INTERNATIONAL CONFERENCE ON ENGINEERING DESIGN
ICED 05 MELBOURNE, AUGUST 15-18, 2005

MULTI-LEVEL PRODUCT MODEL TO SUPPORT THE LCA USE IN THE EARLY DESIGN PHASE

Germani, M., Mandorli, F., Mengoni, M.

Keywords: Life Cycle Assessment, Life Cycle Inventory, CAD/PLM, Multi-Level Product Model

1 Introduction

Eco-design is an activity that aims to minimise the possible environmental impact of a product through the analysis of environmental characteristics at the product design stage. However, it is rather difficult for designers to assess the environmental properties and attributes of their products. They should have simple-to-use methods that allow for evaluations that are quick and easy, but also accurate

The environmental assessment involves the identification, quantification, evaluation and prioritization of environmental aspects in relation to the product system. For this, effective assessment methodologies are required; recently, life cycle assessment (LCA) and matrix-based methods have widely been used as analysis tools to evaluate the performance of a product system. Life Cycle Assessment, as a cradle-to-grave approach, provides a framework and a method to identify and evaluate environmental burdens associated with the life cycles of materials and services. LCA is becoming increasingly more important as an assessment tool within the development process of mechanical products that is our field of interest. LCA can be used to estimate the environmental impact or "cost" of a product during the entire life cycle, and to highlight weak points and areas requiring special attention and improvement [1]. However, a detailed full LCA can be difficult to apply at the design stage, because of its tedious, expensive and time-consuming procedures. On the other hand, matrix-based methods are usually considered as simple approaches to present information about a product's environmental impact in a systematic and clearly arranged manner, while, in many cases, quantitative information is lacking. Therefore, there is a need for dedicated, innovative and simplified methods that involve less cost, time and effort, but still providing similar results to more detailed and costly exercises. A possible solution is the development and use of simplified LCA methods. In simplified LCA, it is necessary to identify those areas that can be omitted or simplified, without significantly affecting the overall results. But a recent study showed that such methods are not yet suitable for all product categories and they lack of robustness [2].

In this research area, the long-term objective of our study is the development of a new approach to facilitate the use of LCA methods during the design process, into the mechanical design field, facilitating the evaluation of design alternatives from the environmental viewpoint. The methodology will be based on the definition of rules, procedures and software tools to organise design data for supporting the LCA analysis. Since the current LCA software

systems are not conceived to be integrated within the design process flow, they will be customised to satisfy the designers requirements.

Firstly, as reported in this work, a correspondence between the information generated along the design process and the LCA inputs has to be established. In particular a multi-level product structure and a preliminary dedicated classification of design information are described.

2 LCA and product design

Designing a product is mainly the task of a multi-disciplinary team trying to consider as soon as possible the constraints of each people involved in the product life cycle. Environmental aspects are, today, a crucial factor. Research work done in this field was concentrated on developing methodologies and tools to support the environmentally conscious design [3], but they provide essentially qualitative suggestions not sufficiently deep to adopt new solutions. However, the complexity and generality of LCA systems, which provide quantitative comparisons, generate restrictions to their use in current mechanical products and systems development, especially for SMEs [4], because it is apparently in contrast with a desirable short time to market and a low product cost. Furthermore, environmental assessment is typically performed after the design process, when the product data are more consistent but the possibility of influencing the design is minimal. In according to Bhander et al. [5], LCA tools need to be integrated into the design process and the environmental impacts need to be evaluated concurrently to the other design procedures providing an immediate feedback to the design team. Similar concepts are reported in Dewulf and Duflou [6] and Rosemann and Meerkamm [7]. In both cases interesting methodologies are proposed to evaluate product alternatives during the early phases of design process. Ernzer and Birkhofer [8] describe a method to enable the designer to carry out an LCA during the detail design phase using data generated in a CAD system. Nawata and Aoyama [9], propose a similar concept specifically applicable to machined parts. The system generates life-cycle assessment feedback for the design process through the linkage of life-cycle inventory data with computer-aided design/manufacturing data by feature-based modeling. By evaluating the machining process in detail, this system provides useful information that can be used to improve the machining process, representing a valuable aid to the life-cycle assessment if machined products. In [10] a feature-based CAD model is extended with explicit and implicit information in a useful manner to be adopted within an LCA system. All methods consider a CAD geometric representation as a collection of data to assess the environmental impact. In our opinion, after experimental tests [10], when a three-dimensional CAD model is completed the design alternatives have been largely decided, hence many further processes, in a concurrent engineering context, have already started. As a consequence, it is difficult to modify a project in order to satisfy the environmental suggestions derived from the analysis, because large investments have already planned.

