

CONFIGURATION GRAMMAR-BASED DESIGN APPROACH FOR PRODUCT FAMILY MODELING IN ADVANCED CAD SYSTEMS

Eugeniu-Radu Deciu^{1,2}, Egon Ostrosi¹, Michel Ferney¹, Marian Gheorghe²

¹ Université de Technologie de Belfort-Montbéliard, Laboratoire M3M

² University "Politehnica" of Bucharest, Manufacturing Department

ABSTRACT

The effective modelling of configurable products must deal with the problem of generation of the new and innovative products. This modelling must handle the complex problem of representation of the configurable products on the one hand, and of learning new configurable products on the other hand. However, in advanced CAD systems, there is no formal representation to support the modelling of the configurable products. This paper proposes and develops a grammar-based design approach to support the computer-aided-design for product configuration. The proposed approach is based on the hypothesis that a product has a final structure that is the result of an ideal evolution from a set of significant structures and on the properties of configurable products. The application is directed to a gear pump family modelling.

Keywords: product family, design for configuration, product configuration, product modelling, computer-aided-design, grammar representation

1 INTRODUCTION

Design of configurable product family or *design for configuration* has emerged as an efficient tool to solve the new challenges of a constantly changing market [3], [4]. Design for configuration is the process which generates a set of *product configurations* based on a *configuration model* and is characterized by a configuration task [10], [14], [15], [17]. The *configuration task* then consists in finding the configuration of a product by defining the relations between its components in order to satisfy a set of specifications and a set of constraints imposed on the product [10], [14], [15], [16].

The *product modeling* is an essential aspect in the product family design for configuration [7], [15], [16]. The effective modeling of a configurable product family must be capable to represent the complex relationship between the components of a product on the one hand, and the members of the family, on the other hand [5], [13]. This modeling must deal with the problem of *generation* and *derivation* of the different products, and thus carry out the variety of the *new* and *innovative* products [1], [8], [12]. Furthermore, this modeling must handle the complex problems of knowledge extraction and representation of the configurable products on the one hand, and of learning new configurable structures on the other [8].

However, in advanced CAD systems, there is no formal representation to support the modeling of the configurable product [7], [8]. Grammars can be considered as formal powerful tools to represent the strong structural relationship in the configurable products [2], [9], [11], [13], [18]. This paper proposes and develops a grammar-based design approach to support the computer-aided-design for product modeling. Two interrelated questions are taken into account:

1. What are the properties of configurable products? (*Properties extraction of configurable products*)
2. How can we develop generative grammars suitable to handle the properties of configurable products? (*Development of Configuration Grammars for Product Family Modeling*)

This paper discusses the problem of product configuration. In the second section, using the hypothesis of virtual product structure generation and the properties of configurable products, a structural grammar for configuration based on features is proposed to model product families. The third section presents an application of this grammar to a gear pump family modelling in CAD systems. Lastly, conclusions and perspectives for future research are presented.

2 CONFIGURATION GRAMMAR-BASED DESIGN APPROACH

During the design process in CAD environment, the mechanical system, i.e. part, module, product, or product family, to be designed evolves in structure and characteristics up to when the design is accomplished. In fact, the transition from the intended product to be designed up to the designed product can be seen as a progressive, evolutive and non-determinist process, where each step is characterised by knowledge acquisition. Also, starting from the initial phase and going to the final phase, the product goes through intermediate phases of evolution and different representations, which become closer to the final virtual product at the end of the process. Based on these observations, we state the following hypothesis:

The final structure of a virtual product is the result of the evolution of a significant structures set.

This hypothesis suggests implicitly a mechanism of generation of the configuration structures. Moreover, this hypothesis is valid and hold for each level of the mechanical system: *configuration feature-part-module-product*. So, this mechanism represents the evolution process of the structures (i.e. configuration features-parts-modules-products), from an initial phase up to the final phase, from simple to complex.

2.1 Properties extraction of configurable products

When a mechanical system is decomposed into its components and when its associated couplings are specified, in fact we are determining and arranging its elements; that is, we are configuring the mechanical system. From the engineering design point of view, the *feature-component-module-product* relationships are adequate structural means for a general product representation. Since such means are **recursive**, any *proper granularity level of representation* must be introduced to asses design for configuration possibility. Let us note with *structure(i)* being a structure of level *i* of the precedent relationship. For instance a *component* is a *structure(2)* level. Then based on the **recursive** representation and generation of a configurable product family the following properties are defined:

- **Property 1** (significant structures): The significant structures(*i*) are: the *primitive structures* or *terminals*, the *intermediate structures* or *non-terminals* and the *final structure*.
- **Property 2** (attaching elements): Each structure(*i*) is provided with a set of particular elements, called *attaching elements* [7], [8];
- **Property 3** (generation of new structures, joint elements and tie elements): From the interconnection between structures (*primitive or intermediate*) evolves a higher level structure to these ones [7], [8]. The attaching elements produce the joint elements and the tie elements [7]. The consequences of the property 3 are:
 - *Recurrence* – refers to the possibility of adding (repeating) several times the same structure(*i*) in order to construct the product configuration
 - *Addition* – refers to the possibility of a structure(*i*) to be added to the existent product structure (of the product family);
 - *Removing* – refers to the possibility of a structure(*i*) to be removed from an existent product structure (in a family);
 - *Replacement or swapping* – refers to the possibility of a structure(*i*) of the product structure to be replaced by another structure(*i*);
- **Property 4** (geometric and topologic constraints): The interconnection between structures can occur if and only if the structures satisfy some conditions defined on the geometric and topological domains (i.e. the geometric and topologic constraints). For instance, *spatial orientation* – indicates the relative spatial orientation of each structure in the product structure.

2.2 Structural Grammars for Configuration based on Features

Grammars are powerful tools to represent the structural relationships in product configuration. In this paper we propose a *grammar for product family configuration* that is developed on the concept of *configuration feature grammar*, initially proposed by Ostrosi et Ferney [8].

We call this grammar *Structural Grammar for Configuration based on Features (SGCF)*. The proposed grammar for configuration works on a graph representation of structures, where the nodes represent the structure elements (i.e. configuration features, parts, modules, products) and the arcs the relationships between them.

Our grammar approach is based on two main concepts: *configuration features* and *configuration connections*.

Based on the property 2 (*attaching elements*), the significant structures possess a set of attaching elements called *configuration features*. The configuration features are a special category of form features. There are different definitions of form features, such as: “a generic shape which carries some engineering meaning” [19]; “a carrier of product information which may aid design or communication between design and manufacturing, or between other engineering tasks”. [20].

