

## **Towards Classification and Automatic Detection of Feature Interactions**

**Rafael Bidarra<sup>1,2</sup> and Willem F. Bronsvooort<sup>1</sup>**

<sup>1</sup> **Faculty of Technical Mathematics and Informatics**  
**Delft University of Technology**  
**Delft, The Netherlands**  
**e-mail: bidarra@twi.tudelft.nl**  
**bronsvoort@twi.tudelft.nl**

<sup>2</sup> **Department of Mechanical Engineering**  
**Faculty of Sciences and Technology**  
**New University of Lisbon**  
**Lisbon, Portugal**  
**e-mail: rafa@mail.fct.unl.pt**

### **Abstract**

Current feature-based modelling systems fail to adequately maintain feature semantics, mainly due to an incomplete perception of feature interaction phenomena. In this paper a formal definition of feature interactions is derived, in order to identify the scope of situations to manage. A taxonomy for interactions is proposed that encompasses a wide range of situations, with either design or technological relevance. This is regarded as a first step in the development of flexible and powerful interaction detection and reaction mechanisms, aimed at improving the level of assistance provided to the user of a feature modelling system.

### **1. Introduction**

Feature modelling systems offer the possibility to build a product model with features. It is now generally agreed upon that features should not only contain shape information, as the modelling entities in geometric modelling systems, but also functional information. Stated differently, a feature should have a certain meaning. An example is that a cylindrical blind hole should have a circular top face, a cylindrical side face, and a circular bottom face, and, in addition, that the first face should not be on the boundary of the modelled object, whereas the two other faces should be.

However, in many of the current systems that are called feature modelling systems, very little is done to maintain the meaning, or validity, of features. If, for example, a slot feature is inserted into a model with a cylindrical blind hole feature in such a way that the bottom face of the hole is no longer on the boundary of the object, the feature is no longer a cylindrical blind hole, but has become a cylindrical through hole instead, and its functional meaning has thus changed. Modelling systems that fail to notify this, are in essence only geometric modelling systems, but no real feature modelling systems, because they do not maintain the meaning of features.

One approach to maintain the meaning of a feature is by using validation constraints. These are part of the definition of a feature and specify, for example, the kind of properties mentioned for the cylindrical blind hole. At the creation of a feature, and at subsequent modelling steps, these constraints can be checked to see whether the feature is still valid (Dohmen et al. 1996). If it is detected that a feature is no longer valid, the modelling operation that causes this invalidity can simply be rejected and forbidden to the user, and he can then take an appropriate action, for example, changing the parameters of a feature, or even taking a feature of a different type.

However, this scheme seems too rigid. The least that is desirable is that the user gets a good explanation on what has caused the invalidation of a feature. A further improvement would be that he gets hints on how to avoid or overcome the problem. Ideally, the system automatically adapts the model to get a valid model again in cases this is possible and desirable, although this should probably not be done without consulting the user. In the previous example, the blind hole might be automatically converted into a through hole, if the user permits this.

Most validity violations are caused by feature interactions, which arise from modelling operations such as the creation of a new feature or the modification of an existing feature. It is therefore important to get insight in feature interaction phenomena, so that all relevant interaction situations can be detected, classified and handled in an appropriate way. This paper is a first step in this direction.

First, in section 2 a survey is given of previous research contributions dealing with feature interactions. Next, in section 3 a formal definition for feature interaction is derived. Based on this definition, in section 4 a taxonomy of feature interactions is presented and illustrated; the taxonomy is based on both design and

technological criteria. Finally, in section 5 some conclusions about the utility and application of the above concepts are drawn.

## 2. Previous research on feature interactions

Researchers in the feature modelling field, taking either a design by features or a feature recognition approach, have often pointed out the convenience of keeping track of the model structure in terms of the relations among its features, in addition to that provided by the low-level, evaluated topologic/geometric entities. A variety of structures has been proposed, expressing insertion, adjacency, connectivity or similar relationships among the features of a model. Some of them were based on a rigid, CSG-like, parent-child relationship, yielding a tree-structured model; many others adopted a general graph structure, in order to better capture feature relations in more complex models (Kyprianou 1980) (Henderson 1984) (Luby et al. 1986) (Shah and Rogers 1988).