In this context the objectives of present work are set as follows. The central research question is how we can rapidly and easily compile the Life Cycle Inventory (LCI) required to perform a complete LCA analysis. The retrieval and collection of inventory information on all processes of product life cycle are time-consuming activities, and they make an accurate LCA very expensive. Our goal, in order to bridge the gap between product design and LCA, is the development of an effective framework to facilitate the data migration between digital

product models, generated along the different design phases (conceptual, embodiment and detail design), and LCI models.

3 Multi-level product model for LCA

A multi-level product model has been studied in according to the hierarchical product structure reported in [11]. The basic approach for the framework to be developed, is to take advantage of the information stored in such product model. The model collects the product information at different levels of detail and it can be managed by dedicated design systems and by widely used design support systems (CAD/PDM).

3.1 Multi-level product model description

One can consider individual models integrated into what is known as a product model as "*implicit data bases*", already containing useful information, though being in an inappropriate form for computer-aided compilation of the LCI. In some cases this information needs to be integrated with other process dependent data. One of our objectives is to define methodologies and techniques to enrich the information contained in the product models and in parallel to define procedures to identify, extract and/or convert part of the product models data in order to make the implicit information stored in the product models explicitly available in a suitable form for computing the LCI.

To provide life-cycle related data for LCI computation, it is necessary to analyse and classify the different types of data stored in the product model: we classify data as geometric and non-geometric according to the different sets of procedures - computation kernel - required to extract information from the model.

We need to represent the product model in such way that is possible to organise information both to support the product configuration and to extract the information needed to perform an LCA analysis. An organic multi-level product model is necessary to manage coherently the product information during the different design phases.

The proposed multi-level product model representation is based on the concept of *self-configuring components* (SCCs). By SCCs we mean structures with the capability of representing the different types of knowledge required to make the component/module able to fit different situations and able to provide the right configuration once the designer has defined the general requirements. Such structures are managed within an appropriate framework, that let the structure interact with each other in order to retrieve the information (at the right level of detail) required to *behave* correctly within the model. Simple components can then be used as constitutive parts for more complex modules, and they can be easily reused within new models thanks to their re-configuration capabilities.

We define a self-configuring component (SCC) as a structure made by internal data and a configuration engine able to process inputs and to produce outputs (figure 1). The internal data are the basic information (shape is a particular type of information) that can define the component. The configuration engine is a set of procedures able to perform computation on the inputs in order to perform the detailing of the component and to produce outputs. The SCC inputs are: the set of functional information (functions required) the SCC's needs to start

the self configuration process and the set of technological constraints imposed by the *environment*.

The SCC outputs are: the component configuration given in terms of shape (at different levels of detail) and the functions that the configured component is able to provide.

The level of detail of the outputs depends on the level of details of the inputs, i.e. if the engine has enough information, it can produce more detailed outputs. For example the geometric information related to a shaft definition could be a simple axis at the lowest level of detail. It could become a cylinder when the information to perform the dimensioning is available. And, finally, it may have a more detailed shape when form features are added to the cylinder to define the shaft interfaces to other components.

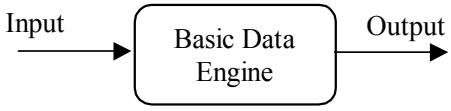


Figure 1. Representation of the SCC concept.

The more the design process advances and the more detailed the information becomes, the more the SCCs engines will produce detailed component configurations, and, therefore, the product model will be more detailed.

From the environmental viewpoint, the shaft cited above, when at high level is represented by an axis and it collects non-geometric attributes, for example the manufacturing material, it can be used for a rough environmental impact estimation, then, when at low level the design information is more detailed and accurately defined (dimensions, manufacturing operations, material treatments, ….), the LCA process can be refined. But the preliminary analysis, that can be partially automated, as reported below, and performed at different steps, allows to evaluate design alternatives by quantitative results in the right moment of the process.

To support the SCC definition it is necessary an effective functions' classification. In literature [12] a well-structured functions taxonomy can be found. We divide them into two main groups: "interface" functions and internal functions.

The first group connects two or more objects and it implies a flow of information (signal, energy, but also design information with different levels of abstraction) within the product model. The internal functions refer to functions limited within the component and do not exchange direct information with the context. For example a shaft on which two gears are positioned at two ends, can receive torque and rotation from a gear and it transports the energy among the ends. The exchange of energy with the gears requires the presence of "interface" functions: "receive/import energy" and "give/export energy". In the gears have to be present the correspondent functions exchanged with the shaft, the functional definitions are "give/export energy" and "receive/import energy". The shaft functional definition includes also an internal function that is "transport energy", but the context (interfaced components) is not influenced by this function.

