In *Sustainable Technologies in Manufacturing Industries*,
The Proceedings of the Irish Manufacturing Committee's 14th Conference,
Editors J Monaghan and C G Lyons, Trinity College, Dublin 3-5 September 1997, pp115-124

INTENT-DRIVEN REASONING PRIORITIES IN A FEATURE-BASED VALIDATION SYSTEM

M. S. Hounsell and K. Case

Manufacturing Engineering Department, Loughborough University.
Loughborough, LEICS, LE11 3TU, England.

ABSTRACT

Feature-based representation validation seeks to find means to verify feature-based representations in order to guarantee that feature's expected behaviours are met and that applications that use the representation can be sure of the correctness of the feature-related data.

To achieve this, a clear definition of features and their behaviour is needed but cannot be found in the literature. Instead of proposing yet another feature definition, an attempt was made to define some basic common-sense characteristics for (prismatic) features that could be tested, analysed and manipulated. These characteristics are called Intents because features are said to be the carriers of designer's intents. Feature-based Designer's Intents (**DI's**) proved to be essential to the validation framework because they define the scope of the Feature-based Modelling **(FBM)** utilisation. Also, some DI's establish clearly the geometric-dependent behaviour of features and were found to be closely related to validation. A prototype system called **FRIEND,** an acronym to **F**eature-based **R**easoning system for **I**ntent-driven **EN**gineering **D**esign, was implemented to perform feature-based representation validation.

This paper details Designer's Intents (DI's) in the context of deign-by-feature representation validation, presents Morphological Functional and Volumetrical DI's, their semantics and their priority organisation inside the validation mechanism, as it was implemented within **FRIEND.**

1. INTRODUCTION

Feature-based Modelling **(FBM)** is considered to be the underlying technology for the next generation of Computer-Aided Design **(CAD)** systems and Concurrent Engineering environments in the same way as Geometric Solid Modelling **(GSM)** is considered to be the underlying technology for existing CAD systems. Some advantages of FBM are the use of a more friendly environment, meaningful entities (see Figure 1) and manipulations, the ability to store information beyond geometry and, the consequent possibility of integration with other engineering applications such as Manufacturing, process planning, etc. In many respects, FBM is considered an evolution of GSM mainly because FBM usually utilises GSM as its core subsystem and adds another layer of information beyond the geometric one.

However, one basic element that makes GSM so well established, important, popular and powerfu1, namely *Geometric Validation*, lacks a sibling in the FBM world. This is so because features also add a layer of complex semantics which make it difficult to establish measurable means and are subjective to implement. Feature-based representation validation is very important because it

is the process responsible for guaranteeing the delivery of a valid representation (and therefore verified, useful and misrepresentations free) to a downstream application.

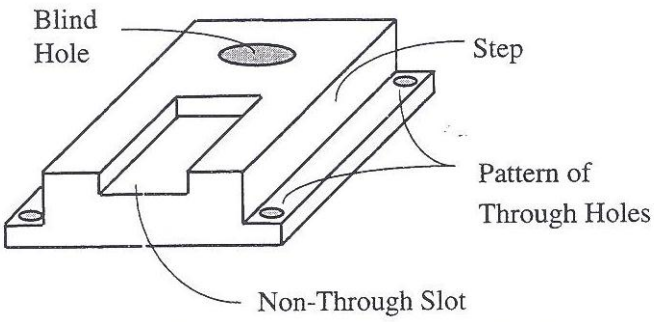

Figure 1: A featured-based model example

This paper presents how validation can be applied in design-by-feature systems driven by Feature-based Designer's Intents (DI's) as the means for the domain characterisation. The definitions of abstract DI's and DI's are first presented followed by the identification of Volumetrical DI's that implement the Morphological Functional Abstract DI. Sets of reasonings are identified and are prioritised to perform the validation. These sets and prioritisation were observed while implementing a design-by-feature prototype system called **FRIEND**, a **F**eature-based **R**easoning system for **I**ntent-driven **EN**gineering **D**esign.