So, in our acceptance, a *configuration feature* or an *attaching feature* represents a geometric entity that is defined by its shape and technological characteristics, typically represented by a set of topologically associated faces. These faces are grouped together, in a single feature, with a *connection intent function*. The *connection intent function* defines what type of connection is intended to be obtained from the features association. So, the configuration features are functional intent carriers.

Moreover, the configuration features belonging to different structures (parts, modules or product) enables them to connect each other. According to the property 3 (*joint and tie elements*), the connection between two or more structures is made on two levels of: *joints* and *ties*, and consequently there are two types of configuration features: the *joint features* and the *tie features*.

A *configuration connection* is expressed as a high-level relation between two or more configuration features that defines a geometric entity of a superior level than the configuration features. The connection relation defines a set of low-level constraints and associates together two or more configuration features from distinct parts.

According to the previous definition of configuration features, there are two levels of connections: *joint connections* and *tie connections*.

Let be two domains, the domain of primitive *configuration features* F and the domain of non-(primitive *configuration connections* C . Each domain defines a collection of elements: the primitive *configuration features* set and non-primitive *configuration connections* set (example: figure 1). Then, based on the previous two sets and a configuration language, a set of configuration structures can be generated.

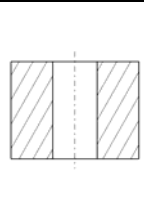
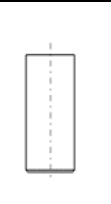
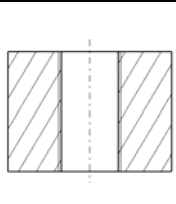
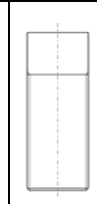
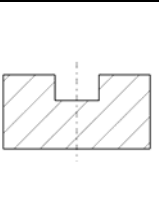
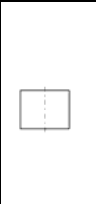
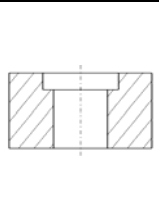
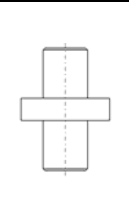
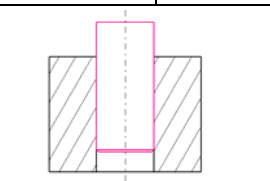
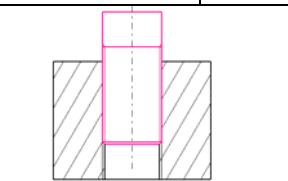
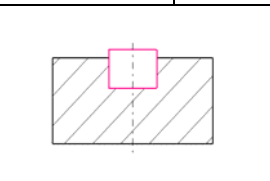
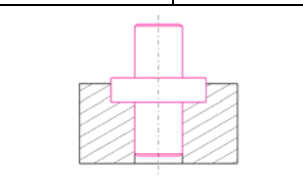
							
<i>Through Hole</i>	<i>Cylindrical Neck</i>	<i>Threaded Hole</i>	<i>Thread</i>	<i>Through Slot</i>	<i>Rib</i>	<i>Stepped Through Hole</i>	<i>Shoulder</i>
							
<i>Cylindrical (Through) Connection</i>		<i>Threaded Connection</i>		<i>Through Slot Connection</i>		<i>Cylindrical Stepped Connection</i>	

Figure 1 Example of configuration features and configuration connections

A *configuration language* describes the generation of configuration structures, joint elements and tie elements [9]. So, a configuration grammar based on features provides the formal and generic description for this configuration language.

Then, the *structural grammar for configuration based on features*, **SGCF**, is defined as the 8-tuple:

$$G = \{V_{structure}^T, V_{joint-connexion}^T, V_{structure}^N, V_{joint-tie\ features}^N, S, \nabla, \Lambda, P\} \quad (1)$$

where:

$V_{structure}^T = \{a, b, c, \dots\}$ is a finite, non-empty set of primitive significant structures called the *terminal vocabulary of configuration structures*. Where a, b, \dots , are terminal configuration structures.

$V_{structure}^N = \{A, B, \dots, S\}$ is a finite, non-empty set of non-primitive significant structures called the *non-terminal vocabulary of configuration structures*. Where A, B, \dots , are non-terminal configuration structures and S is a special non-terminal called *start symbol*.

$V_{joint-tie\ features}^T = \{0, feature_i, \dots\}$ is a finite, non-empty set of *configuration features* called the *terminal vocabulary of joint and tie features*. For example: $feature_i = \{through\ hole, cylindrical\ neck, threaded-through\ hole, thread, stepped-through\ hole, shoulder, slot, \dots\}$.

$V_{joint-tie\ connections}^N = \{O, \dots, connection_j, \dots, \nabla, \Lambda\}$ is a finite, non-empty set of configuration connections between the configuration features called the *non-terminal vocabulary of joint and tie connections*. For example: $connection_j = \{cylindrical\ through\ connection, threaded\ connection, slot\ connection, cylindrical\ stepped\ connection, \dots\}$;

S, ∇, Λ are the starting symbol of configuration structures, the starting symbol of joint connections and respectively of tie connections.

$P : \left\{ \begin{bmatrix} \alpha \\ \Gamma_\alpha \\ \Delta_\alpha \end{bmatrix} \rightarrow \begin{bmatrix} \beta \\ \Gamma_\beta \\ \Delta_\beta \end{bmatrix} \right\}$ is a finite, non-empty set of productions rules.

The following conditions must be held concerning the terminal vocabulary and the non-terminal vocabulary, of configuration structures and respectively of joint features-tie features:

$$V_{structure}^T \cap V_{structure}^N = \emptyset \text{ and } V_{joint-connexion}^T \cap V_{joint-connexion}^N = \emptyset,$$

$$(V_{structure}^T \cup V_{structure}^N) \cap (V_{joint-connexion}^T \cup V_{joint-connexion}^N) = \emptyset.$$

To define the generation of new structures in a non-ambiguous way, we have to state a set of conditions on the production rules. The interconnection between structures can occur if and only if the structures satisfy some conditions defined on the geometric and topological domains. These conditions are represented by the geometric constraints and topological constraints. For instance, *spatial orientation* – indicates the relative spatial orientation of each structure in the product structure.

The geometric and topological constraints are imposed at each level of production rules. This means that the grammar productions rules must meet obligatory conditions or constraints before being applied.

