

# A Survey of the Persistent Naming Problem

David Marcheix    Guy Pierra

Laboratoire d'Informatique Scientifique et Industrielle (LISI)  
Ecole Nationale Supérieure de Mécanique et d'aérotechnique (ENSMA)  
Téléport 2 — 1 avenue Clément Ader — BP 40109 — 86961 Futuroscope Chasseneuil cedex — France  
(+33/0) 5 49 49 80 63

{marcheix | pierra}@ensma.fr

## ABSTRACT

In this paper, we present a survey of existing approaches on persistent naming in parametric system. We identify five common concepts that may be found in most of the studies. We propose two orthogonal criteria for classifying persistent naming approaches. This survey is intended to represent a state of the art of robust parametric reevaluation of geometric models.

## Categories and Subject Descriptors

I.3.5 [COMPUTER GRAPHICS]: Computational Geometry and Object Modeling – *Boundary representations, Hierarchy and geometric transformations.*

## General Terms

Design, Algorithms, Reliability, Performance, Human Factors, Theory.

## Keywords

Persistent naming, CAD, Parametric design, Reevaluation.

## 1. INTRODUCTION

The reuse of part models became an important issue in the CAD domain. Nowadays, feature-based parametric systems pioneered in the late 1980s by Parametric Technology (Pro/Engineer) and Cimplex Inc. (Cimplex) enable easy conception and modification of parts. Indeed, it has been estimated that perhaps 80 % of all design tasks consist in adapting existing models [22] through simple modifications of dimension values, constraint relations and feature definitions. Moreover, feature-based parametric systems makes it possible to generate mechanical families of parts and thus to represent and exchange libraries of standard parts. Finally, they enable to perform analyses of mechanisms or tolerance analyses by applying extreme values of test to the model.

Feature based parametric systems contain the geometric and/or topological representation of the object, a set of parameters

PERMISSION TO MAKE DIGITAL OR HARD COPIES OF ALL OR PART OF THIS WORK FOR PERSONAL OR CLASSROOM USE IS GRANTED WITHOUT FEE PROVIDED THAT COPIES ARE NOT MADE OR DISTRIBUTED FOR PROFIT OR COMMERCIAL ADVANTAGE AND THAT COPIES BEAR THIS NOTICE AND THE FULL CITATION ON THE FIRST PAGE. TO COPY OTHERWISE, OR REPUBLISH, TO POST ON SERVERS OR TO REDISTRIBUTE TO LISTS, REQUIRES PRIOR SPECIFIC PERMISSION AND/OR A FEE.

SM'02, JUNE 17-21, 2002, SAARBRUCKEN, GERMANY.  
COPYRIGHT 2002 ACM 1-58113-506-8/02/0006...\$5.00.

(characteristics of the object) and a set of constraints (equations or functions) applied to the object. More generally, a parametric modeler is a system of geometrical design which preserves not only the explicit geometry of the designed object (so called *parametric object* or *current instance*), but also a mechanism able to reevaluate it when some parameters are changed (so called *design process*, *constructive gestures* or *parametric specification*).

Most of current feature-based parametric systems are known as *procedural* or *history-based* because the parametric specification may be regarded as a composition of modeling functions where each function is attached via its parameters to topological entities defined in previous states of the model. Referenced entities must then be named in a persistent way in order to be able to reevaluate the model in a consistent manner. In particular, when a reevaluation leads to topological modifications, references between entities used during the design process are frequently reevaluated in an erroneous way, giving results different from those expected.

This problem is known as "*topological naming*" when names use only topological information and "*persistent naming*" when names also involve other kind of information such that geometry of feature orientation [13], [6].

In this paper, we present a survey of this domain. In the next section, we discuss the persistent naming problem in parametric modeling systems. In section 3 we identify the common concepts that results from most of the studies and that may be considered as the state of the art for persistent naming in parametric geometry. In section 4 we propose two orthogonal criteria for classifying

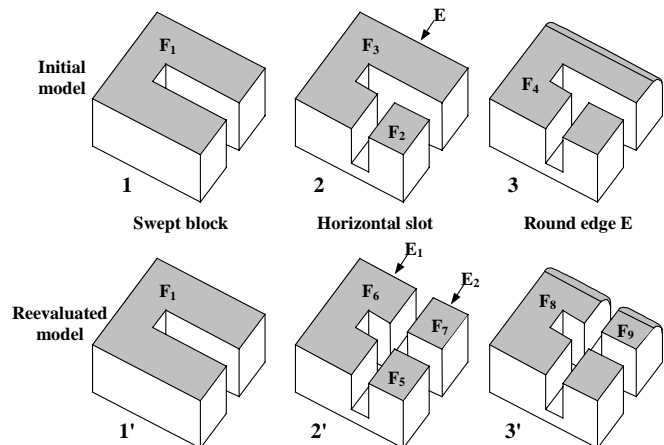


Figure 1: Naming and matching problems.

persistent naming approaches. The first criterion refers to the use of geometry in the naming phase. If all the approaches use some geometrical information to remove ambiguity for non linear cases, for polyhedral geometry naming may be based either on purely topological information, or it may be based on both topology and geometry. The second criterion refers to the reevaluation phase. When a name assigned to an entity created at design time after some constructive gesture is to be mapped on entities of the reevaluated model, the mapping algorithm may involve globally all the entities resulting from this particular constructive gesture in the initial and reevaluated models, or it may only involve locally the named entity of the initial model and search for a best fit in the reevaluated model.

## 2. THE PERSISTENT NAMING PROBLEM

History based modeling systems contain different shortcomings due to a strong dependency on the chronological order of feature creation, constraints solving limitations, the use of manifold boundary representation, etc. [3]. But the main problem is surely to characterize geometric and topological entities of a parametric model. Characterizing entities consists in giving them a name at design time and "finding them" again at reevaluation time (i.e. *matching* entities of the initial model and entities of the reevaluated model.) Let us take the example of Figure 1 to illustrate this problem.

In the above example the initial model is designed by means of a parametric specification containing four successive constructive gestures. The fourth one consists of rounding edge "e". If the initial model is saved after this fourth step, the current instance no longer contains edge "e": it was removed by the rounding function. Thus the rounding function which has edge "e" as input parameter cannot any longer be represented in the parametric specification part of the model by simple reference to current instance entities. Therefore "names" are needed to represent the parametric object entities referenced in the parametric specification whether or not they exist in the current instance.