2. VALIDATION OF FEATURE-BASED MODELS

Representation Validation is a set of verifications and the execution of appropriate corrections or "revalidation operations" on the feature-based model to analyse its conformity with the previously established domain-dependent feature's concept (its role as a 3D modelling technique, common sense expected behaviour and extra meanings). Therefore, it is called *conceptual feature validation* [1]. It subsumes other existing representation validations (geometric- based analysis performed on a feature-based model) because it validates the feature's representational semantics. Feature-based representation validation is a necessary step because, ultimately, downstream applications will work on the representation and they expect correct data.

The following are the elements necessary to conceive a feature-based representation reasoning system (see Figure 2).

- Domain characterisation: features are defined by their underlying "intents" from different perspectives. This approach avoids establishing yet another definition for features. This characterisation must be made clear and verifiable.
- Validity conditions: define the means by which to verify if all features configured to a component comply with their expected intents (DI's).
- Revalidation operations: represent means to operate on the model and to turn representations into valid ones and are selected and requested by the "validity conditions".

It should be observed that the input is expected to be previously validated against the Geometric Solid Modelling (GSM) constraints (such as dangling-edge elimination and Euler-Poincaré analysis, [2]) and the reasoning is thereafter primarily concerned with feature-based data. Thus, the input to the Feature Validation System is the feature-based model and a valid geometric solid model.

It is understood that the process of validation is a loop. Once a particular intent is verified, the process analyses another intent on all features assigned to the model until all intents and all features were analysed. If a "revalidation operation" is performed, it creates a different scenario of features (hopefully a simpler one). The feature(s) involved in the revalidation operation and its (their) adjacencies is (are) then, once more, verified against all intents. When all intents are verified and no more new scenarios are produced, the validation process loop delivers the resulting feature-based valid model to a downstream application.

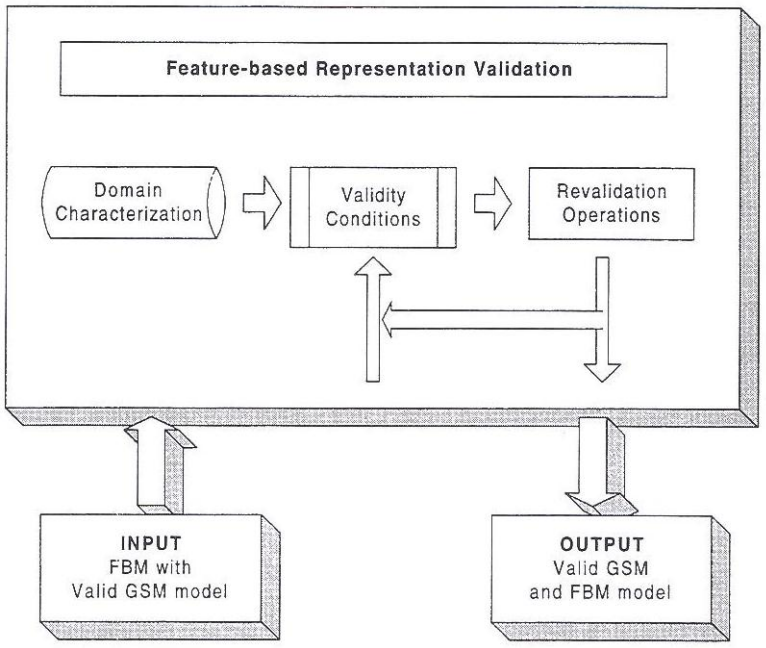

Figure 2: Feature validation reasoning framework

The output of the validation process will be a both valid geometric solid model and a valid feature-based model. This validated model can be then used by any application and no misrepresentation should exist from the selected DI's perspectives.

3. FEATURE'S EMBEDDED DESIGNER'S INTENTS

Capturing Feature-based Designer's Intents (DI's) at early stages of the design through a more user-friendly interface that includes a meaningful design vocabulary are properties of a FBM system that could allow more intelligent decisions and reasonings to be made and are considered "the only possible basis for Intelligent CAD" (**ICAD**) systems [3].