Furthermore, "interface" functions have been classified as follows: functions provided by the component and functions required from the component. If a function is required then the presence of a dual function, encapsulated within another component of the product model, is

necessary (if the component A requires a function *X,* then component B must provide the dual function $\underline{X}$). The terminology used to define/classify functions has been based on this duality (for example import/export or supply/store). In the module architecture instantiation phase, when the SCCs are used, an automatic control of the dual functions presence has to be foreseen.

The development of SCCs for product configuration has been devised because it allows the determination and management of different product representation levels (from abstract to detailed). This, as outlined earlier, is evident if referred to the product shapes, but functions have a similar structure (i.e. the function "receive/import energy" mentioned above for the shaft, is an abstract concept, to obtain a useful definition for shape component is necessary to "transform" the function in a less abstract concept, for example "connect shaft and gear"). Hence, functions can be defined as structures able to store functional information at different level of detail/abstraction. We may look at a function as a tree structure where each branch represents a different implementation of the root function, and the level of depth of the tree represents the level of detail of the function representation (figure 2).

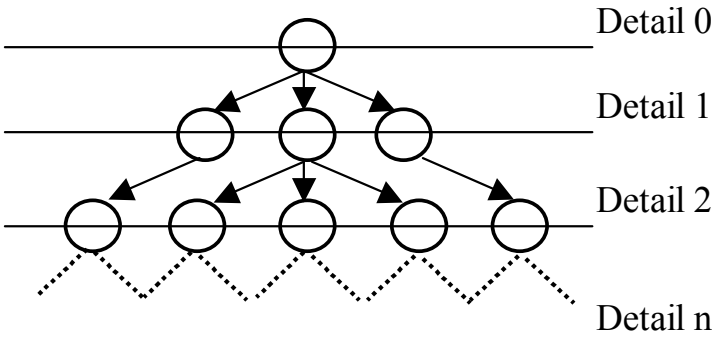


Figure 2. Hierarchical structure for product representation

The nodes and leaves of the structure store the information required to define the function at a particular level of detail, including the geometrical information. The structure branches represent “if-then-else rules” that drive the context-dependent implementation of the function (e.g. returning to the shaft and gear, the function *connect* can be defined as a root node with two branches, one for complete connection and one for partial connection).

At the lowest functional level of detail, the shape information can be related to the function. They can be traditional form features (as holes, housing, etc.) and/or standard mechanical components (as rolling bearings, pins, keys, screws, nuts, etc.). The traditional form features can be retrieved from a CAD data structure, if it is opened to implementation. Standard components can be stored within specific databases linked to low-level functions. For example, if a function "guide rotation" is required from the shaft, a corresponding function "support rotation" will be provided within another component of the module. Such function can be directly linked to a bearings database. When the shaft design information evolves (shaft speed, loads etc.) it can be possible to automatically instance the required specific bearing.

The exploitation of the product model allows organising also the information needed to perform the environmental estimation. Such information is automatically contained, explicitly and/or implicitly, into the different levels of the product definition.

3.2 Framework to manage the product model data migration towards LCA

In order to extract the information from the product model and represent it in a suitable form for LCI computation we use attributed structures. In particular, we use attributed structures to represent features, that are regarded as clusters of information that are relevant in a specific domain and in a specific level of detail. The syntax and semantics of these structures are determined through properties, which are their attributes.

Features are used to explicitly represent the information required for supporting computation of the LCI. Information explicitly represented in the product model and required for LCI can be directly mapped into the appropriate feature attribute, though appropriate procedures are required to map the implicit information into attributes.

These generic features can be generated at different level of details, along the main phases of the design process: the conceptual phase, the embodiment phase and the detailed phase.

The basic approach to achieve the information extraction from the SCCs components and to perform an LCA, is by introducing the concept of *LCI_X features* (*X* indicates the generic property) on an attributed base. *LCI_X features* are used to organise the information and store the procedures required to extract this information from the product model. These features define the basic structure to interface LCA related product and process information with LCI systems.

Due to the analogy with *form features*, that are structures aimed at maintaining consistency among high level functional and technological information with low level geometric data, we have named our proposed structures *LCI_X features*. By assuming that each function, solution principle, component, sub-assembly as well as the whole product assembly can be associated, with growing accuracy along the design phases, to specific manufacturing processes and specific amounts of material, we may consider the LCA product stage input data as a tree structure that replicates the traditional design structure. The SCCs provide data and information in according to the current design phase (figure 3).