More attention has been paid to the nodes in the graphs (the features) rather than to the edges (the relations between them). In this way, questions like “*which features are in the model?*” or “*how should they be represented?*” have received more research efforts than others, such as “*which relationship is there among those features?*” or “*how can this relationship be precisely defined, detected and represented?*”. A positive contribution in this respect was given by Pratt (1988) with the first definition of an interaction relationship between two features. This was based on the notions of *effective volume of interaction (EVI)* and *actual volume of interaction (AVI)*, and a feature graph was proposed to represent such relations on a form feature model. Although limited in scope, in part due to the lack of a convenient counterpart at the geometric representation level, this scheme was indeed able to capture and represent simple interactions between features.

An informal definition for feature interactions was presented by Shah (1991), relating them to intersections between entities of two or more features such that either the shape or semantics of a feature are altered from the generic definition. A number of typical feature interactions was also illustrated; from their analysis, he concludes that unconditionally preventing such occurrences would excessively constrain the designer activity, and suggests that the modelling system should detect and react to them according to pre-defined procedures.

da Silva et al. (1991) proposed a strategy for explicit representation of relations between cavity features, in order to perform manufacturability reasoning. They distinguish between interacting and interfeature relationships. The latter are equivalent to geometric constraints now commonly available in modelling systems; the former are actually no more than simple attach relations between feature constituents. A major limitation of their approach lies in that both the relationships and the feature constituents themselves have to be extracted from or detected in the geometric model, which is virtually impossible for intersecting features in general.

Karinthi and Nau (1992) addressed some interaction problems with a formal perspective, defining an algebra of features. Its purpose is to generate all possible alternative configurations of a feature model in terms of manufacturing features, in order to provide better input for a process planning system. They assume that most of those alternatives are due to feature interactions, although, in practice, the restricted algebra implemented either does not yield all possible interpretations, or generates some that are not suitable for manufacturing purposes.

A methodology for the analytical detection of geometric interactions between features was proposed by Talwar and Manoochchri (1994). The algorithms presented take all the necessary input from the data stored in a manifold boundary representation model and require an intensive, low-level query and traversal. These algorithms provide a primary classification of feature interactions (e.g., contained, touching, intersecting, etc.) and are mainly intended for use in downstream, rule-based manufacturability analysis systems.

Suh and Ahluwalia (1995) approached feature interaction problems from the perspective of incremental feature generation. They propose to analyze each new shape, created in the model with a set operation, identifying the scope of its interaction and the actual feature generated. In addition, changes caused by the insertion into previously existing features are detected, and these are redefined accordingly. This method is suitable for the detection of a limited set of interaction classes only, mainly because identified features are regarded as open face sets in the evaluated geometric model. Furthermore, interaction cases having more than two interacting features are not dealt with.

Rossignac (1990) alerted for the difficulties and inconsistencies that could arise from a naive interpretation of usual editing commands on feature models, mainly due to the occurrence of feature interactions. His analysis provides a good insight into some high-level, design-oriented validity issues of feature models, although some of the problems pointed out only occur with a CSG-based representation of features. He also proposes the use of SGCs (selective geometric complexes) as a representation scheme that would facilitate the validation of simple interactions, but so far there are no known results in the literature that demonstrate its effective use.

Bidarra and Teixeira (1993) proposed a functional classification of feature interactions and proposed the use of interaction constraints in order to prevent features from being corrupted by a number of interaction classes. These constraints, together with a set of validity conditions, would then be integrated in the specification of each feature class, as a means of enhancing its feature instances with the higher-level, non-geometric information required for feature validation mechanisms.

It can be concluded that the perception of feature interactions that has been used until now is either too abstract or too detailed: the former is hard to implement and put into effect, whereas the latter fails to capture important aspects of feature semantics. Most often, difficulties found with feature interaction phenomena arise from approaching them mainly at the geometric level of the feature model. Therefore, it seems clear that adequate feature interaction management requires higher-level, functional information in the product model to be considered as well. This, in turn, can only be achieved based on a precise definition for feature interactions, able to capture all relevant situations.

