Implementation aspects of a feature-based design system for manufacturing planning

This item was submitted to Loughborough University’s Institutional Repository by the/an author.


Additional Information:

- This is a conference paper.

Metadata Record: https://dspace.lboro.ac.uk/2134/13417

Version: Accepted for publication

Publisher: International Federation for Information Processing (IFIP)

Please cite the published version.
This item was submitted to Loughborough’s Institutional Repository (https://dspace.lboro.ac.uk/) by the author and is made available under the following Creative Commons Licence conditions.

For the full text of this licence, please go to:
http://creativecommons.org/licenses/by-nc-nd/2.5/
Implementation Aspects of a Feature-Based Design System for Manufacturing Planning

James Gao
School of Industrial and Manufacturing Science
Cranfield University
Cranfield
Bedford, MK43 OAL
UK

Keith Case
Department of Manufacturing Engineering
Loughborough University of Technology
Loughborough
Leicestershire, LE11 3TU
UK

ABSTRACT. A prototype feature-based design system for process planning is described. The system contains a design by features user interface which is developed within a Boundary Representation modeller; a feature modeller for manipulation of feature data; and an information mapper (feature processor) to transform design information into a form suitable for the planning system. In principle, the design system can be tailored for applications other than process planning by the use of alternative information mapping mechanisms. Emphasis is given to the software implementation aspects of the feature-based design system, such as the design and implementation of the user interface, the arrangement of the additional feature modelling functions together with standard solid modelling functions, the communication mechanism between different data models, and the tools used for the implementation.

KEY WORDS: Feature-based Modelling, Manufacturing Planning.
1. INTRODUCTION

Contemporary Computer-Aided Design (CAD) systems are mostly developed based on one or more of the geometric modelling technologies such as wireframe, surface modelling and solid modelling. A wide spectrum of engineering components can be modelled and specific numerical information about the components can be obtained from the geometric model for subsequent applications such as analysis and manufacturing code generation. However, manufacturing planning systems such as process planning, inspection planning and assembly planning require more information about products than pure geometric data. Such information includes:

- A feature-based product representation. A part consists of a number of components, and each component can be expressed as a number of features such as holes, pockets, slots, steps and surfaces so that manufacturing processes or inspection operations can be associated with every feature.
- Technological specifications and requirements such as material types, heat treatment methods, surface finish, and geometrical and dimensional tolerances between features.
- Feature relationships or connectivities such as parent-child relationships, compound features, external access directions for each feature.

The integration of a CAD system and a manufacturing planning system therefore requires that the above information is either explicitly represented in the design system's database or can be processed into the required form through a data mapping mechanism.

The difficulties facing the research and development community are both in the definition of a generally acceptable feature-based product model and in the implementation of such a data model in practical CAD/CAM systems. Many feature representation schemes have already been developed in the past, among which the most significant ones may be the hierarchical form feature model developed by Computer-Aided Manufacturing - International (CAM-I) [BUT 85] and the S'IEP form feature model which has been proposed as a draft standard by ISO [ISO 92]. However, there is still a long way to go for the draft to become a generally accepted international standard.

In parallel to the development of feature representations, form features have been implemented in both prototype and commercial CAD systems. The most successful ones may be the latest parametric solid modelling systems such as Pro--Engineer [JAM 91] and CADDS5 [COM 91]. Normally, a parametric CAD system maintains a set of parameters describing features and a separate Boundary Representation (Brep) data model for the component. However, the parameters do not contain sufficient information about features especially lower level features such as faces, edges, axes, median planes which are essential for tolerancing, whilst the Brep model is similar to all other traditional Brep representations in which form features are not represented at all.

The objective of the research work described here was to propose and implement a feature representation for manufacturing planning. The feature representation
methodology has been described elsewhere [GAO 92, GAO 93], and this paper emphasises the software implementation of the feature-based design system. The system contains (i) a design by features user interface which has been developed on top of a commercial Boundary Representation (Brep) modeller, (ii) a feature modeller which manipulates the feature parameters in correspondence with the Brep model, (iii) an information mapping mechanism (a feature processor) which processes the design information into the form that is required by an integrated knowledge-based process planning system. The following sections further describe the functional modules of the prototype system.