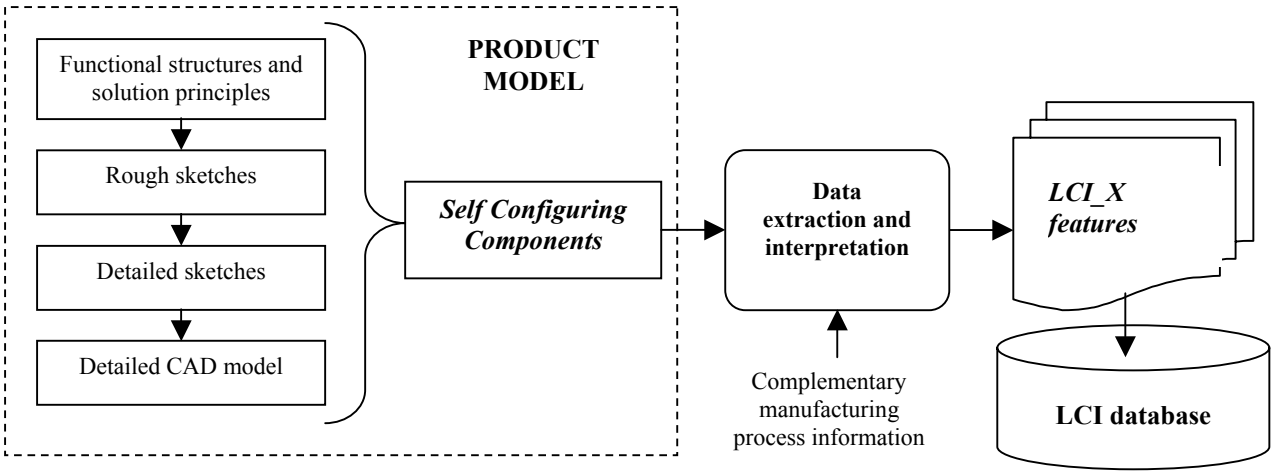


Figure 3. Information flow for LCA analysis

The types of information and data stored in the SCCs can be roughly classified as functional information, energy flows information, material flows information, solution principle used,

product layout information, geometric information, topologic information, modelling history information and information on the global properties of the modelled object.

We may look at *LCI_X features* as structures defined by property/value couples and methods to get and set such values from the information cited above. It is evident that while the approach is general, each specific get/set method needs to be customised according to the design phases, types of product, manufacturing and assembling technologies.

At the present stage of the work we have focused our attention on procedures to retrieve semi-automatically LCI relevant data from CAD models. In order to be able to address this issue first of all we have to classify the different types of information stored within the CAD models from an LCI point of view. In particular, it is necessary to identify information immediately available and information that requires computation to be elicited. Today most CAD systems store the product model assembly as a tree of components. Each component stores its modelling history, including geometric parameters and constraints. Additional information is stored in form of attributes linked to the assembly, to the components or to geometric entities of the component. We have identified the information stored within these models as local or global, implicit or explicit. We define as global the information that is relevant for the processes related to the product as a whole; on the other hand by local information we mean the one that is relevant for the processes related to the different components that are part of the product. Both global and local information can be either explicit or implicit. Explicit information, for example dimensions and form features, can be directly accessed by the mapping procedures while implicit information, for example number of components and component volume, requires the identification of appropriated procedures to be made explicit.

The preliminary identification of some properties that define an *LCI_X feature* are: Name, CAD_Material, Material_Amount, Material, Density, Manufacturing_Process, Assembly_Processes, Assembly_Materials, Assembly_Materials_Amount. Each property's value can be get/set with an appropriate method. The methods to get/set these values are described below.

Name: it is the component name. This value can be retrieved from the CAD tree assembly structure and directly mapped into this feature property. *LCI_CAD_Material*: it is the component material name. This value can be directly retrieved from the CAD component attributes. *LCI_Material_Amount*: it is the amount of material required to produce the component. This computation is a function of the manufacturing process, the material type and the component shape. *LCI_Material*: it is the component material name as required by the LCI system. This name could be slightly different from the name used in the CAD system. The use of classified standard names may reduce the risk of mismatching. However, the method is able to find the right correspondence or to present to the user different alternatives. *LCI_Density*: it is the value of the materials density. The density value is retrieved from a table of densities by using *LCI_CAD_Material* name as entry. *LCI_Manufacturing_Process*: it is the set of processes names to manufacture a component; it can be partially extracted from the CAD model shape, and it has to be completed by the user interaction. *LCI_Assembly_Processes*: it is the array of the processes names used to assembly the group, as required by the LCI System. This computation is a function of the material type and the available assembly technology. *LCI_Assembly_Materials*: this property is meant to store additional materials required for the assembly process. It is an array of material names as required by the LCI system. *LCI_Assembly_Materials_Amount*: it is the amount of material

required to assembly the group. This computation is a function of the assembly process, the material type and the components shape.