### 3. Feature interactions: from an intuitive notion to a precise definition

This section presents a formal definition for the interaction relationship between features. Its derivation is carried out by integrating several contributions from previous research results, with the goal of precisely identifying a wide range of feature interactions with both modelling and technological relevance.

So far, the scheme presented by de Kraker et al. (1995) is the most promising and powerful, yet conceptually simple, proposal on which to ground a sound definition for feature interactions. In the prototype system SPIFF, a feature class can be specified in a generic feature definition, which includes the *nature* of its *feature volume* (FV), denoted as either P or N according to its additive or subtractive character, a description of its *feature boundary* (FB) in terms of *feature elements*, and a set of *constraints* describing feature validity conditions. Among these, *attach constraints* express a specific property of some feature elements, describing where and how a feature instance should be attached to the other ones in the model. With the instantiation of a feature, each of its attach constraints is bound to a feature element of some feature already present in the model, and mapped to a primitive coplanarity geometric constraint. These constraints are then solved and maintained by the system, using degrees of freedom analysis (Kramer 1992).

Another important aspect of feature validity is specified in a generic feature definition by means of so-called *semantic constraints* (Bidarra and Teixeira 1993). Semantic constraints describe the behaviour of a canonical feature and specify how each feature instance is allowed to deviate from that canonical behaviour, stating the extent to which its feature elements should, or should not, belong to the model boundary.

#### 3.1. The overlap relationship

As already mentioned, careful attention has to be paid to the actual way a feature is being attached or, in general, edited in a feature model in order to preserve not only its validity, but also that of the other features already there. Therefore, as a first step in this goal, we should observe that the actual overlapping among those features has an essential role in the occurrence of interactions; actually, no interaction takes place between two features if they do not share somehow a common region of space.

For this purpose, the notion of the *interaction extent* (IE) between two features  $F_1$  and  $F_2$  has been proposed and defined in (Bidarra and Teixeira 1994) as

$$IE(F_1, F_2) = FV(F_1) \cap FV(F_2)$$

where  $FV(F_i)$  denotes the feature volume of  $F_i$ .

Based on this, the *overlap relationship* between two features  $F_1$  and  $F_2$  was defined by Bidarra (1994) as follows:

$$F_1 \leftrightarrow F_2 \Leftrightarrow IE(F_1, F_2) \neq \emptyset$$

The dimensionality of a non-empty interaction extent was then used to define the *interaction mode*: whenever the IE is 3-dimensional, we have a *volumetric interaction mode* (VI); otherwise, the two features exhibit *boundary interaction mode* (BI).

#### 3.2. The dependency relationship

One main advantage of attach constraints is that, as a result of the generalization of conventional parent-child relations between features, a feature instance may have attaches to feature elements of several features already present in the model (Dohmen et al. 1996). In addition, the hierarchical relation they establish between

feature elements may be transposed to a hierarchical relation between the respective features, defining the orientation of each edge in the Feature Attachment Graph (FAG). FAG edges are oriented from the feature that is attached to the one to which the attachment is made, as depicted in Figure 1 for an example model.

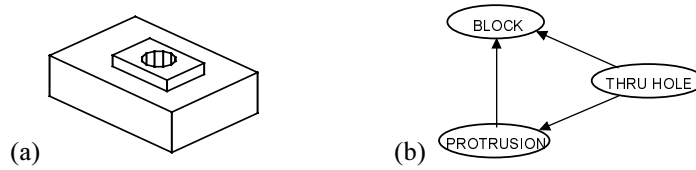


Figure 1 - Model with three features (a) and the corresponding Feature Attachment Graph (b)

Based on the attach information represented in the FAG, we can define the *dependency relationship* as follows:

*Definition:* There is a dependency relationship between features  $F_1$  (called active feature) and  $F_2$  (called passive feature) if there is a path in the FAG from  $F_1$  to  $F_2$ .

Whenever this path has only one edge (i.e.,  $F_1$  is attached to  $F_2$ ) we say that  $F_1$  has an *immediate* dependency on  $F_2$ ; if the only path between  $F_1$  and  $F_2$  has more than one edge, we say that there is a *non-immediate* dependency between them. In either case, we classify the *dependency type* by referring to the natures of both features, putting first the active feature's nature: NP, NN, PN or PP; for example, there is an NP immediate dependency between the thru hole and the block in Figure 1.

