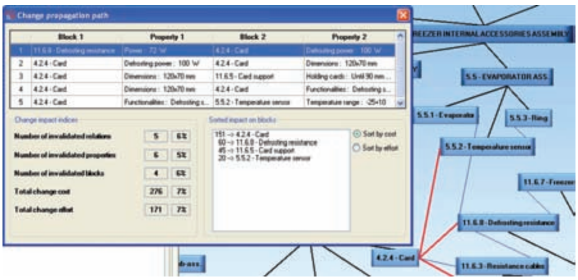


*Figure 4. Prototypal propagation results and impact indices form.*

Basically the system comes out with a composite output made of:

- the detailed list of architecture properties to be updated. They figure in the grid in the upper part of the form and provides the sequence of the propagation through the product;
- some impact indices in order to have a global measurement of the redesign work in the lower left hand side part. They aim to provide a global impact measurement in order to assess and choose different design solutions. In particular the number of invalided relations, properties and blocks is reported in absolute and relative terms. The last ones capture the portion of product, which is affected by the modification and gives an immediate feeling to the designer. Finally, the total change cost and effort is expressed as the sum of each modified property respective indices. Costs and efforts were attributed by the designer to each property at architecture definition stage using normalized values ranging from 1 to 100;
- on the right hand side, the list of blocks to be updated is presented. They are inversely sorted by cost or effort indices. In this way in the top of the list the most critical ones are found;
- a graphical output of the impact of modification in the product structure. The portion of relation graphs is highlighted in order to show the propagation path.

These outputs are drawn from standard graphs analysis techniques. Algorithms traverse the data structures computing impact indices, node grades, counting elements and lengths of paths.

## 5 TEST CASE

The proposed approach has been applied in the study of a family of refrigerators in order to come out with new product variants. The research has been carried out in collaboration with Indesit Company Spa, leader in household appliances.
In particular the method has been applied in the redesign activity of a combined model made of separate fridge and freezer rooms. The aim of the study was to assess the impact of the introduction of an additional specification, which is the defrosting system. Such feature refers to a heating procedure in order to melt ice, performed periodically on refrigerators and freezers evaporators to maintain their operating efficiency.

### 4.1 Fridge functional and modular analysis

The study of the product has moved from the product family analysis. Products requirements and variants have been used to form a functional base for the family. After that a model has been chosen called "Ever fresh", named after a vacuum chamber useful in order to store food for longer time.
The functional analysis has started as usual from a black-box model. Main functionality has been recognized in "Store food" where food is the main material being processed and electricity the main source of energy. The analysis has been deepen for other two levels in order to catch all required sub-functions and avoiding referring to specific solution principles or implementing strategies.
From the third functional level, modular structure has been drawn. Since modularization algorithms have not currently implemented in the software system, Stone heuristics [19] have been used to gather functions in 25 modules. 13 modules comes from dominant flows of material, in particular food, refrigerating fluid, heat, air, signal and electricity; 2 from branching flows of air; 10 from transmission-conversion functions of electricity to heat, energy to signal or mechanical energy to heat. In table 1, modules are listed along with details on heuristics being used to group functions.

*Table 1: In the first and second column functional modules and heuristics used to group functions. On the right list of physical modules as they appear at production BOM top level*