The same formalism can be used to organise the high level features information. For example if the product is represented as a set of solution principles we can define features attributes and properties related to such level considering a rough quantification of needed values (weight, volume, etc.).

4 Multi-level product management tools

4.1 Prototypal software system architecture

A design support system for product design development, in according to the framework described above, should have the following characteristics: the functionality to define components on the basis of functional and technological requirements, the functionality to represent components at different level of geometric detail, an environment where to define the overall architecture of the product and to set the product functional and technological specifications. Such tool should benefit of the links with the geometric kernel of a CAD system and with the database of a PDM system, where processes information and data can be stored. This system should be interfaced with a tool able to elaborate the *LCI_X features* interpreting the explicit and/or implicit product attributes.

The overall system architecture is illustrated in figure 4. The user interaction is based on an environment where the designer can use a Graphic User Interface to interactively sketch the architecture of the configurable product by selecting components from a library, to relate components to each others, to add requested and provided functions (also selected from a library) to the components, and set a priority order among components.

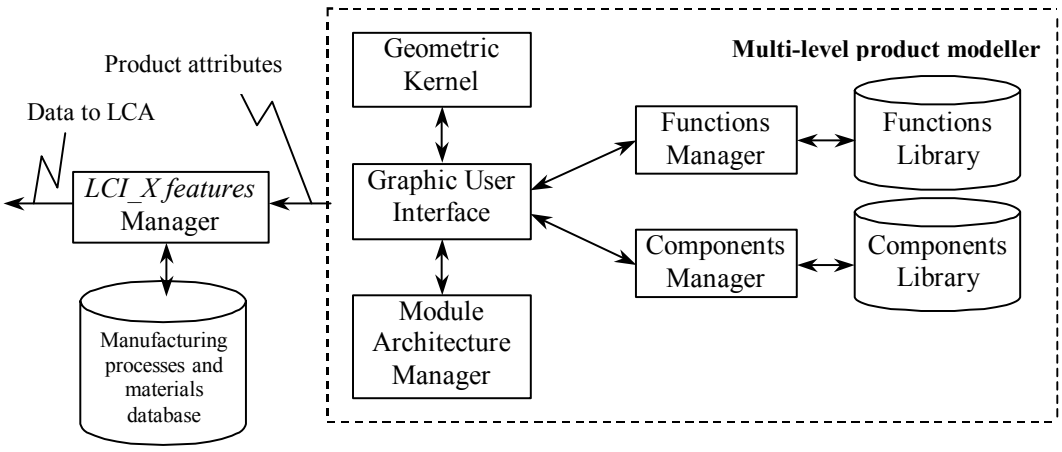


Figure 4. Main modules of the software system architecture

The function library and form features library can be extended with new functions and new relations by the user. The framework will then provide general functionality to manage the flow of information among components (on the basis of the priority order set by the user) and to ask to the component's engine to perform computation.

The main modules of the prototypal system architecture are:

Functions and Components Managers: they provide functionality to select components and functions from the libraries.

Module Architecture Manager: it provides functionality to manage the data structure of the module, made by components related to each others by means of requested and supplied functions, and to manage the activation of the components' engines.

Graphic User Interface: the interactive graphic environment where the user can sketch the configurable module, and where the module evolution is shown.

Geometric Kernel: the geometric engine charged to the geometric computation of the components shape.

The further necessary information (dimensioning rules, materials, technological rules, etc.) is implemented in two distinct knowledge bases (the design knowledge base and the technological knowledge base) linked to the design system. The definitive component configuration can be obtained through an iterative process because the preliminary configuration could be modified on the basis of constraints defined later in the configuration process. A first prototypal tool representing this part of the system has been implemented, the results are described in [11].

The product model implicit/explicit global and local attributes that are defined along the design process, are accessible from an external software module (*LCI_X features* Manager) which navigates the product model data structure and extracts the meaningful information for the LCA computation. This module organises, manipulates, and enriches the information to define the *LCI_X features* in order to be directly used as input for the LCA software. A customisable manufacturing processes and materials database provides further information to support the complete definition of the *LCI_X feature*. For example, if the *LCI_Material* value of a component is attributed, the *LCI_X feature* Manager selects within the database the more appropriate manufacturing processes and proposes them to the user, on the basis of component geometry parameters (rough or detailed). The user can correct and/or complete interactively the manufacturing operations. If the chosen process needs further information, for example the painting operation needs the component surface area, the software tool newly queries the product data structure to obtain necessary data. This part of the system is currently under implementation.

4.2 Example

Due to the complexity and vast amount of data of complete product life cycles, we initially limit our experimental work to the activities and processes related to the pre-use phase, with particular reference to the information related to raw materials, energy and manufacturing processes. The reason for this choice, which has no limiting impact on the approach itself nor on the framework under development, is related to the possibility of testing and evaluating our proposed approach already in its initial stage by using real product models.