The occurrence of overlapping features may be due to an attach relationship between them, i.e. an immediate dependency. For meaningful attachments, the interaction mode between attached features is always a direct consequence of their natures:

- VI, whenever they have different natures (PN or NP immediate dependencies);
- BI, whenever they have the same nature (PP or NN immediate dependencies).

This regular behaviour is depicted in Table 1, showing a simple example model for each dependency type.

Table 1 - Typical overlapping situations between features with immediate dependency

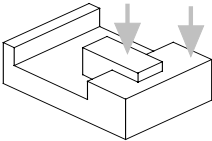
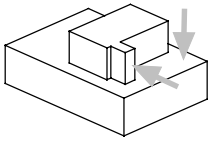
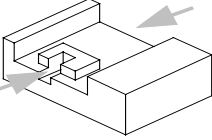
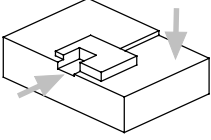
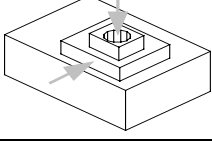
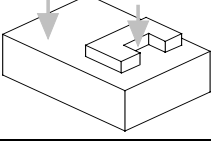
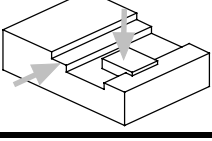
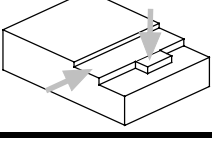
Volumetric Interaction		Boundary Interaction	
NP dependency		PP dependency	
PN dependency		NN dependency	

But overlapping may, of course, also occur between features that, although having no attachments linking any of their feature elements to each other, have a non-immediate dependency. This results in a large variety of possibilities with all four dependency types. Table 2 presents some typical examples for both VI and BI modes.

### 3.3. The interaction relationship

As can be concluded from the examples in Tables 1 and 2, the overlap relationship provides a necessary condition for the occurrence of feature interactions. However, Table 2 also suggests that some overlapping situations may occur that yield no interactions, i.e., overlap is not a sufficient condition for interaction. Such overlappings, of which an example is shown in Figure 2, have no practical relevance and utility from either model manipulation or feature semantics viewpoints: in Figure 2.(a) there is a PP overlapping between the protrusion and the block; in Figure 2.(b) there is an NN overlapping between the slot and the pocket.

Table 2 - Typical overlapping situations between features with non-immediate dependency

	Volumetric Interaction	Boundary Interaction
PP dependency		
NN dependency		
NP dependency		
PN dependency		

These situations occur only with special cases of overlappings between features that have a non-immediate FAG dependency and should, therefore, not be considered in the definition of interactions. To describe such cases, we first define, for each feature in the model, a partition on its feature boundary, according to

$$FB = FB^+ \cup FB^-$$

where

$FB^+$  represents the set of points of the FB that are *on* or *inside* the model boundary, and

$FB^-$  represents the set of points of the FB that are *outside* the model boundary.

It can be easily observed that the overlappings in Figure 2 are such that the  $FB^+$  of the respective features are disjoint.



Figure 2 - Overlapping PP (a) and NN (b) features without interaction

The dependency relationship accounts for a variety of situations that yield interactions. Yet, there is still another possibility for interactions to take place: overlapping may also occur between features that have no dependency in the FAG, i.e., between *independent* features. Examples of such situations are depicted in Table 3, again for all combinations of feature natures.

Obviously, in these cases, the information in the FAG cannot bring any insight into which is the active or the passive feature in the interaction. Moreover, in general, this cannot even be derived from a static analysis of the model: either of the interacting features could be responsible for the interaction, as shown by the NP/PN BI example of Table 3. Therefore, we need to extend these notions to also cover such situations. This can be accomplished taking into account that interaction situations are always a direct consequence of *some modelling operation on a particular feature* (e.g., insertion of a new feature or modification of an existing one). Thus, *active feature* and *passive feature* may be extended to designate the feature that causes the interaction and the one which receives it, respectively. In the remaining of this paper the notion of *interaction type* will be used to characterize an interaction relationship, with the same notation as before for the dependency type: active feature's nature before the passive feature's nature.