Moreover each constructive gesture creates a number of entities

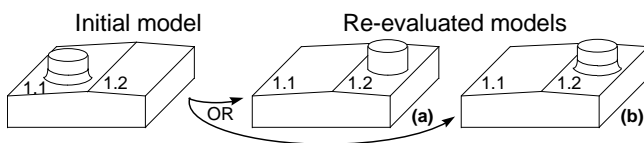


Figure 2: Capturing various semantics.

which have to be distinguished and therefore named, to be referenced by further constructive gestures, even if the same number of entities exist in all possible reevaluation (no topological change). The problem is even more complex for parametric models, of which entity types and /or entity numbers change from one evaluation to another. Let us return to the above example, but now at reevaluation time. We notice that, at step 3', the edge "e" has been split into edges "e<sub>1</sub>" and "e<sub>2</sub>". Thus at step 4' the problem is to determine which edge or edges have to be rounded. The problem is to identify, i.e. to match, edge "e" with edges "e<sub>1</sub>" and "e<sub>2</sub>" despite topology changes. Thus, when reevaluation leads to topology changes a new issue is to match two different structures.

It is thus necessary to have, in addition to the naming mechanism, a robust matching mechanism regarding reevaluation.

The third identified problem is to be able to express different semantics that capture the "design intent". In the example of Figure 1, we have just seen that a fundamental problem is to identify, i.e. to match, edge "e" with edges "e<sub>1</sub>" and "e<sub>2</sub>" despite topology changes. However, it may be possible that the user wanted to round only the edge "e<sub>1</sub>" during the reevaluation. In fact, various results for the same matching might be desirable depending on the user initial goal. The problem here is thus to have a model able to represent, i.e. to name in a persistent way, entities corresponding to various semantic and able to characterize various user design intents. This could be done in particular by naming high level abstractions (e. g., aggregates of geometric/topological entities). For example, the designer might want to express that a feature is applied on the global shell rather than on a particular face. Figure 2 illustrates this problem. In the initial model, the object is designed in three steps : creation of the block by extruding a polygonal contour, creation of the cylinder on the block, then rounding the edge between the block and the cylinder. According to whether the rounding function was expressed between the cylinder and face 1.1 (case 1) or between the cylinder and the upper shell composed of faces 1.1 and 1.2 (case 2), the reevaluated model is different because the "design intent" is different. In the first case one obtains model (a) where the round disappeared since it cannot be made any more between the cylinder and face 1.1. In the second case one obtains model (b) where the round always exists since it could be made between the cylinder and the upper shell. To support these two different semantics, the naming mechanism should provide for persistent naming and matching high level entities such as shells.

## 3. STATE OF THE ART OF PERSISTENT NAMING

Although parametric modeling has developed and expanded for more than one decade, both in research centers and in the industry, probably due to commercial competition only a rather small number of publications were issued in the field of persistent naming. One important precursory work in this domain is that of Hoffmann and Juan [10]. Over the same period of time, several authors have studied the internal structure of a parametric data model, proposing various representations [10] [18] [9] [23] [19] [15] [1], analyzing the underlying mathematical structures [18] [20], and discussing various problems related either to the semantic of the modeling operations [9] [7] [2], or to the management of modeling constraints [4].

In view of this research it becomes possible to ascertain that, even if they are not explicitly stated, there exist a certain number of underlying concepts which are present in most of the suggested models. We thus propose first to identify these common concepts, before exploring possible criteria to classify the various approaches.

The first fundamental concept that one finds in most existing approaches on persistent naming is the distinction between invariant and contingent entities, precisely defined in [1]. An *invariant entity* is a geometric or topological entity which can be, completely and unambiguously, characterized by the structure of a constructive gesture and its input parameters, independently of involved values. In Figure 1, invariant entities includes the end

face of the swept block, the lateral shell of the horizontal slot with its begin and end faces (that may, or may not exist), the shell resulting from the rounding gesture, etc. To characterize, i.e. to "name", such entities, information models are to be defined that relate these entities to constructive gestures and to their input parameters. A *contingent entity* is a geometric or topological entity that results from the interaction between the pre-existing geometric model and invariant entities resulting from the last constructive gesture. For example, in Figure 1, the number of lateral faces of the horizontal slot in the initial model (step 3) and in the reevaluated model (step 3') is not identical. A naming mechanism is also required to define how to name these contingent entities. This distinction exists, more or less explicitly, in most approaches where different naming mechanisms are defined, or implied, for invariant or contingent entities.

The second common concept relates to the naming mechanisms used for invariant entities resulting from constructive gestures which involve two dimensional topological structures as input parameters. For example a profile and a trajectory in the case of an extruded profile. It is always assumed that each two dimensional entity belonging to each two dimensional topological structures is already named in a robust way. Then, each invariant entity is named by reference to the name or names of the two dimensional entity or entities from which it results. In the extruded profile example, each topological entity (vertex, edge, and/or face) of the result is named according to the names of the entities defining the profile and the trajectory, as well as according to rank order the constructive gesture in the parametric history. A side face of a slot builds by extrusion, corresponds to the Cartesian product of an edge of the section by an edge of the trajectory. The face can then be characterized by the names of these two edges. Persistence of names of invariant entities is then related to persistence of names of two dimensional input parameters entities. Here, most approaches guarantee this persistence by prohibiting topological modifications of profiles and trajectories. It is obvious that a robust naming mechanism in a three dimensional space if it exist, might also be applied to lower dimensions, thus authorizing topological modifications. However, most current CAD systems do not allow these kinds of operations or indeed lead, during the reevaluations, to incoherent models. Thus, this question remains open.

The third common concept relates to the building blocks that are allowed for use to name contingent entities. Most approaches only allow the use of invariant entity names. For example, the topological neighborhood of a contingent face can be used to characterize it. This neighborhood can even contain contingent faces. In this case, the names used are those of the invariant ancestors' faces of which the contingent faces are historically resulting. To use names of invariant ancestors makes it possible on the one hand to use more robust references to characterize an entity, and on the other hand to eliminate the problems of cross references which can appear during a matching.