"Designer's intents are of high importance to be preserved but their understanding has a complicated nature" [4]. Although features are a proclaimed and accepted means of capturing and representing DI's, existing FBM systems do not deal with DI's as a major concern for three main reasons: Firstly, there still is a lack of a formal well-accepted definition for features and their role as a geometric modelling technique. Secondly, there is also the same lack of understanding of what DI's are, especially in the FBM context and; thirdly, identified intents are usually tied to the application in particular implementations.

Thus, it is herein interpreted that *"Feature-based Designer's Intents (**DI's**) are a wide variety of concerns that help decide on a specific geometric attribute or configuration to achieve (a set of) higher level functional requirements, called abstract designer's intents (**ADI s**). Designer's Intents*

(DI's) act as a bridge between the (set of) abstract designer's intents (ADI's) and the geometric model. DI's represent information which should be verified and maintained throughout the Detailed Design Process and could be used as constraints to drive the decision-making process of a downstream application. To some extent, it could be said that Feature-based Designer's Intents represent means to implement abstract designer's intents within the geometrical realm".

Abstract Designer's Intents (**ADI's**) include *morphological* functional, *theoretical* functional and *relational* functional ones. A taxonomy of DI's concerned with the feature-based geometric detail design phase for prismatic machining parts has been established and includes: Volumetrical, Parametrical and Geometrical DI's. This paper discusses the reasoning priority behind the implementation of *morphological functional* abstract designer's intents (ADI's) and volumetrical designer's intents (DI's) [5] that, although very important and intrinsic to feature-based design, have been neglected in the literature.

It is considered that with such a taxonomy and clear definition of DI, a set of agents can be conceived to effectively capture, represent and maintain intents through feature-based modelling. DI's can be expressed by relationships between features themselves or elements of the feature-based model such as features' faces (and their attributes) and features' parameters.

3.1 Morphological Functional Designer's Intents

Features have a *morphological functional* ADI. For instance, "if an application considers only functional morphological information, then the term form-feature can be used" [6]. The *morphological functional* ADI identifies shape characteristics that a feature must comply with and imprint on the part. If such an aspect is changed, the feature is possibly no longer of the specified type (say a *slot* or a *hole).*

3.2 Volumetrical Designer's Intents

Volumetric Designer's Intents (VDI's) implement *morphological functional* ADI within the geometrical realm and are concerned with the feature's expected geometric behaviour. Volumetrical DI's are to be considered specially, but not solely, when an interaction between feature volumes occurs. Intermediate states, delete operations and editing manipulations have direct influence over VDI's in design. To deal with VDI's the semantics of non-conflicting and conflicting interactions between features must be defined. These situations could result in normal, obsolete (redundant) or undesirable cases (such as *hollows* and *satellite* features).

Four VDI's can be observed in a feature-based model (see Figure 3):

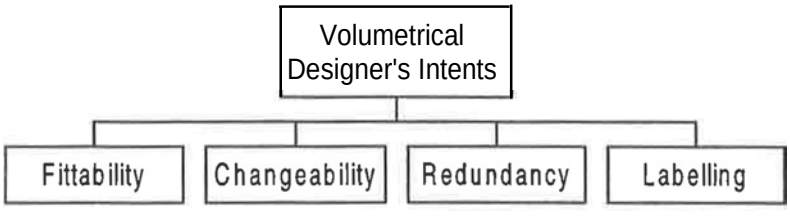

Figure 3: Volumetrical Designer's Intents.

Labelling VDI identifies the relationships between all feature's faces and their attributes. Every feature has a set of labelling relationships that is kept as the feature's *label.* Labelling is basically implemented by defining a template of *virtual* and *real* faces that bound the produced volume of a feature type. *Virtual* faces basically identify tooling external access directions and *real* faces identify surfaces to be imprinted on the part.