We can now establish our formal definition of the interaction relationship.

Table 3 - Typical overlapping situations between independent features

	Volumetric Interaction	Boundary Interaction
PP interaction		
NN interaction		
NP interaction		
PN interaction		

*Definition:* A modelling operation on a feature  $F_1$  (active feature) creates an interaction relationship with feature  $F_2$  (passive feature), denoted  $F_1 \rightarrow F_2$ , whenever it results in a situation for which the following conditions hold:

(a)  $IE(F_1, F_2) \neq \emptyset$

(b) if  $F_1$  and  $F_2$  have a non-immediate FAG dependency, then their  $FB^+$  intersect

From this definition, we can conclude that all overlapping situations between features with either an immediate FAG dependency (cf. Table 1) or no dependency at all (cf. Table 3) are always considered as interaction. Over-lappings between features with non-immediate FAG dependencies are subject to the restriction (b), which leaves out cases as those in Figure 2, while still capturing the actual interactions shown in Table 2.

#### 4. A Taxonomy for Feature Interactions

As it became clear from the previous section, feature interactions may have a very wide range of consequences and effects on a feature model. Although these may often be intended, in many situations they may affect the semantics of a feature, ranging from slight changes in actual parameter values to the complete suppression of its contribution to the model shape.

In this section, we propose a taxonomy for feature interactions, in conformity with the definition derived in section 3. Among the interaction classes presented, some have a clear technological connotation, whereas others are more biased towards direct capture of modelling or functional phenomena. For each interaction class, we give a full definition together with some examples, in order to illustrate its relevance and show typical situations.

The ultimate goal of the proposed taxonomy is threefold: (a) to make possible the development of appropriate detection mechanisms, (b) to allow specific potential reactions of the modeller to each interaction class, and (c) to provide the designer with the capability to selectively allow/disallow individual interaction classes, through the use of *interaction constraints* in feature generic definitions.

##### 4.1. Splitting interaction

*Definition:* volumetric interaction that splits the  $FB^+$  of a feature into two (or more) disconnected subsets.

This kind of interaction may have both functional and technological consequences, hence the need to detect it. On the one hand, the knowledge that a particular feature became split in the model gives useful hints on feature precedence in a process planning view of the model. On the other hand, the functional purpose of some feature

classes may also be affected by a splitting interaction, e.g. the stiffening function of a rib protrusion, or the sliding functionality assigned to a certain slot in an assembly view.

Splitting interactions always have their origin in some N feature and may, thus, occur due to either NN or NP interactions. NN splitting interaction is certainly the most relevant situation of splitting interaction. The active feature suppresses a subset from the  $FB^+$  of the passive feature, in such a way that this becomes split into two disconnected components (see Figure 3.(a)). Splitting interactions, however, may also occur with P passive features, provided they are not a primary block (i.e., they have some FAG dependency on some other feature), as illustrated in Figure 3.(b).

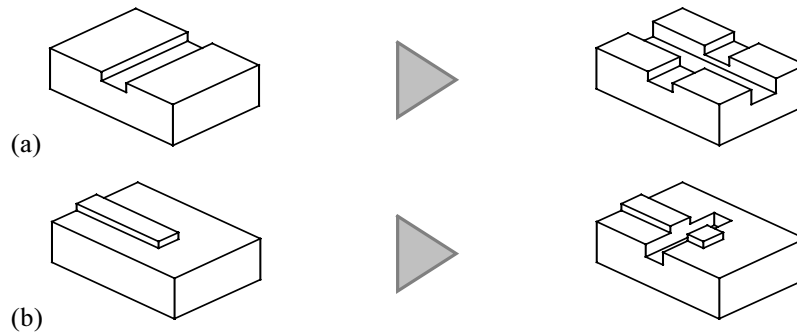


Figure 3 - Examples of NN (a) and NP (b) splitting interactions

#### 4.2. Disconnection interaction

*Definition: interaction that causes the FV of a P feature (or part of it) to become disconnected from the remaining model.*

Disconnection interactions are important to deal with, because they always result in topologically disconnected components in the part being designed, which makes it also unfeasible from the manufacturing point of view.