The fourth common concept is precisely to represent by some means history of names. The use of invariant ancestor names requires capability to trace the history of each entity. This may be done either using a dedicated structure [13], [5], [2], or by means of a mechanism that propagates attributes from an entity to any entities that result from it, thus names of invariant entities may be propagated as a particular attributes and they are directly available from their descendant entities [7], [25]. Moreover, most

approaches use only face history. Other entities are named by reference to faces. There seems to be two main reasons for this. Firstly, topological edges and vertices in the current instance may result from invariant entities belonging to various types (a vertex may result from the intersection of an invariant face and an edge, or from the intersection of two edges, etc). Conversely, each contingent face results from an initial invariant face of some features. Secondly, during Boolean operation evaluations, topological edges and vertices are changed dynamically, whereas the faces are relatively stable.

Finally, the fifth common concept is the global software architecture that results from most contribution and that we call the three layers architecture. A parametric model consists of three following layers:

- The *parametric specification layer* contains the description of the modeling operations (feature-based operation, Boolean operations, chamfering operations, blending operations, 2D contour and sweeping operations, etc.) that defined the parametric object; these operations refer to a layer of names
- The *name layer* allows to represent geometric or topological entities that existed at some time during the parametric design process; these name may reference geometric or topological entities if they still exist in the geometrical layer
- the *geometrical layer* contains the geometric and topological entities that constitute the current instance; it may also contain geometric and topological entities that existed in some previous steps of a parametric design to support UNDO functions.

## 4. A TAXONOMY OF PERSISTENT NAMING APPROACHES

### 4.1 Entity naming : use of geometry/topology

Entities referenced by constructive gesture are mainly topological entities. As stated before, invariant entities can be completely and unambiguously characterized by the structure of a constructive gesture and its input parameters. Conversely, contingent topological entities results from the interaction between the pre-existing geometric model and invariant entities resulting from the last constructive gesture. The main problem for naming methods relates to the characterization of topological entities resulting from same invariant topological entities. The first criterion for classifying persistent naming approaches refers to the method used to identify these contingent entities. If all the approaches use some geometrical information to remove ambiguity for non linear cases (for example characterization of the two contingents edges resulting from the intersection of two cylinders), for polyhedral geometry naming may be based either on purely topological information (more or less wide topological neighborhoods, local topological orientations, etc), or it may be based on both topology and geometry (parameterization of local spaces, etc.).

#### 4.1.1 Identification based on topological entities

A particularly significant work of identification, based on topology, is certainly that of Chen. This work was carried out at the University of Purdue under the direction of Hoffmann. The

proposed model is composed of two representations [7]. For the first one, he uses an editable representation, called Erep [9], which is an unevaluated, high-level, generative, textual representation, independent of any underlying core modeler. It abstracts the design operations, contains the parametric specification and stores all entities by name. The second representation, evaluated and modeler dependent, contains the geometry (the current instance). The link between these two representations is obtained by a name schema which establishes the link between the geometric entities of the geometric model and the generic names (persistent) of the unevaluated model. Chen defines a precise structure for naming invariant entities, vertices, edges and faces (before the interaction with the actual geometry) which belong to a specific class of form features build by sweep operations (extrusions or revolutions). In this precise context, every entity in a sweep is named by reference to the corresponding source entity of the swept 2D contour and the constructive gesture. He also proposes an identification technique for contingent entities based on compositions of topological contexts (more or less extended topological neighborhoods) and on feature orientations. Each contingent entity is described by its origin, either an invariant entity or an intersection of invariant faces, its smallest unambiguous topological context and its local orientation in the Brep model [7] [6]. So, in the linear case, ambiguity may be considered as removed by topological information only.

To also ensure uniqueness of names in curved domains, one additional information based on geometry is added to the previous topological information : the orientation of any edge against the extrusion direction of the feature(s) it belongs to.

The matching of an entity is realized through a local comparison of topological neighborhoods. For example, in case of faces, the face which must be matched is compared with the whole set of faces issued from the same invariant face (preliminary set). At each stage of the construction, the contingent faces inherit the same name of their parent face which makes it possible to construct the preliminary set. A grade is associated to each face of this preliminary set. The grade for each candidate face is the number of matched boundary edges. The face is kept if this number exceed a threshold.

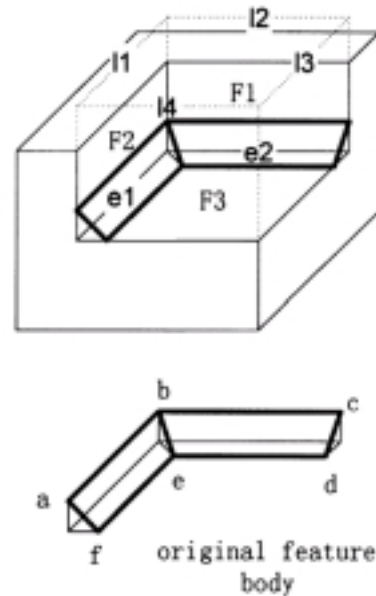
In his study, Chen restricted to three kinds of features: sweep (extrusion and revolution), blend and fillet. For these features, he shown the feasibility to identify unambiguously any topological entities of models defined by successive attachment of such features, even when faces are curved, in most practical case, i.e., when there are not too many symmetries in the model. A matching algorithm is also proposed that support some level of topological changes in the re-evaluated model. However, it is not clear how the reduced context is used in this algorithm. Moreover this algorithm uses some thresholds, and no precision is given on the choice of these thresholds and the rational for the choices. Finally, the matching algorithm is local to the entity to be retrieved (see 4.2). In case of Figure 4 for instance, and depending on the threshold used,  $F_2$  would probably be mapped onto  $F_x$ .

The suggested model represents two major contributions in this domain : on the one hand, two main concepts for topological identification of entities, i.e. topological context and feature orientations which will be used thereafter per many of other

approaches, and on the other hand a very precise study of cases of ambiguity.