| Functional modules | Flows Heuristics | Physical modules |
|---|---|---|
| 1. Fridge insulation | Heat flow | |
| 2. Fridge closing | Mechanical energy flow | 1. Documents |
| 3. Fridge storing | Food flow | |
| 4. Fridge lighting | Electricity to light conversion | 2. Fridge internal access. |
| 5. Air suction | Air flow | |
| 6. Vacuum storage | Air flow | 3. Packaging |
| 7. Fridge temperature | Heat to signal conversion | |
| 8. User interface | Electricity to signal conversion | 4. Fridge external access. |
| 9. Refrigeration control | Signal flow | |
| 10. Refrigeration cycle | Refrigeration fluid flow | 5. Freezer internal access. |
| 11. Electricity supplier | Electricity flow | |
| 12. Freezer storage | Food flow | 6. Freezer door |
| 13. Freezer insulation | Heat flow | |
| 14. Freezer closing | Mechanical energy flow | 7. Fridge door |
| 15. Freezer opening control | Mechanical energy to signal conversion | |
| 16. Freezer temperature | Heat to signal conversion | 8. Complements |
| 17. Freezer air | Branching air flow | |
| 18. Fridge air | Branching air flow | 9. Isobutane RA600A |
| 19. Air flows | Air flow | |
| 20. Vacuum control | Mechanical energy to signal conversion | 10. Compressor unit |
| 21. Fridge opening control | Mechanical energy to signal conversion | |
| 22. Acoustic alarm | Electrical to mechanical energy conversion. | 11. Fridge case |
| 23. Defrosting control | Signal flow | |
| 24. Defrosting | Electricity to heat conversion | |
| 25. Defrosting temperature | Heat to signal conversion | |

### 4.2 Fridge architecture and its mapping to functional layer

Fridge architecture structure in terms of component and assemblies has been retrieved from production Bill of Material. Such BOM is oriented on production and assembly process and product is arranged in 11 physical modules. In the table 1, physical modules are compared with functional modular structure.
The table clearly shows that the connection between functional and physical structure is not clearly detachable. Even if modularity is recognized from functional analysis, it is not clearly identifiable in the product for a number of reasons. In particular, physical assembly such as "Documents" or "Packaging" cannot be linked to functions level but are affected from modifications at functional

level, however. For instance, manuals, labels, packaging components need to be updated when product is changed. For the rest, physical components need to be mapped to corresponding functional module. As shown in table 2, assemblies contain parts that implement different functionalities. In the software system both functional and physical levels have been represented and connected as in the table.

*Table 2: Part of "Fridge Internal Accessories" physical module and its mapping to corresponding functional modules*

| BOM LEV. | CODE | DESCRIPTION | FUNCTIONAL MODULE |
|---|---|---|---|
| ..2 | W21014598500 | FRIDGE INTERNAL ACCESSORIES | *(MULTIPLE)* |
| ...3 | W21012832403 | "EVER FRESH" SHELF ASSEMBLY | *(MULTIPLE)* |
| ....4 | W14600319801 | Shelf glass AR NF PTF60 | VACUUM STORAGE |
| ....4 | W21500896403 | Button "EF" assembly | *(MULTIPLE)* |
| .....5 | W14400199000 | "EF" silicon seal | VACUUM STORAGE |
| .....5 | W14400222802 | "EF" pipe CB 2000 PTF60 | AIR SUCTION |
| .....5 | W14802934500 | Empty elbow | AIR SUCTION |
| .....5 | W14803208701 | Vacuum button NF PTF60 | VACUUM CONTROL |
| .....5 | W14803266600 | Serigraphed support "EF" | VACUUM STORAGE |
| .....5 | W15200089601 | Vacuum spring | AIR SUCTION |
| ....4 | W14803083804 | POLARW glass front profile 505X78X15 | VACUUM STORAGE |
| … | … | … | |

## 4.3 Example of modification from the functional level

The approach in managing has been tested by simulating the insertion of a defrosting system in the fridge that is a change of the functional layer (figure 5).

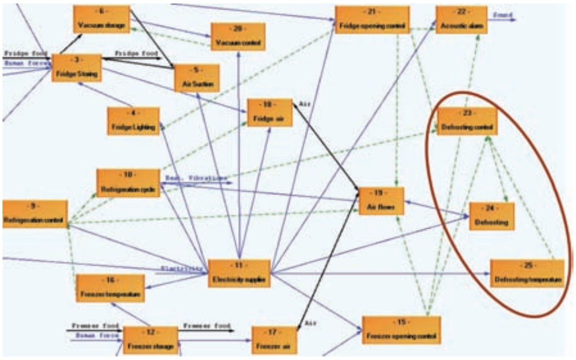


*Figure 5. Insertion of defrosting specification at functional level. Three circled modules and relative flows are added to the modular structure.*

This new requirement introduces new functions and can be grouped in three modules:

- **Defrosting**: Converts electricity to heat in order to melt ice on batteries;
- **Defrosting control**: Receives signal from various modules and trigger the defrosting process;
- **Defrosting temperature**: Converts temperature on batteries in a signal.