A disconnection interaction may occur as a result of either the insertion (a) or the modification (c) of an N feature, as depicted in Figure 4; in any case, the result (b) is always a nonsense, unrealizable model, because of the P feature's attach loss.

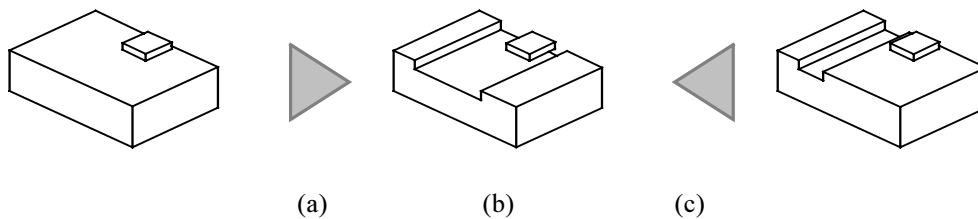


Figure 4 - Examples of disconnection interaction

Disconnection interactions may also take place when the actual dimension parameter values of the N active feature are too large, causing a P passive feature to become topologically disconnected, as shown in Figure 5.

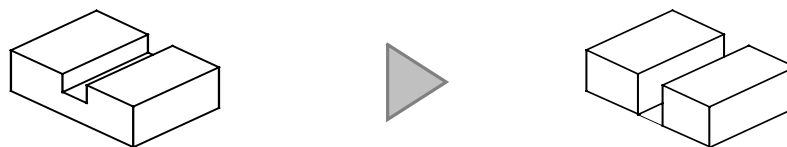


Figure 5 - Example of disconnection interaction on a primary block due to slot depth enlargement

#### 4.3. Clearance interaction

*Definition: interaction causing (partial or total) obstruction of N features.*

Clearance interactions have a clear technological connotation: they usually constrain the machining process possible for a certain N feature in several ways, e.g. approach tool direction, machine accessibility and tool path clearance. Some subtractive feature classes (e.g. slot or step classes) pose strong manufacturability requirements in terms of restricting positive volume intrusion or interference into their own FV.

Clearance interactions may take place in either VI or BI modes, as depicted in Figure 6.



Figure 6- Examples of volumetric (a) and boundary (b) clearance interaction

#### 4.4. Closure interaction

*Definition: interaction that causes some N feature volume(s) to become a closed void inside the model.*

In a sense, closure interactions are an extreme case of clearance interactions, in that they cause a total inaccessibility of an N feature. Therefore, their detection is even more important, from both the functional and the manufacturing viewpoint.

Single closure interactions cause the whole FB of the N feature to become part of the model boundary (i.e.  $FB^- = \emptyset$ ). Multiple closure interactions, however, may occur without causing  $FB^- = \emptyset$  for any N feature; this is the case in the model shown in Figure 7, where the closure event is shared by the 3 previously interacting features.

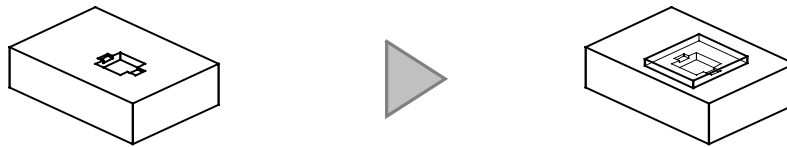


Figure 7 - Example of multiple closure interaction

#### 4.5. Absorption interaction

*Definition: volumetric interaction that causes a feature to cease completely its contribution to the model shape (i.e., no part of its  $FB^+$  remains actually on the model boundary).*

Absorption interactions appear to be clearly undesirable, because they turn useless a feature that was previously created and inserted into the model.

The simplest case of absorption interaction occurs whenever the  $FB^+$  of a feature becomes completely enclosed within the FV of an N feature, as depicted in Figure 8.