In addition to establishing a label-to-shape relationship, features are usually expected to imply a volumetrical behaviour, which is called the feature's *nature* by Lenau et. al. [7], of *adding* material (when it is said to have a *positive* volume) or *removing* material (when it is said to have a *negative* volume) from the stock. A feature's *nature* is identified by a Boolean operation (union for a *positive* volume and difference for a *negative* volume). The feature's *nature* implies that a change in the feature-based representation must result in a change in the volume and surface of the component being modelled. This feature's requirement and ability to change the existing model is called the **changeability** VDI. The changeability requirement invalidates *obsolete features* [8] that occur when a feature is completely inserted into another and has the same *nature*. However, it does not require that all the boundaries of the feature's produced volume should be shaped into the part.

A feature must have adequate parameters to exactly fit and define the intended form (in the same way as an edge is limited by its two exact ends, called vertices) thus, the feature must fit within the limits of where it is intended to be placed. This ability to fit is called the **fittability** VDI. The fittability requirement invalidates *feature's parameters made obsolete* [8] where feature's parameters do not describe exactly the extent of what it imprints on the part.

Furthermore, interesting and difficult situations arise when redundant intents are found. Features that have overlapping volumes usually present a **redundant** VDI. This is a feature interaction problem that has been receiving much attention in the literature as being of special difficulty to handle (see [9] and *The Contiguity Problem* in [8]).

4. DESIGNER'S INTENTS INTERRELATIONSHIPS

A prototype system called **FRIEND,** short for **F**eature-based **R**easoning system for **I**ntent-driven **EN**gineering **D**esign, has been implemented with special concern for the validation of feature-based geometric design representations [8]. A clearer definition of feature's semantics within **FRIEND** was achieved with the help of the *morphological functional* ADI and *volumetrical* DI's [5], briefly presented before. The verification reasoning is based on the spatial geometrical feature interaction (such as those that will be exemplified here: abuts, touches, inserted) applied at various levels, such as the feature volume and feature face levels [10].

Taken together, the feature's interaction and feature's data structure (which includes feature's *nature* and *label)* offer a vocabulary that permits a knowledge-based system to reason and validate the feature-based model. **FRIEND** represents all the relationships mentioned above and uses a declarative knowledge-based environment (rule-based system) to reason with them.

Special agents were identified that allows **FRIEND** not only to verify the model, but also maintain the model validity by operating on it. These "revalidation operations" include split, merge, delete and search label operations. The selective firing of these operations guarantee that **FRIEND** delivers valid representations from the *morphological functional* perspectives.

At first, the rules were conceived without any concern for their global organisation because the emphasis was on the usefulness and feasibility of the *conceptual validation* framework. However, after the implementation was working for most of the test cases, interesting self-organising interrelationships were observed, as follows.

4.1 Reasoning Aspects

Morphological functional ADI is considered to be closely related to feature's semantics and thus to feature-based representation validation. An analysis of VDI's suggests that there are basically two sources of concern that help describe the feature's behaviour: the volumetrical-dependent interaction aspect and the labelling-dependent aspect.

Volumetrical-dependent interaction aspects happen when the feature's *nature* is considered and when the "feature produced volume" (FPV, [8]) interact with each other at volumetrical or boundary level. The absence of this aspect means that no interaction analysis is performed whatsoever, another geometrical analysis is being performed or, the interaction is performed at a very low level (the face level).

Labelling-dependent aspects happen when the feature's *label* is the major affected element or when features' *labels* affect or determine the reasoning.

Volumetrical-dependent interaction aspects are related to changeability, fittability and redundancy VDI's while labelling-dependent aspects are related to labelling VDI's.

4.2 Reasoning Sets

The combination of these two concerns leads to four sets of reasonings according to whether or not they are part of the rule (see Table 1). For short, the volumetrical-dependent interaction aspect will be identified by V while the labelling aspect will be identified by L.