The evaluation of the impact has been carried out on the base of possible implementation. Required components and assembly have been introduced in the architecture including them in existing product structure.

In particular:

- Defrosting: is realized with a resistance to be mounted by evaporator battery. It is part of the *fridge case assembly*;
- Defrosting control: It is combined with the *power card assembly* in the *fridge external accessories assembly* which control refrigeration cycle by adding defrosting control functionality;
- Defrosting temperature: It is a sensor that measures temperature on evaporator batteries. It is put

in the *freezer internal accessories assembly*.

Such mapping shows how the introduction of the defrosting functionality impacts in part of three physical modules, as graphically shown in figure 6. In particular for two of them, changes at existing components are required while a new component, i.e. the *defrosting resistance*, needs to be introduced.

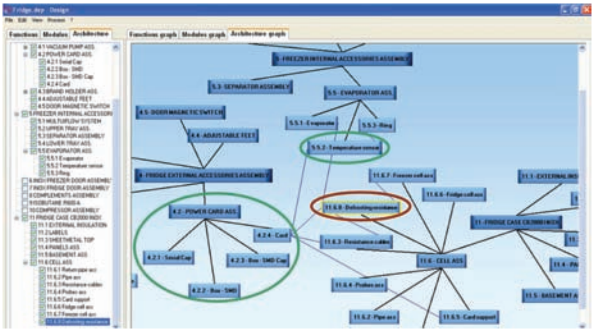


*Figure 6. Insertion of defrosting specification at implementing level. Two circles enclose existing components, which are required of new functions. The red circle highlights the new resistance component.*

An example of the impact of the property *Resistance power* of the *Defrosting resistance* component has already been evaluated in figure 4. It shows how a modification spreads from new components to existing ones. An overall impact cannot be estimated at the moment, but the designer can already have a result from the analysis: *power card assembly* must be redesigned, *temperature sensor* verified if compliant with defrosting requirements and *resistance* added to the *cell assembly*. This change may require finding space for it in the *freezer cell assembly*.

## 5 CONCLUSIONS AND FUTURE DEVELOPMENTS

This paper describes an approach in order to support the adoption of lean product design. In particular the main effort has been focused to the development of a method dedicated to the evaluation of design alternatives in the early design phases. The multilevel product structure has been defined in order to collect the heterogeneous product information, from abstract (functions) to concrete (components). Such a structure has been used as base in order to develop a software tool able to support the estimation of product change impact. The experimentation on a practical case study showed a satisfactory result in terms of useful feedback for the designer work.

This can be considered a preliminary work that needs further study to complete different aspects. First of all, it is necessary a more detailed definition of product change factors thanks to a verification through an extended number of case studies. Then, the software usability needs to be improved in order to increase the navigability and the user interaction through new dedicated “operators”. The approach also needs a further validation to guarantee the inheritance of decision among the different levels, especially between functional and modular ones. Finally, it will be fundamental to improve the software system robustness by experimenting in different design contexts.

## REFERENCES


[1] Ballard G. and Koskela L. On the Agenda of Design Management. In *Proc. 6th Annual Lean Construction Conference*. Guarujá, Brazil, August 1998.

[2] Freire J. and Alarcón L. F. Achieving a lean design process: an improvement methodology. *ASCE Journal of construction engineering and management*, 2002, 128 (3), 248-256.

[3] Palani Rajan P.K. Van Wie M. Campbell M. Otto K. and Wood K.. Design for flexibility – measures and guidelines. In *Proceedings of 16th International Conference on Engineering Design, ICED 03*, Stockholm, August 2003. CD-ROM.