We are going to consider different mechanical products typologies and different types of manufacturing processes. We briefly report the washing machine test case.

To perform the washing machine LCA analysis we need the following information:

1. Materials and weight of components.
2. Energy consumption linked to the washing machine production and manufacturing cycles.
3. Water consumption and characteristics of the plant for purification of the water supply.
4. Paint shop data and consumption.
5. Smokes composition and other emissions (chemical, acoustic, …).
6. Logistic to support the input, the output and the internal movements of components (vehicles, distances, …) and the related consumptions .
7. Industrial wastes: typologies, quantity, destination, ….

It is necessary a preliminary analysis to link such information to the manufacturing and assembly processes and, then, to the components characteristics, more or less detailed on the basis of the design phase. We have studied the product structure, identifying the main groups useful for LCI compilation (figure 5). Neglecting the documentation group that is not directly related to the manufacturing process, the other groups have been analysed to define the general product functional architecture. Each group has been associated with the corresponding high level *LCI_X features.* A further subdivision carries out the components typologies, materials and manufacturing process characteristics (figure 6, the oscillating group is reported).

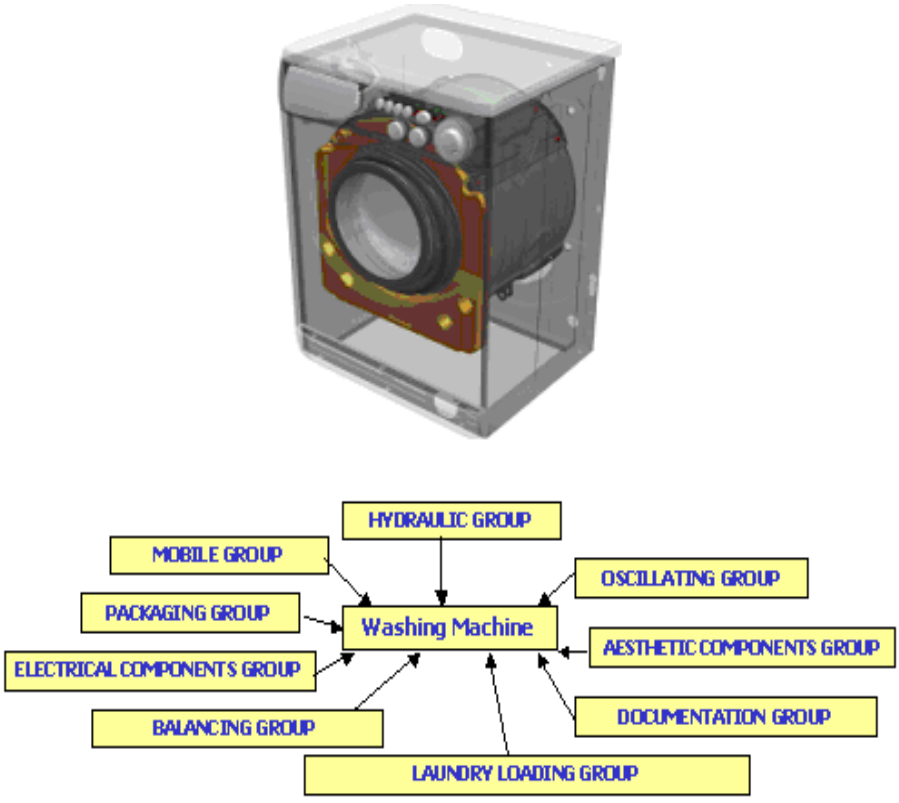


Figure 5. CAD model and meaningful washing machines groups useful for LCI definition