		Labelling-dependent Aspect (L)	
		With	Without
Volumetrical-dependent Interaction Aspect (V)	With	a)- L,- V	b)+ L,- V
	Without	c)- L, + V	d)+ L, + V

Table 1: Sets of Validation Reasoning.

These four sets of reasonings identify distinct and important situations when dealing with feature validation:

- Situations of type (a) are not considered to be feature-related reasonings because there seems to be no interesting feature-based reasonings when the feature's volumetric interaction, its *nature* and its *label*, are not important. However, they are included here because they are responsible for performing the geometric interaction scenario identification among features as well as performing other geometric reasonings defined by other applications. It can be said that situations of type (a) represent simply geometrical (-L, -V) analysis/reasonings but, do not include GSM validations.
- A situation of type (b) happens when, instead of volumetrical interaction, *labels* are the main focus of the reasoning, such as when the system is searching for the right *label* for a specific feature according to its faces' properties. It can be said that situations of type (b) implement simply labelling (+ L, -V) reasonings. Simply labelling rules include all those where low level interactions (face level) could result in a change in a feature's face property (from *virtual* to *real*, or vice-versa) and consequently a change in its labelling, regardless the feature's *nature.*
- A situation of type (c) happens when volumetrical reasonings and/or the feature's *nature*, despite the feature's *label*, are enough to fire an action such as when conflicting volumetrical intents *(hollows* or *satellite* volumes) appear in the model. It can be said that situations of type

(c) represent simply volumetrical (-L, +V) analysis. Simply volumetrical rules also include those when an incoming feature interacts with the stock material, regardless the former's *label*. This last reasoning example has priority because the stock material is considered the envelope of the whole component (and all its features) thus, any volumetrical analysis involving the stock would speed up the processing of the newly added feature.

- A situation of type (d) happens when both feature's volumetrical interaction and *label* determine the actions to be taken. To distinguish from previous sets they will be called here complex (+L, +V) reasonings as in "cut-out" cases (see item 4.3). All other further interactions between features, but the stock, are also considered as complex rules.

4.3 Priority

It was found that a priority scheme exists among the four situations in Table 1 in a way that every time a situation of higher priority occurs, it is dealt with immediately and in a descendent order of priority up to the point where there is no pending situation.

Within the same priority level, any sequence of rules can be expected to be fired according to a certain unpredictable behaviour that defines a declarative knowledge-based implementation (in opposition to a procedural implementation).

The priority found, from the highest to the lowest (see Figure 4), is:

1. **Simply geometrical** reasonings (type (a): -L,-V).
2. **Simply volumetrical** reasonings (type (c): -L, +V).
3. **Simply labelling** reasonings (type (b): +L,-V).
4. **Complex** reasonings (type (d): +L, +V).

In fact, the actual implementation has three separate rule-based files that reflect the division and hierarchy for the last three feature-based reasoning sets. Set (a) could have been implemented as rules as well but was implemented as functions for efficiency reasons. Also, the search for a feature's *label*, presented before as the search label operation, identifies the right *label* according to its faces' properties which could be in any orientation compared to the original template so, it was also implemented as a function for efficiency reasons.