[4] Szykman S. Racz J. and Sriram R. The Representation of Function in Computer-Based Design. In *Proceedings of the ASME Design Theory and Methodology Conference, DETC99/DTM-8742*, Las Vegas, 1999.

[5] Stone R. and Wood K. Development of a Functional Basis for Design. *Journal of Mechanical Design*, 2000, 122(4), 359-370.
[6] Hirtz J. Stone R. McAdams D. Szykman S. and Wood K. A Functional Basis for Engineering Design: Reconciling and Evolving Previous Efforts. *Research in Engineering Design*, 2002, 13 (2), 65-82.
[7] Otto K. and Wood K. *Product Design: Techniques in Reverse Engineering, Systematic Design, and New Product Development*, 2005 (Prentice-Hall, New York).
[8] Chakrabarti A. and Bligh T. P. A scheme for functional reasoning in conceptual design. *Design Studies*, 2001, 22, 493–517.
[9] Fixson S.K. Product architecture assessment: a tool to link product, process, and supply chain design decisions. *Journal of Operations Management*, 2005, 23, 345-369.
[10] Van Wie M. J. Rajan P. Campbell M. I. Stone R. B. and Wood K. L. Representing product architecture. In *ASME design engineering technical conferences, DETC2003/DTM-48668*, Chicago 2003.
[11] Holtta K. M. M. and Salonen M. P. Comparing three different modularity methods. In *Proceedings of DETC'03 ASME 2003 Design Engineering Technical Conferences and Computers and Information in Engineering Conference, DETC2003/DTM-48649*, Chicago, September 2003.
[12] Raffaeli R. Germani M. Graziosi S. and Mandorli F. Development of a Multilayer Change Propagation Tool for modular product. In *Proceedings of 16th International Conference on Engineering Design - ICED '07*, Paris, August 2007, CD-ROM.
[13] Kitamura Y. and Mizoguchi R. Towards Redesign based on Ontologies of Functional Concepts and Redesign Strategies. In *Proc. of 2nd International Workshop on Strategic Knowledge and Concept Formation (SKCF 99)*, 1999, pp.181-192.
[14] Keller R. Eger T. Eckert C.M. and Clarkson P.J. Visualising change propagation. In *Proceedings of International Conference on Engineering Design, ICED '05*, Melbourne, August 2005, CD-ROM.
[15] Eger T. Eckert C.M. and Clarkson P.J. Engineering change analysis during ongoing product development. In *Proceedings of International Conference on Engineering Design, ICED '07*, Paris, August 2007, CD-ROM.
[16] Sosa M. E. Agrawal A. Eppinger S. D. and Rowles C. M. A network approach to define modularity of product components. In *Proceedings of IDETC/CIE 2005ASME 2005 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference*, September, Long Beach, 2005.
[17] Germani and M. Mengoni M., Raffaeli R. Multi-Level Representation for Supporting the Conceptual Design Phase of Modular Products. In *The future of product development*, Berlin, March 2007, pp. 209-224. (Editor: F.L. Krause)
[18] Pahl G. and Beitz W. *Engineering Design. A systematic approach*, 1988 (The Design Council, Springer-Verlag, London).
[19] Stone R. Wood K. Crawford R. A heuristic method to identify modules from a functional description of a product. In *Proceedings of ASME 1998 Design Engineering Technical Conference DETC/DTM*, Atlanta, September 1998.

Contact: Michele Germani
Polytechnic University of Marche
Department of Mechanics
Via Brecce Bianche
60131, Ancona, ITALY
Phone: +39 0712204969, Fax: +39 0712204801
E-mail: m.germani@univpm.it

Michele Germani is aggregate professor of Computer Aided Design at Polytechnic University of Marche, He received his PhD in the field of Mechanical Engineering in 1999 (University of Padua, Italy). He is author of 112 scientific papers about Design Automation and CAD systems published on scientific journal and international conference proceedings.