Then, a *conditional production rule* C is defined as follows: $\begin{bmatrix} \alpha \\ \Gamma_\alpha \\ \Delta_\alpha \end{bmatrix} \xrightarrow{C} \begin{bmatrix} \beta \\ \Gamma_\beta \\ \Delta_\beta \end{bmatrix}$, where $C = \begin{bmatrix} C_{\alpha \rightarrow \beta} \\ C_{\Gamma_\alpha \rightarrow \Gamma_\beta} \\ C_{\Delta_\alpha \rightarrow \Delta_\beta} \end{bmatrix}$ are

semantic conditions associated to each level of a productions rule.

2.3 Grammar-based applications

Example 1. Let us consider two parts: a *plate* and a *pin*, represented in the figure 2. The two structures possess some *configuration features*, namely: *ThroughHole* for the *plate* part and *CylindricalNeck* for the *pin* part. So, the two structures (parts) can connect each other through these attaching features, in order to form and generate a new structure of a higher level that is called, in this case, the *plate-pin*.

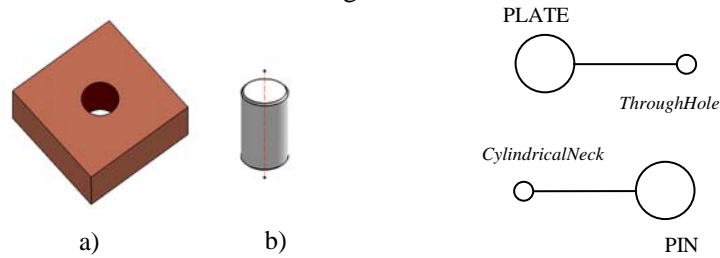


Figure 2 The plate (a) and the pin (b) parts and their graph representation (c)

The production that generates the *plate-pin* structure has the following graph representation form (figure 3):

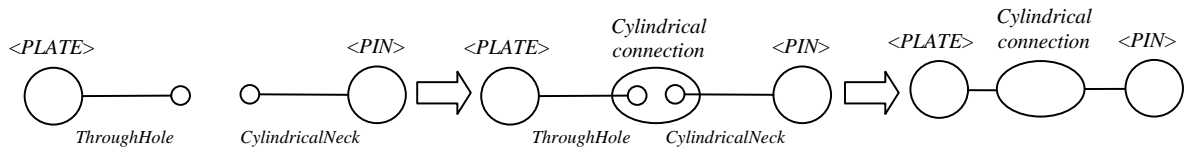


Figure 3 Graph representation of the production that generates a new structure

Similarly to the *structure level*, on the *configuration features level* (*joint* or *tie*), the features connect each other and generate higher level structures that we call *configuration connections*. In our example this type of connections is represented by the *cylindrical connection* structure. These configuration connections are *non-terminal* structures which result from the connection of two or more attaching features.

The corresponding formal production to generate the *plate-pin* structure is:

$$P : \left[\begin{array}{l} \langle PLATE - PIN \rangle \rightarrow \langle PLATE \rangle \langle PIN \rangle \\ \langle CylindricalConnection \rangle \rightarrow \langle ThroughHole \rangle \langle CylindricalNeck \rangle \\ \langle \Lambda \rangle \rightarrow \langle ThroughHole \rangle \langle CylindricalNeck \rangle \end{array} \right] \begin{array}{l} \text{Structure level} \\ \text{Joint level} \\ \text{Tie level} \end{array} \quad (2)$$

The start symbol Λ present on the third line of the configuration matrix (2) means that the *tie level* is saturated.

The graphical representation of this formal production is (figure 4):

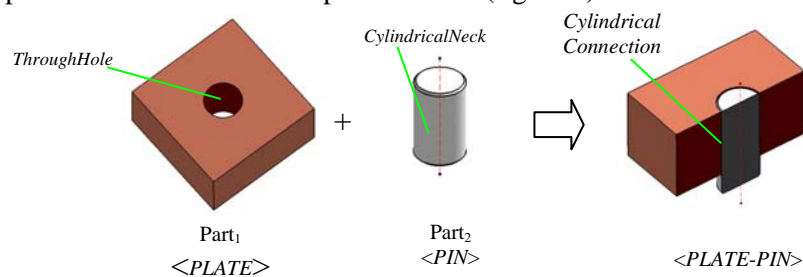


Figure 4 Evolved structure generated from the interaction between two primitive structures through their connecting features

As it can be seen, the configuration features used in this example are joint features, because there is no possibility to use them in subsequent connections. So, there is no tie feature available in the resulted structure.

Example 2. Let us consider two parts: a *plate* (similar to that of example 1) and a *shaft*, represented in figure 5. The two structures possess some *configuration features*, namely: the *joint features* *SteppedThroughHole* (for the *plate* part) and the *LowerShoulder* (for the *shaft* part) and the *tie features* *UpperShoulder* (for the *shaft* part). The configuration features specific to both structures are a composition, a set of regrouped faces with configuration function intent: the *SteppedThroughHole* is composed of an internal cylindrical face and a flat face; the *LowerShoulder* is composed of an external cylindrical face and a step (flat) face; and the *UpperShoulder* is analogue to the *LowerShoulder* feature. So the two structures (parts in this case) can interact through these attaching features, in order to form and generate a new structure of a superior level that we call *plate-shaft*.

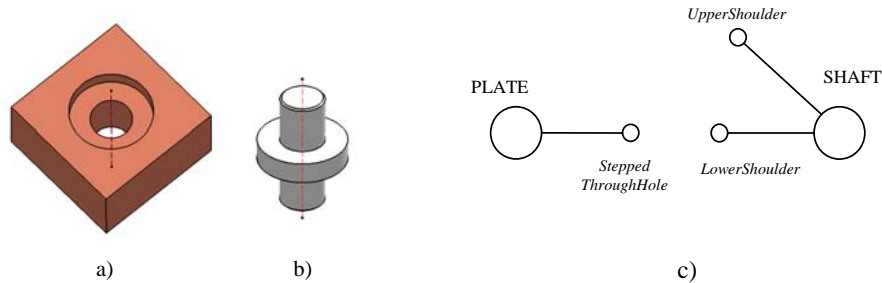


Figure 5 The plate (a) and the shaft (b) parts and their graph representation (c)