#### 4.1.2 Identification based on geometry

A second approach consists in using geometry instead of topology to discriminate contingents entities. A recent work, which investigates in this direction, is that proposed by Wu and al. [25] from Huazhong University of Science and Technology. Parametric models are considered as a sequence of regularized operation of construction between a part body and the body of an original feature. As the body of an original feature is created, every face of this feature is attached with a name (called ON for original name). Then a Boolean operation is conducted between the part and the body with names. If it is needed to reference an entity (face, edge or vertex) in the part body when creating the later features, they should be recorded with other names (called RN for real names), which are derived from original names. When regenerating the whole part, all the real names in the part model should be retrieved to prepare data for the reevaluation. ONs are stored in the geometry and are propagated to the descendant faces, in the case of scission, of fusion or modification, which makes it possible to link each contingent face to its invariant initial face. An ON is compound of an identifier of constructive operation (Sweep feature, Revolve feature, Chamfer feature), and by an identifiers which make it possible to know from which invariant entities results the face. For example in the case of the sweep illustrated in Figure 3, the part is made by subtracting one block from the other one and chamfering edge  $e_1$  and  $e_2$ . Although the original feature body contains eight faces, actually only faces (a,b,e,f) and (b,c,d,e) are the feature faces.  $F_1$ ,  $F_2$ ,  $F_3$  are the



**Figure 3:Name chamfer feature**

feature faces in the cut Extruded feature, whose identity value is assumed to be 6. This Extruded feature has a profile, which consists of four elements,  $I_1$ ,  $I_2$ ,  $I_3$  and  $I_4$ . The identity of the profile is 5. The ON of a swept entity is given by :

$$ON(F)=\{\text{FeatID}, \text{FeatID}_p, \text{ID}_{\text{element}}, \text{FeatID}_{\text{path}}, \text{ID}_{\text{trajectory}}\},$$

where  $featID$  is the feature identity of the Sweep,  $FeatID_p$  and  $FeatID_{path}$  are the identities of the profile and the path,  $ID_{element}$  and  $ID_{trajectory}$  are the identities of one edge of the profile and one edge of the trajectory. In case of Extrude feature, which is formed by sweeping a profile along the normal of the profile, the  $FeatID_{path}$  and  $ID_{trajectory}$  can be set as zero.

Then, the context of the ON for  $F_1$  is :

$$ON(F_1) = \{6, 5, ID_{12}, 0, 0\},$$

When an entity is referenced, while the ON are propagated with the descendant faces, then several faces can have the same ON. Wu and al. propose to use an RN compound of the ON to which one adds additional information allowing to eliminate ambiguity in construction. This purely geometric information, is based on the parameter space  $(u,v)$  of the surfaces on which the faces belongs. An arrangement of the subdivided faces in  $u$  or  $v$  directions (for each face, distance between a characteristic representative point and the origin of the reference coordinate system), makes it possible to associate a different number to each face. Other topological entities (vertices, edges) are named with the RN of the adjacent faces and, in a similar way, with an information on the parameter space. The calculation of geometric information on the parameter space is only carried out for the referred entities, which limits largely spatial and temporal complexities.

Wu and his colleagues build an unambiguous name during the construction stage, mainly using geometric information, but there is no explanation about how the matching is to be performed. Moreover, this method need a large stability of the geometric parameterization. The arrangement of the subdivided faces seems to be very sensitive to geometric and topological variations. That could lead to not easily predictable matchings.

Nevertheless, this innovative method is an alternative to purely topological identification which needs to be further experimented.

## 4.2 Entity retrieval : global/local matching method

Most methods that concentrate on the retrieval phase do not name all the entities but only those contingent entities that are referenced by a constructive gesture and all the invariant entities, mainly the faces, that might be needed to name the referenced entities. The second classification criterion relates to the globality versus locality of the matching method. In the global solution, matching is carried out by the comparison of two sets of entities (one resulting from the initial model and the other one resulting from the reevaluated model) and by the mapping of all these entities. In the local approach, this combinative calculation is limited, since only the entity referred to in the initial model is compared with the set of entities resulting from the reevaluated model. A global solution would be more expensive in computing times, but theoretically makes it possible to obtain a more robust and reliable naming mechanism. For example, the solid represented in Figure 4 results from the construction of a slot on a swept block. During reevaluation, the parameter concerning the orientation of the slot is modified. Let us suppose that the face  $F_2$  is referenced by a following constructive gesture. A local matching method would look for the face whose topological neighborhood is more similar to the neighborhood of  $F_2$  among  $F_x$  and  $F_y$ . The number of common neighbor faces between  $F_2$  and  $F_x$  is higher than between  $F_2$  and  $F_y$ . The face  $F_2$  will thus be

mapped on the face  $F_x$ . Conversely, a global method will seek to match the whole set of faces resulting from the initial model ( $F_1$  and  $F_2$ ) with those resulting from the reevaluated model ( $F_x$  and  $F_y$ ). In this example a global matching method will map the face  $F_1$  on the face  $F_x$ , what will induce the mapping of the face  $F_2$  on the face  $F_y$ .

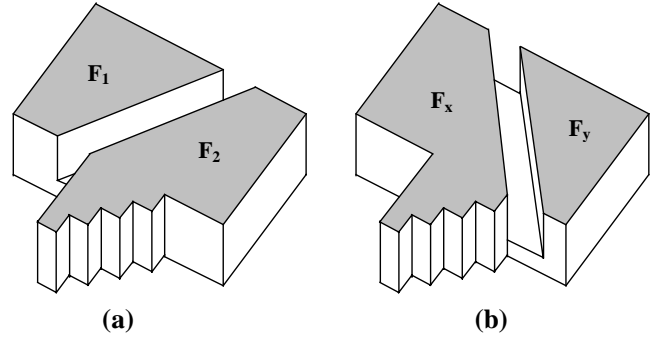


Figure 4: (a) A slot on a swept block. (b) Reevaluation of the model after a modification of the slot direction

### 4.2.1 Local matching

The Matra Datavision company [16] proposes as open form source, its OCAF development platform (*Open CasCade Application Framework*) which contains a geometric kernel, a data structure and a local naming mechanism which can be used in parametric geometric system [5], [17].