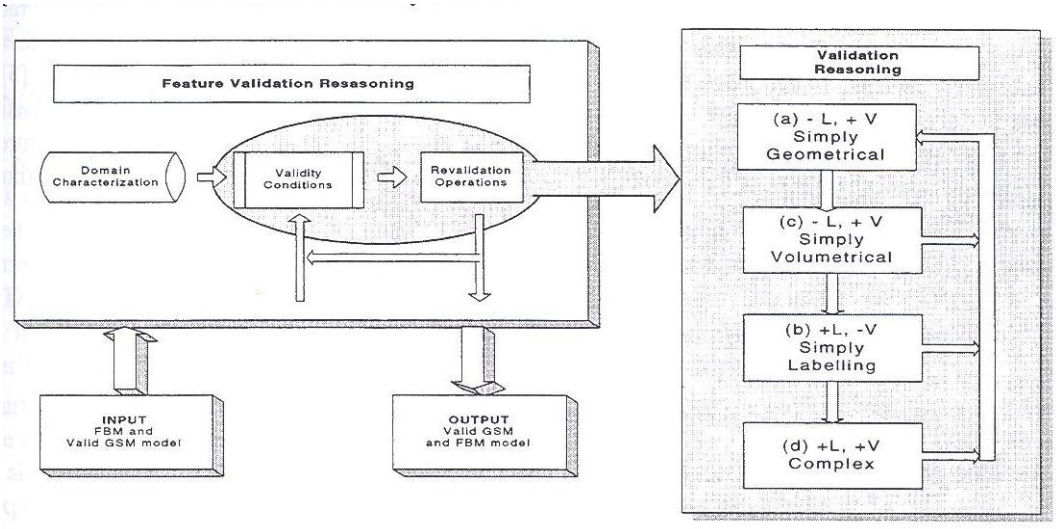

Figure 4: Sets of Reasonings

Set (a) performs GSM reasoning and generates the interaction scenario between features at various levels of interest (initially volumetric interaction up to face interactions, as it is requested [10]). The interaction scenario is then considered by the subsequent sets of reasonings. The first, afterwards, is the volumetric dependent reasonings (situations (c)). If there is enough information to deal with, the *labels* are then verified and (re)assigned if required. If the model is not yet valid then, there will be enough information with both *labels* and geometric interactions defined. In such case, face interactions are added and situations of set (d) are then considered. These situations already consider some designer's experience, product type and application's constraints.

Figure 4 shows that to implement the loop of reasoning in **FRIEND**, the validation reasoning consists of the four sets just described organised in a hierarchical fashion (as implied by the right-hand side of the figure). There is also a priority relationship among the situations mentioned above and the feature interaction identification level (Volumetrical, Boundary and, Face - [10]). The reasoning goes deep into the interaction level if it cannot reason with the information and interaction already available, and this is another reason why the scheme in Figure 4 is a loop.

4.4 Feature-Based Designer's Intents Reasoning Examples

A *simply volumetrical* reasoning (type (c)) is exemplified below. It is considered of type (c) because it uses the "match" volumetrical interaction (thus, it is +V) but not the *label* of the features (thus, it is -L):

IF
(Feature1 has a different *nature* from Feature2)
(Feature1's volume "matches" Feature2's volume)
THEN
Ask "Is Feature1 being used to delete Feature2 ?"
Answer YES, Deactivate Feature1, Deactivate Feature2
Answer NO, Deactivate Feature1 (it was a mistake !)
END-RULE

A *simply labelling* reasoning is exemplified below. If a face of a given feature "abuts" and is completely inserted into another feature's *real* face then, the former must be a *virtual* face [11]. Using reasonings such as this the labelling aspect can be maintained. If the template and the realisation do not match, the feature's *label* is invalid, and a "revalidation process" [5] called search label will then search for the right match. The search label process is responsible for keeping the label-to-shape relationship matching as defined by the template of every feature's type. It considers a feature interaction - "abuts"- at the face level, not at the volumetrical level (thus, it is -V), and immediately affects the feature's *label* (thus, it is +L):

IF
(Feature1’s face FF1 ‘abuts’ Feature2’s face FF2)
AND
(Feature1AND Feature2 are active in the model)
AND
(FF! has a *Real* code)
AND
(FF2 has a *Virtual* code)

THEN
Change FF1's code to *Virtual*
Apply labeling to Feature1 (search_label)
END-RULE

An example of *complex reasoning* is exemplified below as a (simplified version of the) "cut-out" situation discussed in [9] , which represents a particular configuration of features that should not be machined as only one operation. It considers the volumetrical "enter" interaction between the features (thus, it is a +V) as well their *labels* (thus, it is a +L).