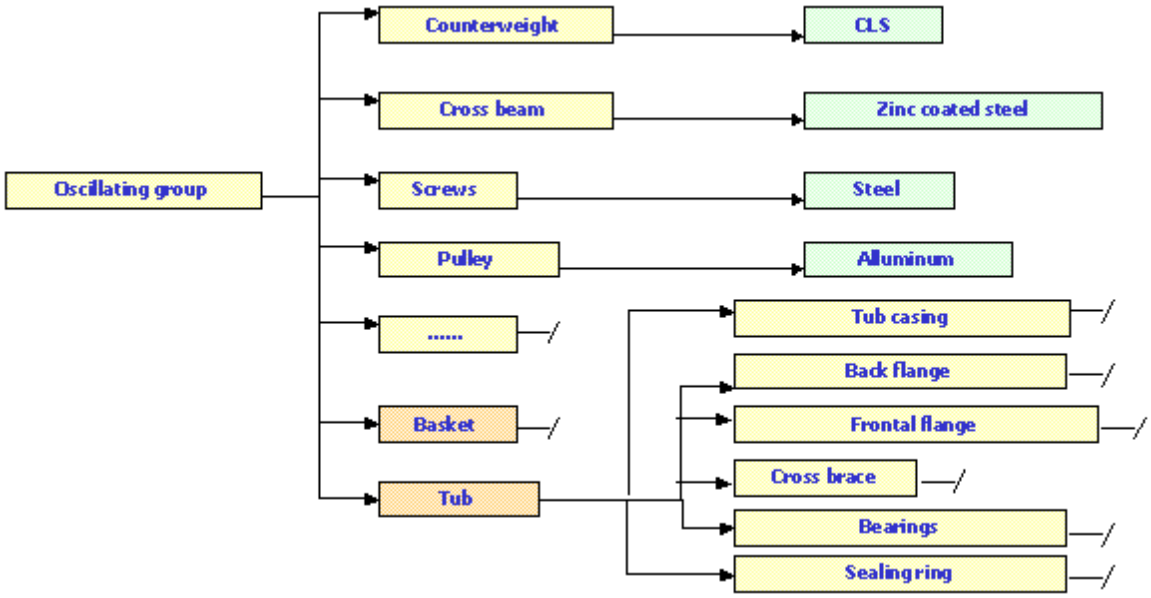


Figure 6. Oscillating sub group components and related materials

Such information is used to define the low level *LCI_X features*. Thus we analyse whether and where the required information is available, implicitly or explicitly in the product model.

For example if we consider the frontal frame, during the conceptual phase of a new product definition it can be defined as a part of a container used to separate the internal components from the environment, and also to structurally support the same components.

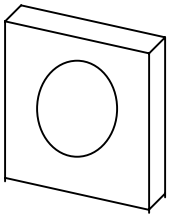
SCC graphical representation	SCC description	*LCI_X features*	*LCI_X feature* values	Attributes extracted from
	Bounding box	LCI_Name LCI_Material LCI_Amount_Material LCI_Manufaturing_ Process LCI_Assembly_Proces s	Frontal frame Steel 6 Kg Punching or injection moulding Null ……	Product model DS Product model DS Product model DS Proc database ……
	Detailed 3D geometric model	LCI_Name LCI_Material LCI_Amount_Material LCI_Manufaturing_ Process LCI_Assembly_Proces s	Frontal frame 10A50 Fe P 10 5,45 Kg Cutting Bending Punching Painting Drilling Welding ……	Product model DS Product model DS Product model DS Product model DS Product model DS Proc database Proc database Product model DS Product model DS ……

Figure 7. Frontal frame and the related *LCI_X features* at two different design phase (conceptual, upper, detailed, lower).

During the following phases it assumes a well defined aesthetic aspect and specific dimensional and morphological characteristics. We can extract progressively, from the multi-level product model, the different representations of the self configuring frontal frame,

extrapolating the information useful for the *LCI_X features* definition, as reported in figure 7, and, thus, we can perform product LCA analysis. Analogously for the remaining groups and/or components.

LCI-relevant information considered necessary, but which is usually missing in standard models, is included thanks to the definition and integration of suitable procedures. A preliminary prototypal software system allowing the integration of further information navigating a dedicated processes database (figure 8).

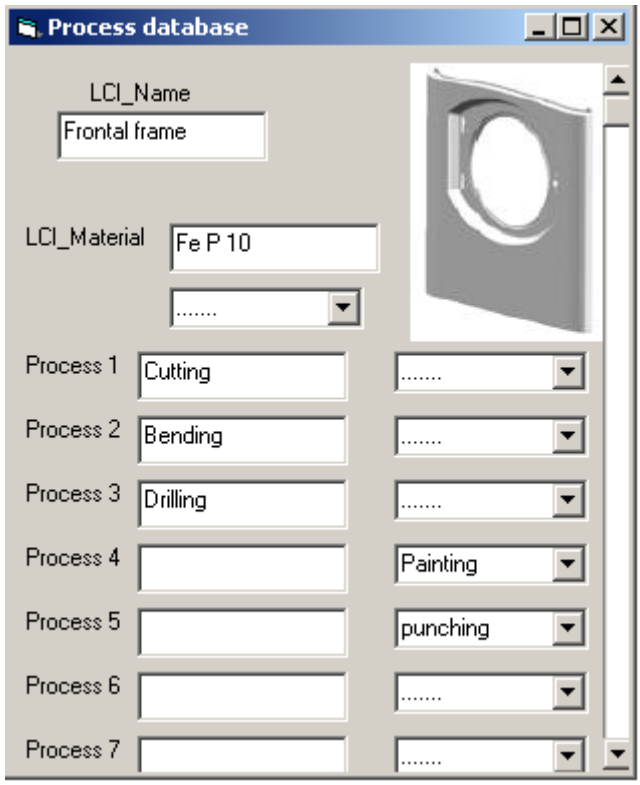


Figure 8. Processes database user interface

It is evident that the current results are still partial and they show that the research project is in embryo, but the framework developed demonstrated to be a good and robust base to continue the study both from the theoretical viewpoint and from the practical tools implementation viewpoint.