The production that generates the *cylindrical-flat connection* structure is the following (3):

$$P : \left[\begin{array}{l} \langle \text{PLATE} - \text{SHAFT} \rangle \rightarrow \langle \text{PLATE} \setminus \text{SHAFT} \rangle \\ \langle \text{CylindricalSteppedConnection} \rangle \rightarrow \langle \text{SteppedThroughHole} \setminus \text{LowerShoulder} \rangle \\ \langle \text{UpperShoulder} \rangle \rightarrow \langle \emptyset \setminus \text{UpperShoulder} \rangle \end{array} \right] \begin{array}{l} \text{Structure level} \\ \text{Joint level} \\ \text{Tie level} \end{array} \quad (3)$$

The graph representation of this rule is (figure 6):

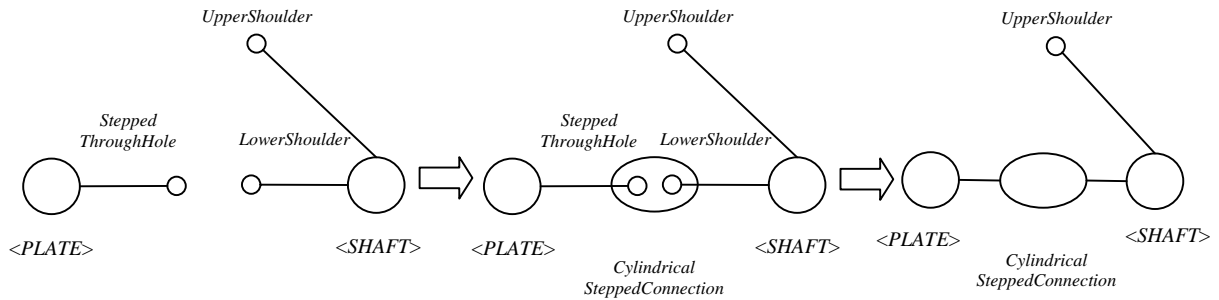


Figure 6 Graph representation of the production that generates the new structure

As we can see, in addition to the previous example there is still available an attaching feature on the *Plate-Shaft* structure that is the *UpperShoulder* feature. This is a typical example of *tie features*. The tie features belonging to a part, *shaft* in our example, are used subsequently to connect the current structure with other available structures of the product.

The graphical representation of the production rule from figure 5 is the following (figure 7):

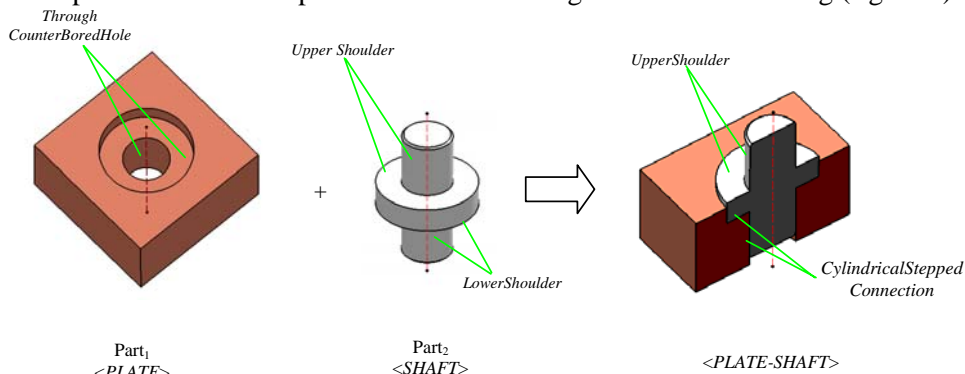


Figure 7 Evolved structure generated from the interaction between two primitive structures through their connecting features

3 GRAMMAR-BASED GEAR-PUMP FAMILY MODELLING

The *external gear pumps* are the most widely used pumps in hydraulic systems, due to their simplicity, reliability, and very high power ratings. External gear pumps are fixed displacement and are used in the hydrostatic units mobiles, equipments for transport, machine tools and other applications with a large number of variants and configurations. The structure of the gear pump is summarized in figure 8 and is composed of the following main components [6]:

- | | |
|------------------|--------------------|
| 1) – Flange | 7) – Screws |
| 2) – Body | 8) – Dowel pin |
| 3) – Thrust seal | 9) – Lip seal |
| 4) – Bearings | 10) – Woodruff-key |
| 5) – Gears | 11) – Washer |
| 6) – Rear cover | 12) – Nut |

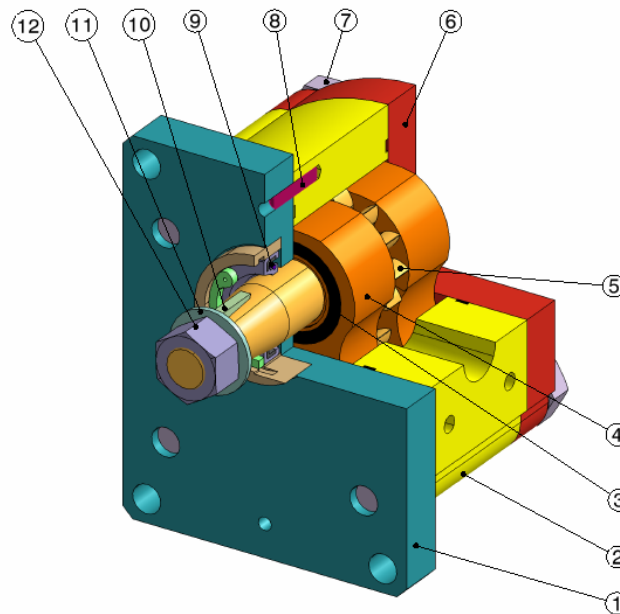


Figure 8 The gear pump generic structure

In this application, the purpose is to represent the external gear pump structure by the help of configuration grammars: first to generate the structure of a single gear pump, and then to generate a family of gear pumps.

The productions of the configuration features grammar are inferred in order to generate external gear pump structures. First, the model and the generation of a gear pump structure is presented. Based on this generation, we derive and generate a new family of gear pumps.

Next, we present the steps of generation of the *bearing-shafts* pump module by inferring grammar productions that use the properties 1, 2, 3 (*addition*) and 4 (*spatial orientation*) (see section 2.1).

The first production of the grammar defines the connection between the first bearing of the pump ($BEARING_1$ or BE_1) and the driver shaft ($SHAFT_1$). The connection between the two parts is made through their respective set of attaching features, highlighted in red color (figure 9): $\langle SteppedThroughHole_1 \rangle$ and $\langle SteppedThroughHole_2 \rangle$ for the $BEARING_1$ part and $\langle LeftShoulder_1 \rangle$, $\langle RightShoulder_1 \rangle$ for the $SHAFT_1$ part. For the sake of clarity, we are indicating, in figure 9, only the attaching features that participate in the generation of the *bearing-shafts* pump module (structure).