IF
(Feature1 "enters" Feature2)
AND
(Feature1 AND Feature2 are active in the model)
AND
(Feature1 is a cylindrical *hole* or *pocket)*
AND
(Feature1 AND Feature2 opens to the same surface)
THEN
Assign relationship "cut-out-case" between the features
END-RULE

In essence, the priority scheme suggests that, after identifying the feature interaction case (at an appropriate level - initially volumetrical), some basic volumetric reasonings analyse the model searching for obvious mistakes of placement and *nature* (regardless the feature's *label).*

After correcting basic mistakes a set of reasonings would guarantee that all features are correctly labelled (helped by further feature-interaction analysis) because subsequent reasonings will perform more complex analyses (possibly application-dependent ones) that will subsume correct labelling and none basic volumetric misrepresentation.

5. CONCLUSIONS

A feature-based representation validation framework has been presented with emphasis on feature's expected behaviours called Designer's Intents. The reasoning mechanism has also been presented. Four sets of reasonings were identified and presented as well as their priority arrangements. These priorities were detected while implementing a prototype system called **FRIEND. FRIEND**, short for **F**eature-based **R**easoning system for **I**ntent-driven **EN**gineering **D**esign, is capable of performing the described *morphological functional* ADI and Volumetrical Designer's Intents validation.

Further to the validation structure found, the research is now looking to discover if any similar structure or general organisation can be established within rules of type (d), i.e., it is supposed that there is also an hierarchy for rules that validate the model from perspectives both from the common sense expected behaviour of features and application's constraints perspectives

To some extent, it could be said that the grouping and priority relationships discussed here, although realised after they were conceived and implemented, help to reinforce the notion that *morphological functional* ADI's and Volumetrical DI's represent a good means to establish and validate the semantics of a feature-based model.

6. REFERENCES

1. Hounsell, M. S. and K. Case. "Representation Validation in Feature-based Modelling: A Framework for Design Correctness Analysis and Assurance". 12th National Conference on Manufacturing Research (NCMR'96), September, Bath, UK, 1996, pp 156-161.
2. Zeid, I. *"CAD/CAM: Theory and Practice"*. Computer Science Series, Vol. 1. McGraw Hill International Editions. 1991
3. Dixon, J. R., E. C. Libardi JR, .and E. H. Nielsen. "Unresolved Research Issues in Development of Design-with-Features Systems". IFIP WG 5.2/ NSF Working Conf. on Geometric Modelling, ResselaerVille - USA, Elsevier Sc. Pub., Vol. 1, 1990, pp 183-196.
4. Yoshikawa, H. and K. Ando. "Intelligent CAD in Manufacturing". Annals of the CIRP, Vol. 36/1, 1987, pp 77-80.
5. Hounsell, M. S. & K. Case. "Morphological and Volumetrical Feature-based Designer's Intents". To appear in the 13th National Conference on Manufacturing Research (NCMR'97), September, Glasgow, England, 1997.
6. Dohmen, M. "Constraint Techniques in Interactive Feature Modeling". TUDelft - Delft University of Technology, Faculty of Technical Mathematics and Informatics, 1994.
7. Lenau, T. and L. Mu. "Features In Integrated Modelling of Products and Their Production". International Journal of Computer Integrated Manufacturing, Vol. 6(1-2), 1993, pp 65-73.
8. Shah, J. J. "Philosophical Development of Form Feature Concept". CAM-I Inc. Arlington, Texas, USA, 1990.
9. Mill, F. G., J. C. Salmon, and A. G. Pedley. "Representation Problems in Feature-based Approaches to Design and Process Planning". International Journal of Computer Integrated Manufacturing, Vol. 6(1-2), 1993, pp 27-33.
10. Hounsell, M. S. & K. Case. "Geometric Feature Interaction Identification". To appear in the 32nd International MATADOR Conference, July, Manchester, UK, 1997
11. Silva, R. E. d., K. L. Wood, and J. J. Bearman. "Representing and Manipu.lating Interacting Interfeature Relationships in Engineering Design for Manufacture". (ASME) Advances in Design Automation, Vol. DE-32-1, 1990, pp 1-8.