To our knowledge, this naming method and this name structure has never been published. We briefly outline the method with the agreement of the authors [5]. Their naming mechanism contains two parts. Firstly, a data structure to represent the evolution of some entities in the model (primitive, generated, modified, removed, etc.), and to enable tracing the face history. Secondly, a name structure which uses this data structure and which is based on the topological neighborhoods to characterize entities. This approach is minimalist insofar as the names are built only for the entities referenced by the parametric specification. In fact, the names are expressions built from **operators**, **invariant faces** and **type of entity** (vertex, edge or face). The invariant faces, contained in the name, are found by a "backward traversal" of the structure preserving the history of the faces. The five operators used are the following :

- **CURRENT (F)** : Return the active faces resulting from the invariant face F.
- **INTERSECTION ( $E_1, E_2, \dots$ )** : Return the intersection of the entities  $E_1, E_2, \dots$
- **UNION ( $E_1, E_2, \dots$ )** : Return the set union of the entities  $E_1, E_2, \dots$
- **GENERATEDBY ( $E_1, E_2$ )** : Return the entities in  $E_2$  generated by the entities in  $E_1$ . This operator can be used during a sweep operation, to find the faces in  $E_2$  generated by edges in  $E_1$ , or during a Boolean operation to find in the resulting object  $E_2$ , the faces coming from a solid  $E_1$ .

- FILTERBYNEIGHBOURGS ( $S_1, E_1, E_2, E_3, \dots$ ) :  
Return entities of  $S_1$  which are connected with all the entities  $E_1, E_2, \dots$

Here are some examples of names :

- Descendant face  $F_1$  of an invariant face  $F_{10}$  :  
 $F_1 = (\text{FACE}, \text{CURRENT}(F_{10}))$ ;
- An intersecting edge of the descendant faces of two invariant faces  $F_{10}$  and  $F_{25}$  :  
 $E_1 = (\text{EDGE}, \text{INTER}(\text{CURRENT}(F_{10}), \text{CURRENT}(F_{25})))$ ;
- A vertex:  
 $V_1 = (\text{VERTEX}, \text{INTER}(\text{CURRENT}(F_{10}), \text{CURRENT}(F_{25})))$ ;
- A wire:  
 $W_1 = (\text{UNION}((\text{EDGE}, \text{INTER}(\text{CURRENT}(F_{11}), \text{CURRENT}(F_{2}))), (\text{EDGE}, \text{INTER}(\text{CURRENT}(F_{10}), \text{CURRENT}(F_{25}))))$

The operators allow to build sets of entities, then to filter them (with topological neighborhoods) and to specialize them until obtaining a minimal, single and unambiguous name on the current instance during construction stage. When an entity is interactively designated, the modeler returns pointers on the incidental faces. Then OCAF traverses the faces history looking for leave faces which have identical pointers on entities of the topological model. When the faces are identified, OCAF makes a backward traversal in the historical structure until their invariant ancestor faces. Then these invariant faces are used to create an initial version of the name. Finally, OCAF carries out the evaluation of the name just generated. Either the number of returned entities correspond with the initial request (one face has been referenced and one face is returned, or a wire of six edges has been referenced and a wire of six edges is returned), the name is then validated, or the number of returned entities does not correspond with the initial request, then it is necessary to refine the name by adding a filter by neighbors. In this case, OCAF builds initial names for all the same dimensional entities in the topological neighborhood of the referenced entity, and adds them to the name of the interactively designated entity as operands of FILTERBYNEIGHBOURGS operator. This is equivalent to name adjacent contingent entities of the referenced entity, and to use these entities to eliminate the ambiguity. This refined name, generated during the construction process, is stored in the model, then similarly, it is evaluated on the new historical structure of the faces built during the reevaluation. In Figure 5.(a), the user selects the edge  $E_1$  by an interactive selection. Then an initial naming is build for this edge.  $E_1 = \text{INTER}(F_1, F_{23})$  where  $F_1$  and  $F_{23}$  are respectively the invariant ancestors of the face  $F_7$  and  $F_6$ . In Figure 5.(b), the system evaluate this expression and find two resulting edges ( $E_1$  and  $E_2$ ). The number of returned entities does not correspond with the initial request, then the naming process specializes the name by adding of a filter by neighbors. Thus, all the adjacent edges of  $E_1$  are named (Figure 5.(c)) :

$$E_5 = \text{INTER}(F_1, F_3)$$

$$E_3 = \text{INTER}(F_1, F_{32})$$

$$E_4 = \text{INTER}(F_{32}, F_{23})$$

$$E_6 = \text{INTER}(F_3, F_{23})$$

Where  $F_3$  and  $F_{32}$  are respectively the invariant ancestors of the face  $F_9$  and  $F_8$ .

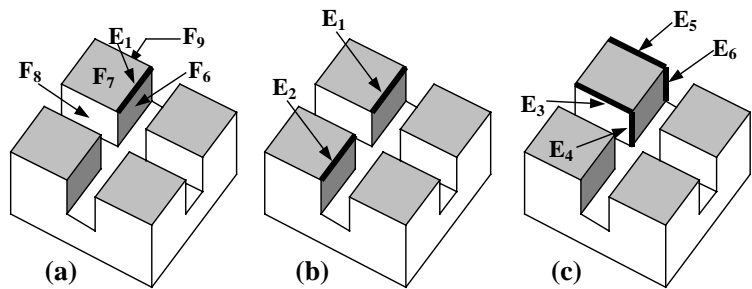
Then these names are added in the name of  $E_1$  :

$$E_1 = (\text{INTER}(F_1, F_{23}); \text{FILTERBYNEIGHBOURGS}(E_2, E_3, E_4, E_5))$$

The OCAF method is rather similar to the Chen approach as it is both topology oriented for naming in a polyhedral context and based on local matching method. However there exist two main differences. As OCAF is designed to support any kind of modeling gesture, including Boolean operations, it does not use the Chen's concept of feature orientation which is basically the sweep direction of the 2D profile. Thus its scope is broader, but on the other hand few details are provided about how to remove ambiguity in non-linear cases. The second difference is that naming and matching of any entities is only based on faces and face history. When Chen names (and matches) edges to identify faces, and vertices to identify edges, OCAF only uses invariant faces for building and matching names of any kind of topological entity without using edge orientations.

This naming system of OCAF offers a fair level of robustness as based on invariant entities. However, the model is local, since to evaluate the entity's name on a new faces history structure is similar to compare an entity of the initial model with several entities of the reevaluated model. Avoiding to manage in parallel two significant data structures simplifies considerably the matching process, but on the one hand requires, during the construction process, to be able to establish some judicious heuristic for refining the names and on the other hand can leads to not easily predictable matchings as shown in Figure 4.