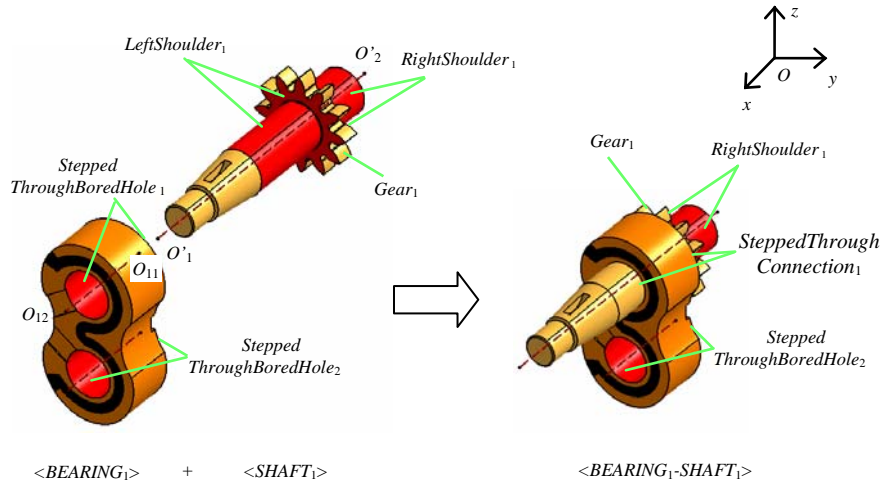


Figure 9 The first production to generate the <bearings-shafts> module

The production that generates the $bearing_1$ - $shaft_1$ structure is the following (4):

$$P_1 : \left[\begin{array}{l} \langle BEARING_1 - SHAFT_1 \rangle \rightarrow \langle BEARING_1 \rangle \langle SHAFT_1 \rangle \\ \langle SteppedThroughConnection_1 \rangle \rightarrow \langle SteppedThroughHole_1 \rangle \langle LeftShoulder_1 \rangle \\ \langle RightShoulder_1 \rangle \rightarrow \langle 0 \rangle \langle RightShoulder_1 \rangle \\ \langle SteppedThroughHole_2 \rangle \rightarrow \langle SteppedThroughHole_2 \rangle \langle 0 \rangle \\ \langle Gear_1 \rangle \rightarrow \langle 0 \rangle \langle Gear_1 \rangle \end{array} \right] \begin{array}{l} \text{Structure level} \\ \text{Joint level} \\ \text{Tie level} \end{array} \quad (4)$$

$$C_1 : \left[\begin{array}{l} \langle BEARING_1 - SHAFT_1 \rangle \rightarrow \langle BEARING_1 \rangle \langle SHAFT_1 \rangle \\ \langle O_{12}O_{11} \equiv O'_1O'_2 \rangle \rightarrow \langle O_{12}O_{11} \rangle \langle O'_1O'_2 \rangle \\ \langle \angle(O_{12}O_{11}, O'_1O'_2) : (0,0,0) \rangle \rightarrow \langle O_{12}O_{11} \rangle \langle O'_1O'_2 \rangle \\ \langle O \rangle \rightarrow \langle 0 \rangle \langle 0 \rangle \end{array} \right] \begin{array}{l} \text{Structure level} \\ \text{Joint level} \\ \text{Tie level} \end{array} \quad (5)$$

The equation (5) specifies the spatial constraints that must be held in order to generate a valid structure. For example, the directions of $O_{12}O_{11}$ and $O'_1O'_2$ axis must be coincident and the vectors $\overrightarrow{O_{12}O_{11}}$ and $\overrightarrow{O'_1O'_2}$ must have the same direction. Similar spatial constraints were defined for the rest of grammar productions.

Below, we give as example the graph representation of the first production (figure 10). The rest of the graph productions are built on similar basis.

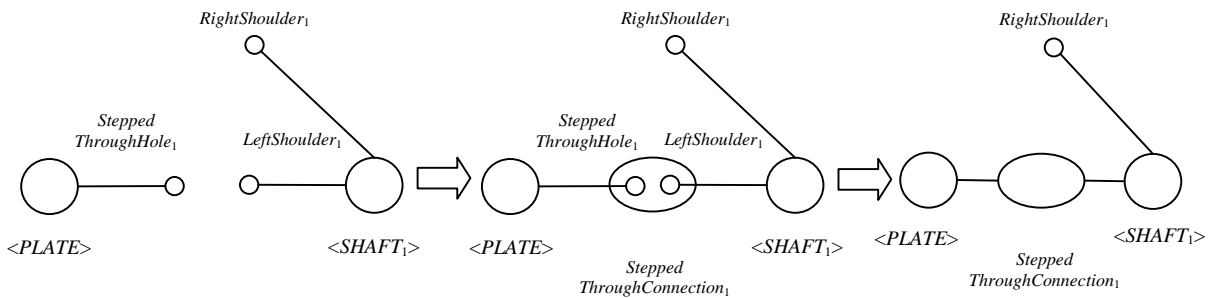


Figure 10 Graph representation of the first production to generate the <bearings-shafts> module

The second production defines the connection between the resulted non-terminal structure $\langle BEARING_1-SHAFT_1 \rangle$ and the terminal structure of the driven shaft, $shaft_2$, to define the generation of a new structure, $\langle BEARING_1-SHAFT_1-shaft_2 \rangle$, composed of the first bearing and the pair of shafts (figure 11):

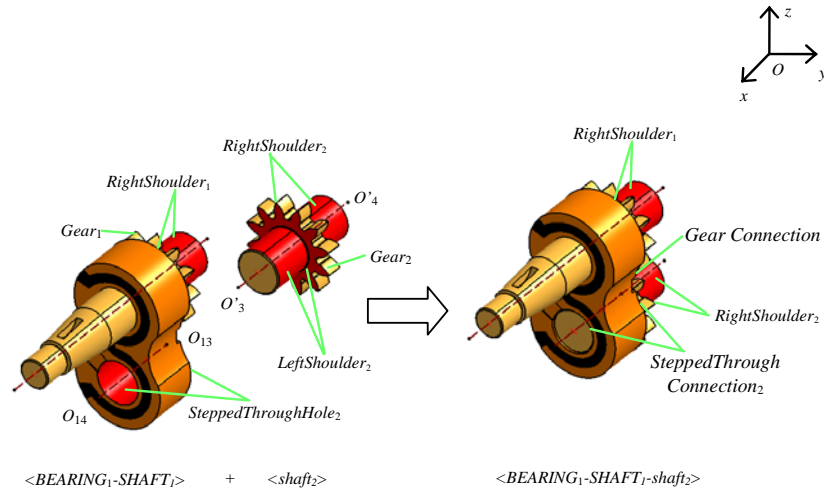


Figure 11 The second production to generate the $\langle bearings-shafts \rangle$ module