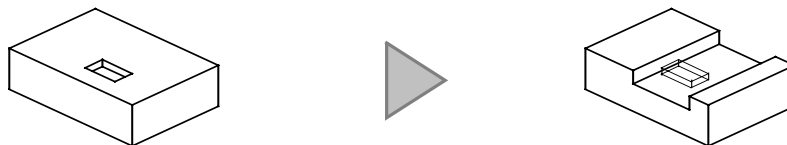


Figure 8 - Example of single absorption interaction

Absorption interactions, however, may also occur in a multiple or distributed fashion, i.e. when some subset of the passive  $FB^+$  has already been suppressed from the model in a previous modelling step; in such cases, the absorption is carried out by just suppressing from the model the remaining part of the passive  $FB^+$ . This means these absorptions may take place without the passive  $FB^+$  being totally contained in either of the existing FVs.

#### 4.6. Geometric interaction

*Definition: volumetric interaction that causes some dimension parameter to loose its correspondence to the actual feature geometry.*

Geometric interactions may modify the functionality intended for a particular feature, in the sense that some actual parameter values it exhibits are different from those initially specified by the designer. This mismatch between *actual* and *nominal* (or intentional) parameter values may also be relevant for a process-planning view (e.g., availability of required tools or convenient precedence for machining features). An example of these is a decrease in blind slot length due to a step insertion, as depicted in Figure 9.



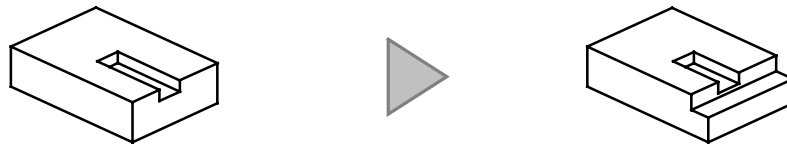


Figure 9- Examples of geometric interactions

#### 4.7. Transmutation interaction

*Definition: interaction that causes a feature instance to exhibit the behaviour of another feature class.*

The detection and control of transmutation interactions may be of particular importance from a process planning point of view, provided that the passive feature may allow (or require) a machining process different from what would be expected, were it in the canonical form. Similarly to what geometric interactions do at the parameter level, transmutation interactions capture the actual behaviour of feature instances at the feature boundary level, in order to allow a closer assistance to the designer whenever his previous intention is in risk of being overruled. An example of this is given in Figure 10: a blind slot is turned into a thru slot due to a step insertion.

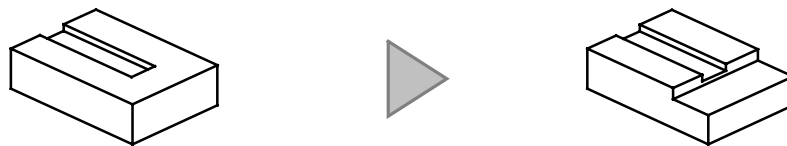


Figure 10 - Transmutation of a blind slot into a thru slot

#### 4.8. General interaction

*Definition: interaction shown by a feature instance in all interaction situations not covered in the previous interaction classes.*

General interactions make up the catch-all class for interactions that may somehow affect a feature instance. The functionality of a feature may be modified in a variety of ways due to general interactions, as a result of the changes in the composition of its boundary caused by overlapping. Similarly to other interaction classes, the occurrence of general interactions does not necessarily mean that some feature has become invalid, hence the necessity for their detection; actually, these may cover a wide range of situations that are intentionally desired in the model. Figure 11 presents two examples of models with several features in general interaction.



Figure 11 - Examples of general interactions

### 5. Conclusions and future work

The occurrence of feature interactions as a result of the incremental manipulation of feature models has been addressed in this paper. A formal definition of feature interactions was proposed, as an indispensable condition for delimiting a comprehensive domain on which to focus the development of an interaction management mechanism. This definition covers all meaningful interaction situations, and is directly mappable to a detection algorithm. Based on this, a taxonomy for feature interactions was presented and illustrated, taking into account relevant modelling and technological criteria.

An immediate goal in this research is the detection of each interaction class, as a result of monitoring each modelling operation performed in the feature modeller. This classification, under implementation in the SPIFF modelling system, a prototype multiple-view feature modeller developed at Delft University of Technology, is based on the analysis of the interaction extent and feature natures, the evaluation of semantic constraints, and a variety of topologic and geometric queries to the underlying cellular model.