In addition to this naming mechanism, the CasCascade system



**Figure 5: Generative process of a name in OCAF**

enables to support a relatively sophisticated Undo/Redo operation. Authorizing this operation implies to be able to regenerate the state of the model (topological and geometric layer, names layer, parametric specification layer) at any construction stage  $k$ . Therefore, two approaches are widely used. The first one consists in rebuilding the model by a reapplication of all the first operations (stage 1 to  $k$ ). Most of the time, this approach leads to high computational complexity. The second one consists in storing the various layers of the model for some construction stage (last stages or some significant stages). The modeler keeps thus all the references towards topological entities valid, but that implies a storage cost becoming rapidly excessive in CAD context. To limit this cost, the b-rep topological model used by CasCascade represents only downward pointers (compound compsolid solid shell

face wire edge vertex). As illustrated in the Figure 6, a same topological entity in the b-rep can be pointed by several higher level topological entities, corresponding to various stages of modeling process. That enables to avoid the duplication of topological entities which remain identical between two modeling stages (F1.2, E1.2, etc.). Only entities generated (intersecting edges, etc) or modified (F2.0, F2.1, etc) during a modeling stage are represented explicitly. For any modeling stage, creation of a complete b-rep object (with upward pointers) must be realized in an unique downward traversal of the previous structure. The level of complexity for an update of the model is sub-linear, since all the unmodified parts are reused without duplication.

A more detailed analysis, as well as a significant reverse engineering work based on the sources available, can be found in [8].

### 4.2.2 Global matching

#### 4.2.2.1 Kripac

At the Czech Technical University of Prague, Kripac focuses on the name matching [12], [13], [14]. He proposes an API (Application Programming Interface) encapsulating its topological identification system and guaranteeing the persistence of the names using a table of correspondence between an entity of the initial model and one or more entities of the reevaluated model. In this particularly innovative work, Kripac proposes an interesting structure for identification of any topological entities based on face history (creations, splits, merges and deletions of faces) and a name matching algorithm to search for the entities after modification. Kripac's Topological ID System consists of 3 parts. First a face graph allowing both to name all the other topological entities according to their neighborhood in term of adjacent faces, and to carry out a name matching during reevaluation. Second a table recording names for the only entities that are referenced in the parametric specification (edges and vertices are named in terms of their adjacent faces). This table also contains pointers to

the geometry (current instance) and some information for the matching algorithm. Third the geometric models for each step of the modeling process. During each reevaluation all the faces, as well as every referenced entity in the parametric specification, are matched with the new ones.

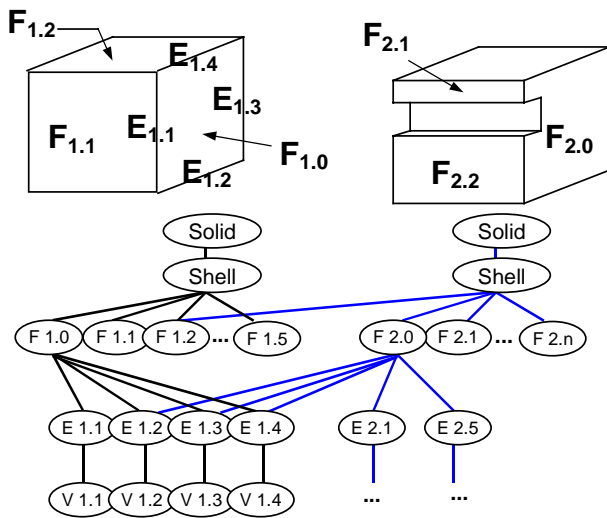
This matching is based on a comparison of the topological neighborhood (stored in the face graph) between one or more faces of the initial model and one or more faces of the reevaluated model. The faces to be matched together are calculated with a "backward-forward" traversal algorithm. This algorithm traverses successively the initial and reevaluated graphs, finds all ancestors faces already mapped, and then respectively, in each graph, finds the set of leave faces resulting from this face. The explicit representation of the faces' history in the face graph enables to reference and to manipulate, at each step, a certain level of aggregation. The aggregation of all the faces which have a common historical ancestor. Thus, for example, the abstraction of face  $F_3$  in Figure 1 will exist, and could be referenced, even in case of subsequent split of this face, because the history is preserved.

So, in the linear case, face identification and matching are based on purely topological information: the invariant face or faces from which a face results, and the set of invariant faces that constitute its topological neighborhood. To remove ambiguity also in the curved domain, a geometrical information is added to each face that belongs to the neighborhood of a face to be identified. This information, called the geometric type of the intersecting edge, allows to characterize each one of the different possible intersections between two surfaces based on the type of the surface (height types are considered) and on the relative position of the two surfaces. Only intersection with NURBS cannot be so characterize because their shape may be arbitrary and they may intersect at any number of curves. This geometric information plays a similar role as the feature orientation used by Chen.

In addition to the used face graph structure, Kripac's approach is innovative because the proposed matching mechanism is global. The robustness and reliability induced by the global character of the matching method imply an overcost both in spatial (maintaining of two parallel structures) and temporal complexity (more entities to compare). Finally, Kripac's model does not allow to record that a selected mapping was only approximated as it uses a discrete metrics for evaluating a matching, and not a continuous one. For example, during the "backward" traversal phase, the "backward-forward" algorithm tries to locate an already mapped face. In general, this face was not exactly mapped (i.e. the topological neighborhoods of the faces of the initial and reevaluated graph were not identical). That strongly induces the later mappings and would deserve to be taken into account. Moreover, no explanation is given on the manner of representing and of using this relation in the graphs for the following operations.

#### 4.2.2.2 $PS^2$ Model (Persistent Solving Structure)

More recently, Agbodan, Marcheix and Pierra proposed in [1], [2] a complete framework, named  $PS^2$ , for identifying and matching any kind of entities based on their underlying topology. Up to now, the work on the  $PS^2$  method mainly concentrated on improving existing topological naming techniques to support dramatic topological changes in polyhedral domain. We are



**Figure 6: Extract of the Undo-Redo structure of OCAF-CasCade. A slot put on a lateral face of a block. After the second step of construction, the unmodified topological entities are not duplicated.**

currently evaluating the use of Kripac-like edge characterization to disambiguate edges that are topologically the same in the curved domain.