5 Summary and conclusions

This paper describes a preliminary study of a methodology which aims to support the execution of LCA analysis in the early phases of design process. The current work has been dedicated to the definition of a product model framework, useful to collect the design information in a multi-level structure. Such a structure, based on the concept of self-configuring components, maps the design phases and the related generated information with a multi-level product model representation. The SCCs contain rules context-sensitive to configure itself on the basis of required function; it can be used to collect information useful to perform the LCA at different levels of detail. *LCI_X features* have been defined to organise and transmit the information from product model to LCI database.

The future work will be dedicated to make a more extensive, accurate and detailed classification of information to be collected within the *LCI_X* features (properties, attributes and values), compatibly with the informational content of the specific design phase. On the other hand, we are working on a more robust implementation of the multi-level product modeller and the *LCI_X* features Manager to generate the product design and, simultaneously, the data useful for LCA analysis. Finally, having applied the methodology to mechanical products, commercial LCA software packages showed an excessive generality in the representation of main manufacturing processes (injection moulding, sheet-metal punching, …), thus, a further future goal will be the improvement of such tools defining detailed processes to be more efficiently interfaced with the *LCI_X features*.

References

[1] Rebitzer, G., et al., "Life Cycle Assessment – Part 1: Framework, Goal and scope Definition, Inventory Analysis and Applications", Environment International, Vol.30, 2004, pp.701-720.

[2] Hur, T., et al., "Simplified LCA and Matrix Methods in Identifying the Environmental Aspects of a Product System", Journal of Environmental Management, 2005, pp.1-9.

[3] Sun, J., et al., "Design For Environment: Methodologies, Tools and Implementation", Journal of Integrated Design and Process Science, Vol.7, No.1, March 2003, pp.59-75.

[4] Van Hemel, C., Cramer, J., "Barriers and Stimuli for Eco-Design in SMEs", Journal of Cleaner Production, Vol.10, 2002, pp. 439-453.

[5] Bhander, G.S., Hauschild, M., McAloone, T., "Implementing Lifec Cycle Assessment in Product Development", Environment Progress, Vol.22, No.4, 2003, pp.255-267.

[6] Dewulf, W., Duflou, J., "Simplifying LCA using Indicator Approaches – A Framework", Proc. of CIRP Seminar on Life Cycle Engineering, Copenaghen, May 2003.

[7] Rosemann, B., Meerkamm, H., "Eco-Design: Make It Happen by an Environmental Innovative Product Design", Proceedings of International Design Conference – Design 2004, Dubrovnik, 18-21 May 2004.

[8] Ernzer, M., Birkhofer, H., "From Product Ideas to Sustainable Products – Life Cycle Design Suitable for Product Designer", Proceedings of EDC 02, London 2002, pp.647-656.

[9] Nawata, S., Aoyama, T., "Life-Cycle Design System for Machined Parts – Linkage of LCI Data to CAD/CAM Data", Proceedings of 2nd International symposium on Environmentally Conscious Design and Inverse Manufacturing, ECODESIGN 2001, Tokyo 11-15 December, 2001, pp.299-307.

[10] Otto, H, Mandorli, F., Germani, M., Kimura, F., "Integration of Solid Modeling with LCA using Feature Technology: an Application for SME", CD-ROM Proceedings of ECODESIGN 2003, Tokyo 7-9 December, 2003.

[11] Germani, M., Mandorli, F., "Definition of Self-Configuring Components for Product Variant Development", AI EDAM special issue on Platform Product Development for Mass Customisation, Vol.18, No.1, 2004.

[12] Hirtz, J., Stone, R., McAdams, D., Szykman, S., Wood, K., "Evolving a Functional Basis for Engineering Design", CD-ROM Proceedings of ASME 2001 Design Engineering Technical Conference DETC/DTM'01, Pittsburgh 9-12 September, 2001.

Michele Germani
Polytechnic University of Marche, Faculty of Engineering
Department of Mechanics
Via Brecce Bianche, 60100 Ancona, Italy
Phone: +39-71-2204969
Fax: +39-71-2204801
E-mail: m.germani@univpm.it
Web site: www.dipmec.univpm.it