Our next research goals include providing the user of SPIFF with a choice of reactions and suggestions, whenever each of the above interaction situations is detected (e.g., readjust some feature parameters,

reconsider its location, reject the feature operation, suggest a change in attachments or perform a change in feature class). In addition, a new type of *interaction constraints* may be provided for use in feature generic definitions, enhancing feature classes in a library to selectively allow/disallow particular classes of interactions.

### Acknowledgements

Rafael Bidarra's work is supported by the Praxis XXI Program of JNICT, under grant BD3279/94.

### References

- Bidarra, R. (1994) "Uma Abordagem Semântica aos Problemas de Interação em Modelos Baseados em Características de Forma", *Provas de Aptidão Pedagógica e Capacidade Científica (Master's Thesis; in Portuguese)*, Universidade de Coimbra
- Bidarra, R. and Teixeira, J.C. (1993) "Intelligent Form Feature Interaction Management in a Cellular Modeling Scheme", *Proceedings of the Second Symposium on Solid Modeling and Applications*, Rossignac, J.R., Turner, J and Allen, G. (Eds.), Montreal, ACM Press, pp. 483-485
- Bidarra, R. and Teixeira, J.C. (1994) "A Semantic Framework for Flexible Feature Validity Specification and Assessment", *Proceedings of the ASME International Computers in Engineering Conference and Exhibition*, Minneapolis, Vol. 1, pp. 151-158
- Dohmen, M., de Kraker, K.J. and Bronsvort, W.F. (1996) "Feature Validation in a Multiple-View Modeling System", submitted for publication
- Henderson, M.R. (1984) "Extraction of Feature Information from Three Dimensional CAD Data", *PhD Dissertation*, Purdue University, West Lafayette, IN
- Karinthi, R.R. and Nau, D. (1992) "An Algebraic Approach to Feature Interactions", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 14, No. 4, pp. 469-484
- de Kraker, K.J., Dohmen, M. and Bronsvort, W.F. (1995) "Multiple-way Feature Conversion to Support Concurrent Engineering", *Proceedings of the Third Symposium on Solid Modeling and Applications*, Hoffmann, C., Rossignac, J. (Eds.), Salt Lake City, ACM Press, pp. 105-114
- Kramer, G.A. (1992) "Solving Geometric Constraint Systems: a Case Study in Kinematics", The MIT Press, Cambridge, Mass.
- Kyprianou, L.K. (1980) "Shape Classification in Computer-aided Design", *PhD Dissertation*, Cambridge University, UK
- Luby, S.C., Dixon, J.R. and Simmons, M.K. (1986) "Designing with Features: Creating and Using a Features Data Base for Evaluation of Manufacturability in Castings", *Proceedings of the ASME International Computers in Engineering Conference and Exhibition*, Chicago, IL
- Pratt, M.J. (1988) "Synthesis of an Optimal Approach to Form Feature Modelling", *Proceedings of the ASME International Computers in Engineering Conference and Exhibition*, San Francisco, CA, Vol. 1, pp. 263-274
- Rossignac, J.R. (1990) "Issues on Feature-Based Editing and Interrogation of Solid Models", *Computers & Graphics*, Vol. 14, no. 2, pp. 149-172
- Shah, J.J. (1991) "Conceptual Development of Form Features and Feature Modelers", *Research in Engineering Design*, Vol. 2, pp. 93-108
- Shah, J.J. and Rogers, M.T. (1988) "Expert Form Feature Modelling Shell", *Computer-Aided Design*, Vol. 20, no. 9, pp. 515-524
- da Silva, R., Wood, K.L. and Beaman, J.J. (1991) "Interacting and Interfeature Relationships in Engineering Design for Manufacturing", *International Journal of Systems Automation: Research and Applications*, Vol. 1, pp. 263-286
- Suh, H. and Ahluwalia, R. (1995) "Feature Modification in Incremental Feature Generation", *Computer-Aided Design*, Vol. 27, No. 8, pp. 627-635
- Talwar, R. and Manoochehri, S. (1994) "Algorithms to Detect Geometric Interactions in a Feature-Based Design System", *Proceedings of the ASME Design Engineering Technical Conferences*, Minneapolis, MN