The identifying method is based on the invariant structure of each class of form features and on its topological evolution. This approach is similar to Kripac's one but provides a more general formalized framework. There are three main differences: First, they use a shell graph as basis of the persistent naming. Second, they use a hierarchical aggregation structure to capture, at various levels, the "design intent". Third, they introduce an iterative matching algorithm, which enables to weight geometrically the influence of the topology. The model contains:

- A definition, for each kind of feature, of the various semantics that might be expressed by a designer. This is done through a new taxonomy [2] where each kind of feature is associated with an invariant hierarchical aggregation structure.
- A naming mechanism that consists of two parts. First an oriented two dimensional shell graph. On one hand the *hierarchical* dimension which enables to represent different levels of granularity, and then the expression of different semantics. Each shell is composed of *connected* sub-shells where leaves are faces. On the other hand, the *historical* dimension which provides for tracing evolution (modification, split and deletion) of each aggregate in the hierarchical aggregation structures. Thus, a new graph node represents a new connected part which appears during a modification or split and enables to identify it and to record it (see Figure 7). A sub-shell can be part of several shells. The overlapping hierarchical structure

enables, during construction, the creation of any (connected) shell aggregate. This presents both a flexible and an extremely powerful designation mechanism. Especially for referencing aggregates stemming from the union of a feature with some parts of the object. Following [1] they distinguish invariant and contingent shells and they propose to identify, unambiguously and uniquely, both the invariant faces and shells, then the contingent faces and shells. Second, a table where edges, paths and vertices referenced in the parametric specification are named in terms of their adjacent faces or shells.

- A matching mechanism, where nodes of the shell graph are matched from the initial design shell graph onto the current reevaluation shell graph. Other topological entities are matched by reference to the shell graph. The matching method compares the initial and the reevaluated topological histories stored in these shell graphs. For each construction step, the matching consists of three phases. The first one consists in finding, in the initial and reevaluated graphs, the **smallest** bipartite set of nodes which can potentially be matched together, in order to limit the matching calculus. This is realized in the two shell graphs, through a recursive backward-forward traversal algorithm. The second one is the local phase, which consists in computing two measures of topological similarity between any couple of entities occurring in this bipartite set. These two measures allow to represent a mutual probability of inclusion, based on their topological neighborhoods weighted by the geometry. The third one is the global phase, which

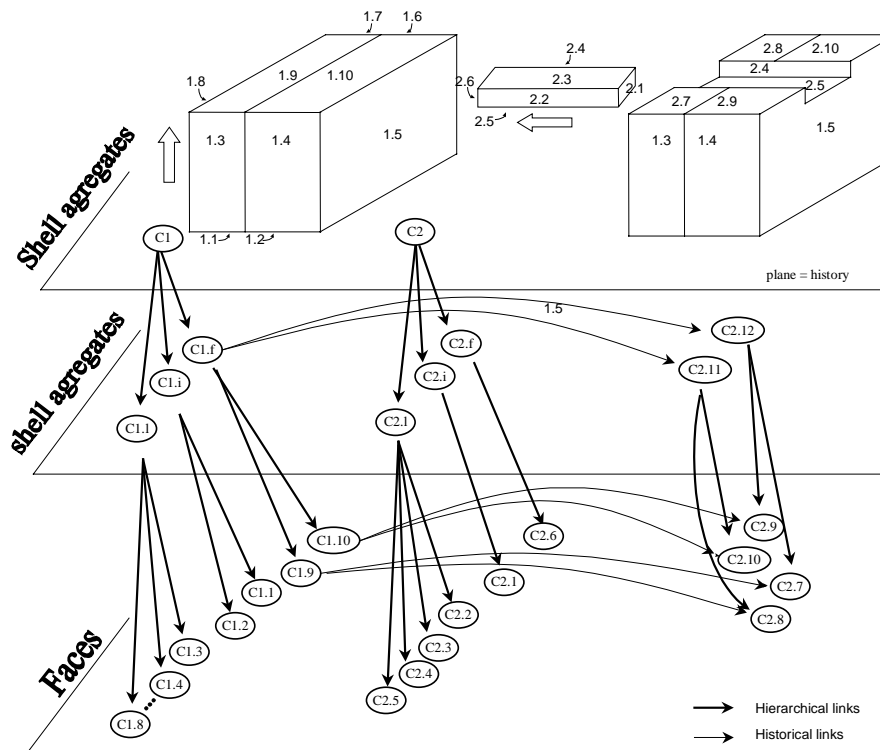


Figure 7 : The PS<sup>2</sup> naming structure

consists in defining the final matching as a binary relation that maximizes the topological similarity between the matched entities of both models. This maximizing operation uses measures of topological similarity calculated during the previous local phase. Then the maximizing measures are represented as bi-valued links between each nodes of the two shell graphs. In a later matching stage, these bi-valued links are used to calculate recursively the smallest bipartite sets of the first phases.

This PS<sup>2</sup> model proposes a persistent naming mechanism during the construction process, and a whole matching algorithm which deals with the semantic representation problem stated in section 2. It proposes an explicit representation of shells which make it possible to reference meaningful high level abstraction. For example, in an extrusion there is a single lateral shell which is composed of several lateral faces. The designer may reference this shell for example for matching a fillet. This model enables to follow the shell evolution in order to be able, during model design, to identify the involved shells, then, during reevaluation, to identify the effective shells (in the current instance) corresponding to the referenced shell. Then, a persistent naming mechanism is proposed for vertices, edges and faces, like for the various levels of wires and shells aggregates. In addition to the possibility of referencing these various levels of aggregates, the explicit history representation in the shell graph enables to reference another level of aggregation. The aggregation of all the shells having had a common historical ancestor. Thus, for example, the abstraction of final shell C1.f in Figure 7 will exist and could be referenced, even in case of split of this shell, since at each level of aggregation the history is preserved.

The proposed matching method is a global method which used a recursive backward-forward traversal algorithm that enables to match restricted set of entities. The entities characterization is based on topological neighborhoods weighted by the geometry, which enables more robust matching mechanism than by a simple use of topological neighborhoods. Finally, contrarily to Kripiac's approach, the use of bi-valued links between each node of the shell graphs permits an explicit representation of continuous metrics of matching. The values associated to this links define probabilities of mutual inclusion of the weighted topological neighborhoods, which are used in the successive modeling steps in order to avoid recalculate several times the same comparisons.