Formally, this production has the following form (6):

$$P_2 : \left[\begin{array}{l} \left[\langle BEARING_1 - SHAFT_1 - shaft_2 \rangle \rightarrow \langle BEARING_1 - SHAFT_1 \rangle \langle shaft_2 \rangle \right] \\ \left[\langle SteppedThroughConnection_2 \rangle \rightarrow \langle SteppedThroughHole_2 \rangle \langle LeftShoulder_2 \rangle \right] \\ \left[\langle GearConnection \rangle \rightarrow \langle Gear_1 \rangle \langle Gear_2 \rangle \right] \\ \left[\langle RightShoulder_1 \rangle \rightarrow \langle RightShoulder_1 \rangle \langle 0 \rangle \right] \\ \left[\langle RightShoulder_2 \rangle \rightarrow \langle 0 \rangle \langle RightShoulder_2 \rangle \right] \end{array} \right. \begin{array}{l} \text{Structure level} \\ \text{Joint level} \\ \text{Tie level} \end{array} \quad (6)$$

To connect the second bearing to the previously formed structure, the first one has to be rotated with an angle of 180° along the Oz and Ox axis (according to figure 12). Next, we give the necessary spatial conditions to deal with spatial rotation of the second bearing around the $\langle BEARING_1-SHAFT_1-shaft_2 \rangle$ structure.

So, we have applied the following steps for the rotation strategy: first, we rotate the $\langle BEARING_2 \rangle$ with 180° around Oz axis (7). This is done by imposing a coincidence condition between the direction of $O'_1O'_2$ and $O_{24}O_{23}$ axis. In the second line of joints level we specify the rotation angle, i.e. 180° . We consider the positive sign for the anticlockwise rotation.

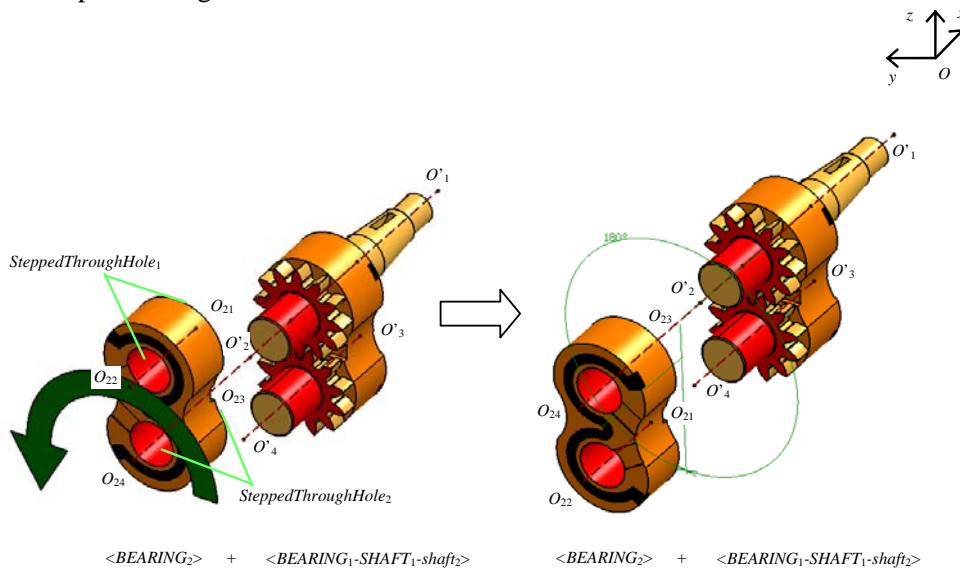


Figure 12 Spatial rotation conditions to generate the $\langle bearings-shafts \rangle$ module

$$C_{31} : \left[\begin{array}{l} \langle \langle BEARING_2 - BE_1 - SHAFT_1 - shaft_2 \rangle \rangle \rightarrow \langle \langle BEARING_2 \rangle \langle BE_1 - SHAFT_1 - shaft_2 \rangle \rangle \\ \left[\begin{array}{l} \langle O_{24}O_{23} \equiv O'_1O'_2 \rangle \rightarrow \langle O_{24}O_{23} \rangle \langle O'_1O'_2 \rangle \\ \langle \angle(O_{24}O_{23}, O'_1O'_2) : (0,0,180) \rangle \rightarrow \langle O_{24}O_{23} \rangle \langle O'_1O'_2 \rangle \\ \langle O_{22}O_{21} \rangle \rightarrow \langle O_{22}O_{21} \rangle \langle 0 \rangle \end{array} \right] \end{array} \right] \quad (7)$$

In the second step, the set of constraints relative to the rotation of $\langle BEARING_2 \rangle$ with 180° around the Ox axis are specified: the first line of joints level conditions is the same as previous; the second line of joints conditions specifies the second rotation angle around the Ox axis and the third line specifies the tie level condition, that is the coincidence condition between the direction of $O_{22}O_{21}$ and $O'_3O'_4$ axis (8).

$$C_{32} : \left[\begin{array}{l} \langle \langle BEARING_2 - BE_1 - SHAFT_1 - shaft_2 \rangle \rangle \rightarrow \langle \langle BEARING_2 \rangle \langle BE_1 - SHAFT_1 - shaft_2 \rangle \rangle \\ \left[\begin{array}{l} \langle O_{24}O_{23} \equiv O'_1O'_2 \rangle \rightarrow \langle O_{24}O_{23} \rangle \langle O'_1O'_2 \rangle \\ \langle \angle(O_{24}O_{23}, O'_1O'_2) : (-180,0,0) \rangle \rightarrow \langle O_{24}O_{23} \rangle \langle O'_1O'_2 \rangle \\ \langle O_{22}O_{21} \equiv O'_3O'_4 \rangle \rightarrow \langle O_{22}O_{21} \rangle \langle O'_3O'_4 \rangle \end{array} \right] \end{array} \right] \quad (8)$$

Now, the two structures satisfy the spatial orientation conditions and can be connected. The connection is made through the set of available attaching features: $\langle SteppedThroughHole_1 \rangle$, $\langle SteppedThroughHole_2 \rangle$ for the $BEARING_2$ part; and $\langle RightShoulder_1 \rangle$, $\langle RightShoulder_2 \rangle$ for the $BE_1-SHAFT_1-shaft_2$ structure: (figure 13).