Unfortunately, explicit representation of the various level of aggregation implies an overcost both in spatial and temporal complexity, in order to calculate the connected components used to create the historical structure.

## 5. CONCLUSION

In this paper, we have presented a state of the art about the well known persistent naming problem in parametric modeling systems. We have identified five common concepts that result from most of the studies on this domain. First, the distinction between invariant and contingent entities. Second, the naming mechanisms used for invariant entities resulting from constructive gestures which involve two dimensional topological structures as input parameters. Third, the invariant entity names used to name contingent entities. Fourth, the capability to trace the face history either using a dedicated structure, or by means of a mechanism that propagates attributes from an entity to any entities that result

from it. Other entities are named by reference to faces. Fifth, the global software architecture that results from most contribution and that we call the three layers architecture.

We have also proposed two orthogonal criteria for classifying persistent naming approaches. The first criterion refers to the use of pure topology or of both geometry and topology to disambiguate entities of polyhedral model in the naming phase. The second criterion refers to the locality or globality of the matching method during the reevaluation phase. This survey leaves open two major issues which are currently subject of a number of researches.

- How to ensure that the various techniques used to disambiguate entities in the curved domain will lead to intuitive and predictable behavior when the topology of the model change ?
- Is it possible to define families of non linear geometric models such that parametric reevaluation could always be performed in a deterministic way ? Various recent works propose to restrict "*a priori*" either the variation ranges of the parameters [23], [20], [11], or the naming domain [3] in order to simplify the matching problems during reevaluation phase.

## 6. REFERENCES

- [1] Agbodan, D., Marcheix, D., Pierra, G. A Data Model Architecture For Parametrics in Journal for Geometry and Graphics, Vol.3, N°.1, pp.17-38, 1999.
- [2] Agbodan, D., Marcheix, D., Pierra, G. Persistent Naming for Parametric Models in WSCG'2000, Vol., pp.17-38, 2000.
- [3] Bidarra, R., Bronsvort, W.F., Semantic feature modelling in Computer-Aided Design Vol. 32 pp. 201-225, 2000
- [4] Bouma, W., Fudos, I., Hoffmann, C.M., Cai, J., Paige, R. Geometric constraint solver in Computer-Aided Design, vol. 27, n° 6, pp 487-501, June 1995.
- [5] Fricaud, Y., Matra Datavision private communication.
- [6] Capolyas, V., Chen, X., Hoffman, C.M. Generic naming in generative, constraint-based design in Computer-Aided Design Vol. 28 pp. 17-26, 1996.
- [7] Chen, X. Representation, Evaluation and Editing of Feature-Based and Constraint-Based design. Ph.D. thesis, Department of Computer Sciences, Purdue University, West Lafayette, Indiana, 1995.
- [8] Delmas, R. Nomination persistante d'entités topologiques dans les modèles paramétriques. DEA report LISI-ENSMA 2001.
- [9] Hoffmann, C.M, "On semantics of generative geometry representations", 19<sup>th</sup> ASME Design Automation Conference, New York, New York, USA, pages 411-420, 1993.
- [10] Hoffmann, C.M., Juan, R. EREP: an editable high-level representation for geometric design and analysis in Technical Report CER-92-24, Department of Computer Sciences, Purdue University, West Lafayette, Indiana, 1993.
- [11] C.M. Hoffmann and K-J. Kim, "Towards valid parametric CAD models", Computer-Aided Design, Vol. 33, N° 1, pages 81-90, 2001.

- [12] Kripac, J., "Topological ID System — A mechanism for persistently naming topological entities in history-based parametric solid models" Ph.D. Thesis, Czech Technical University, School of Electrical Engineering, Department of Computer Science, Czech Republic, Prague, 1994.
- [13] Kripac, J. A mechanism for persistently naming topological entities in history-based parametric solid models (Topological ID System) in Proceedings of Solid Modeling'95, Salt Lake City, Utah USA, pp.21-30, 1995.
- [14] Kripac, J., "A mechanism for persistently naming topological entities in history-based parametric solid models", Computer-Aided Design, Vol. 29, N° 3, pages 113-122, 1997.
- [15] Laakko, T., and Mäntylä, M., "Incremental constraint modeling in a feature modeling system", Proceedings of EUROGRAPHICS'96, Computer Graphics forum, Vol. 15, N° 3, Poitiers, France, pages 366-376, 1996.
- [16] Matra Datavision Group, <http://www.matra-datavision.fr>.
- [17] Open CASCADE <http://www.opencascade.org> or <http://www.opencascade.com>
- [18] Pierra, G., Potier, J.C., Girard, P. The EBP system: Example Based Programming for parametric design, Workshop on Graphic and Modeling In Science and Technology, Coimbra, Springer Verlag. 27-28 June 1994.
- [19] Pierra, G., Ait-Ameur, Y., Besnard, F., Girard, P., Potier, J.C. A general framework for parametric product model within STEP and Part Library in European Conference Product Data Technology, London, 18-19 April 1996.
- [20] Raghothama, S., and Shapiro, S., "Boundary Representation Deformation in Parametric Solid Modeling", ACM Transactions on Graphics, Vol. 17, No. 4, pages 259-286, October 1998.
- [21] Raghothama, S., Shapiro, V. Boundary Representation Variance in Parametric Solid Modeling in Report SAL 1997-1, Spatial Automation Laboratory, University of Wisconsin-Madison, 1997.
- [22] Shah, J.J., Mäntylä, M. Parametric and feature-based CAD/CAM: Concepts, Techniques, Applications, John Wiley and Sons Inc., july 1995
- [23] Shapiro, V., and Vossler, D.L., "What is a parametric family of solids", Solid Modeling'95, Proceedings of the 3<sup>rd</sup> ACM symposium on Solid modeling and applications, Salt Lake City, Utah USA, pages 43-54, 1995.
- [24] Solano, L., Brunet, P. Constructive Constraint-based model for parametric CAD systems in Computer-Aided Design, Vol.26, N°8, pp.614-621, 1994.
- [25] J. Wu, J., Zhang, T., Zhang, X., and Zhou, J., "A face based mechanism for naming, recording and retrieving topological entities", Computer-Aided Design, Vol. 33, N° 1, pages 687-698, 2001.