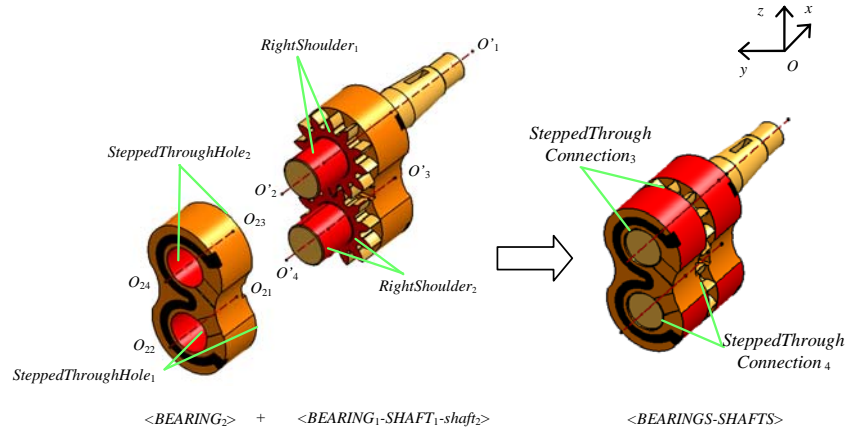


Figure 13 The third production to generate the $\langle bearings-shafts \rangle$ module

So, the formal production to connect the $BEARING_2$ part and $BE_1-SHAFT_1-shaft_2$ structure is (9):

$$P_4 : \left[\begin{array}{l} \langle \langle BEARINGS - SHAFTS \rangle \rangle \rightarrow \langle \langle BEARING_2 \rangle \langle BE_1 - SHAFT_1 - shaft_2 \rangle \rangle \\ \left[\begin{array}{l} \langle \langle SteppedThroughConnection_3 \rangle \rangle \rightarrow \langle \langle SteppedThroughHole_2 \rangle \langle RightShoulder_1 \rangle \rangle \\ \langle \langle SteppedThroughConnection_4 \rangle \rangle \rightarrow \langle \langle SteppedThroughHole_1 \rangle \langle RightShoulder_2 \rangle \rangle \end{array} \right] \\ \left[\begin{array}{l} \langle \langle CylindricalFace_1 \rangle \rangle \rightarrow \langle 0 \rangle \langle \langle CylindricalFace_1 \rangle \rangle \\ \langle \langle CylindricalFace_2 \rangle \rangle \rightarrow \langle \langle CylindricalFace_2 \rangle \rangle \langle 0 \rangle \end{array} \right] \end{array} \right] \quad \begin{array}{l} \text{Structure level} \\ \text{Joint level} \\ \text{Tie level} \end{array} \quad (9)$$

As it can be seen in the figure 13, the generated structure $\langle BEARINGS-SHAFTS \rangle$ is not saturated. It still posses a set of attaching features, highlighted in red color, which enable the structure to interact with other structures of the product, and so, to develop toward higher level structures.

In the same way, the rest of grammar production rules were inferred to generate the gear pump structure: 1) the first of this rules is used to generate of the $\langle FLANGE-BODY \rangle$ structure (module); 2) the second production is used to connect the previous generated structure $\langle BEARINGS-SHAFTS \rangle$ to the latter $\langle FLANGE-BODY \rangle$ structure; 3) and 4) the third and the fourth productions define the connection between the structure generated at step 2) and respectively the structures of $\langle REAR COVER \rangle$ and $\langle SCREWS \rangle$; 5) the rest of productions such as the addition of *lip-seal*, *Woodruff-key*, *nut*, *washer* etc. are used to define and generate the final $\langle GEAR PUMP \rangle$ structure.

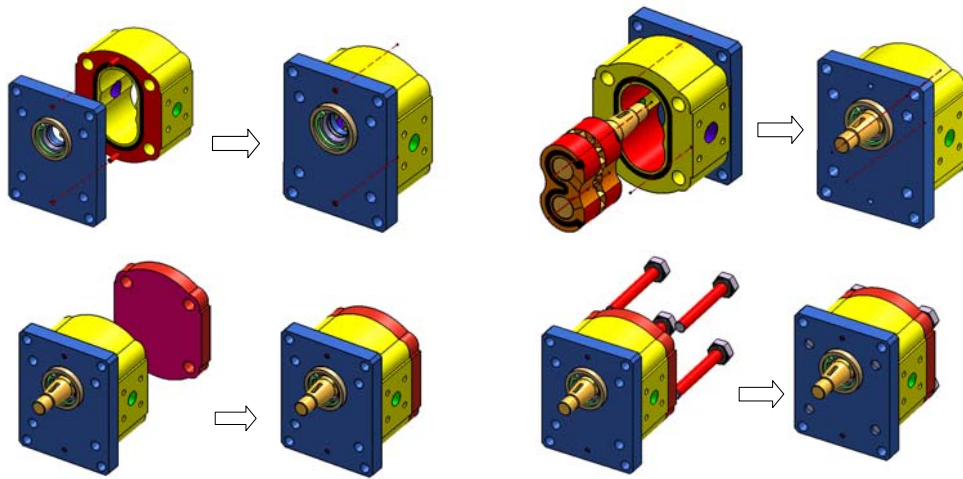


Figure 14 Productions to generate the gear pump structure

We have inferred our structural grammars for configuration based on features, **SGCF**, to generate a family of external gear pumps (figure 15). By applying the property of *replacement* or *swapping*, different instances of a single gear pump structure were generated (figure 15,a). By applying the grammar property of *recurrence* on the gear pump structure, multi-gear pump structures were generated (figure 15,b). The number of instances of the pump structure can be infinite. So, if from the strict grammar productions point of view, all these structures that were generated can be valid, afterwards these pump configurations have to be validated according to the technical constraints of the hydraulic domain.

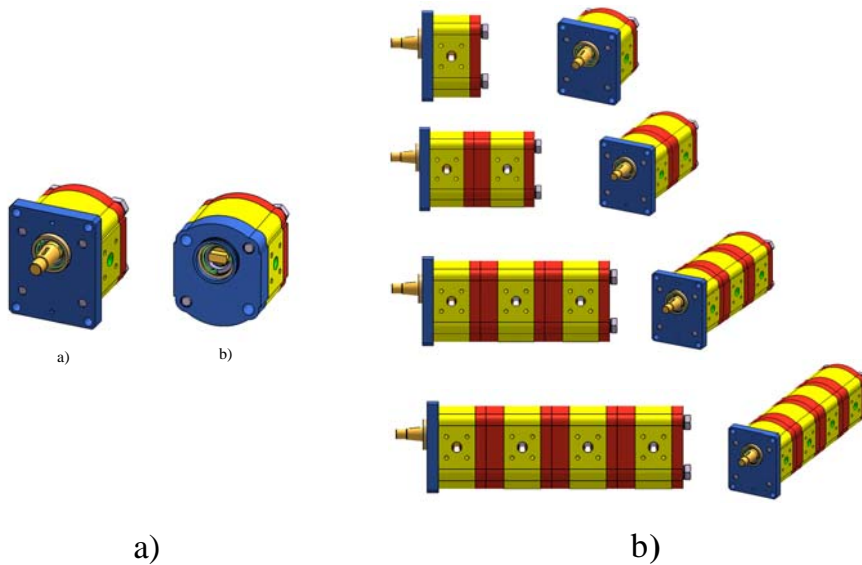


Figure 15 Different configurations of gear pumps generated with the SGCF grammar

4 CONCLUSIONS

Structural Grammars for Configuration based on Features, **SGCF**, are powerful tools for highlighting the structural relationships inside the product configuration. Developed on the configurable product properties, they permit to represent and to generate various and new products. Hence, *SGCF Grammars* can be used to capitalize and to create the *configurable product knowledge*. In our research, we have considered the industrial case study of the design of a gear pump family to validate our *configuration grammar-based design approach*. This case study investigates the possibilities and the conditions of inference of the *configuration grammar* through a set of production rules applied to pump structures in order to generate alternative and innovative gear pump configurations. The configuration grammar used in this approach is to be implemented in CAD environment software.

REFERENCES

- [1] Agard, B. *Contribution à une méthodologie de conception de produits à forte diversité. PhD Thesis*, Institut National Polytechnique de Grenoble, 2002.
- [2] Chase S. A model for user interaction in grammar-based design systems. *Automation and Construction*, Vol. 11(2), 2002, pp. 161-172.
- [3] Deciu E.R., Ostrosi E., Ferney M., Gheorghe M. Configuration of product families using fuzzy set approach. *Proceedings of the 14th International Conference on Engineering Design (ICED'03)*. In Research for practice: Innovation in Products, Processes and Organisations, edited by Folkesson A., Grallén K., Norell A., Sellgren U., Stockholm, ISBN 1-904670-00-8, 2003.
- [4] Deciu E.R., Ostrosi E., Ferney M., Gheorghe M. Configurable product design using multiple fuzzy models. *Journal of Engineering Design*, Vol. 16(2-3), 2005, pp. 209-235.
- [5] Du X., Jiao J., Jiao J. and Tseng M. Graph Grammar based product family modeling. *Concurrent Engineering: Research and Applications*, Vol. 10(2), pp. 113-128, 2002.
- [6] External gear-pumps catalogue: *Hesper, Rexroth Bosch, Enerflux Industrie*.
- [7] Männistö T., Soininen T., and Sulonen R. Modeling Configurable Products and Software Product Families. *Presented at the IJCAI'01 Workshop on Configuration*, Seattle, 2001.
- [8] Ostrosi E., Ferney M. Feature modeling grammar representation approach. *AIEDAM*, Vol. 19(4), 2005, pp. 245-259.
- [9] Ostrosi E., Ferney M., Deciu E.R., Garro O. Feature-based reasoning for designing structural configuration in advanced CAD systems. *5th International Conference on Integrated Design and Manufacturing in Mechanical Engineering*, 5-7 April, Bath, 2004.
- [10] Sabin D. and Weigel R. Product Configuration Frameworks – A survey. *IEEE Intelligent Systems*, Vol. 13(4), 1998, pp. 32-85.
- [11] Schmidt L.C. and Cagan J. Optimal configuration design: an integrated approach using grammars. *ASME Journal of Mechanical Design*, Vol. 120(1), 1998 pp. 2-9.
- [12] Schmidt L.C., Shi H., Kerkar S. A constraint satisfaction problem approach linking function and grammar-based design generation and assembly. *ASME Journal of Mechanical Design*, Vol. 127(2), 2005, pp. 196-205.
- [13] Siddique Z. and Rosen D. Product platform design: a graph grammar approach. *In Proceedings of DETC'99, ASME Design Engineering Technical Conferences*, 1999.
- [14] Soininen T., Tiihonen J., Männistö T. and Sulonen R. Towards a General Ontology of Configuration. *AIEDAM*, Vol. 12(4), 1998, pp.357–372.
- [15] Snavely G.L., and Papalambros P.Y. Abstraction as a configuration design methodology. *Advances in Design Automation*, (New York: ASME) DE - (65)-1, 1993, pp. 297-305.
- [16] Tiihonen J., et al. Modelling configurable product families. *Proceedings of ICED'99*, Vol. 2, Munich, 1999, pp.1139-1142.
- [17] Veron M., Fargier H. and Aldanondo., M. From CSP to configuration problems. *In Workshop AAAI'99 on Configuration*, Orlando, FL, 1999.
- [18] Schmidt L.C., Shi H., Kerkar S. A constraint satisfaction problem approach linking function and grammar-based design generation and assembly. *ASME Journal of Mechanical Design*, Vol. 127(2), 2005, pp. 196-205.
- [19] Winger, L., Introducing form features in product models, a step towards cadcam with engineering terminology, Licentiate Thesis, Dept. of Manufacturing Systems, Royal Institute of Technology, Stockholm, 1991.
- [20] Shah, J.J., Philosophical development of form feature concept, CAM-I report P-90-PM-02, 1990.

Contact: E.R. Deciu
Université de Technologie de Belfort-Montbéliard
Laboratoire M3M
90 010 Cedex Belfort
France
Phone : +33 (0)3 84 58 31 88
Fax : +33 (0)3 84 58 31 46
eugeniu-radu.deciu@utbm.fr