2. The Main Functional Modules of LUT-FBDS

The main functional modules of the prototype system LUT-FBDS (Loughborough University of Technology Feature-Based Design System) include a user interface written in Horses (a proprietary user interface management system), standard solid modelling functions from the Imaginer Brep solid modeller, a feature modeller and a feature mapping mechanism (a feature processor). Figure 1 illustrates the relationships between the functional modules.

![Diagram](https://via.placeholder.com/150)

**Figure 1: The Functional Modules of the Feature Based Design System**

The user interface is an extension of the existing Imaginer interface to include feature modelling facilities. Imaginer functions such as create, draw, lift, colour and rotate are
called through the interface to generate feature geometry (a Boundary Representation) and to graphically manipulate feature entities. The feature modeller creates a parametric feature data model for every feature which can be retrieved and edited through the feature modeller and the user interface. The feature modeller is written in C and is linked with Imaginer so that it becomes part of Imaginer's functions. The feature processor is also written in C, but runs independently to read the parametric feature data and reason about the information and to generate a more complete data model with additional information for process planning.

3. The Design by Features User Interface

The user interface is written as Horses Network files. The functional modules can be described as shown in figure 2. The top level in figure 2 has six functional models, i.e.

**Component**, which allows the user to define general component specifications, overall size and two types of stock materials (prismatic and rotational shown as Block1 and Block2 respectively at level 2 in the figure).

**Feature**, which allows the user to define features of class boss, hole, pocket, step, notch, slot, non-through slot, and surface (as shown at level 2). Every feature class has a number of profile shapes, which are shown as Prof.a, Prof.b, Prof.c, Prof.d and Prof.e at the bottom level in the figure. Once the profile type is selected, the topology of the feature is determined. Features of the same class and the same profile are instances of the same feature primitive. The total feature primitives available are listed in figure 3. It should be noted that the number of feature primitives may be increased for each feature class by defining more profile shapes.

**Compound feature**: which allows the user to define compound features e.g. Cross-slots, T-slots, pattern holes, counter-bores as shown at the second level of figure 2. Compound features can be defined as primitives as seen in figure 3, and as relationships between existing features.

**Operation**: which allows the user to perform feature operations such as drag, rotate, delete, drag and copy, rotate and copy as shown at the second level of figure 2. A feature validation mechanism has also been developed which detects changes in feature class and dimension during feature operation. This issue has been covered in detail elsewhere [CAS 93].

**Relationship**: which allows the user to define relationships such as dimensions and tolerances (D&T), parent-child relationships (P&C) and compound features as shown at the second level of figure 2.

**Query**: which allows the user to make queries of the Brep model concerning mass and face properties and to output this into a feature data model. Examples are volume of component, normal vectors of faces and names of each geometric entity.
4. The Imaginer Standard Modelling Functions

As with all other commercial solid modellers, Imaginer [PAF 91a] has standard functions for visual display, creating and drawing solids, Boolean operations, solid editing (drag, rotate, copy, delete), colour shading and screen layout, etc. These functions are organized into a number of packages which can be called through the design by features user interface written in Horses network files [PAF 91b] to generate geometric models for features and components. For example, to create a square pocket on a block, the Horses network file would include the following Imaginer functions:

- Create a 2 dimensional square profile of the given size of the pocket.
- Lift or sweep the profile to create a solid of the given size of the pocket.
- Delete the 2D profile, since it is no longer useful.
- Perform a Boolean operation (subtract) between the solid and the block which already exists. The resultant solid is a block with a pocket Note that the solid for the pocket is still kept, since it carries volumetric information about the pocket.
- Name both the resultant solid (called the component) and the solid used to created the pocket (called the pocket). Note that when the pocket and the component are named, their faces are also given distinctive names so that features and their faces are associated with information in the feature data model.
- Draw the component on the screen (usually the solid for the pocket is only drawn if requested).

The above functions would create a Brep model for the feature and the component. At the same time, the parametric feature information is stored in a parametric feature data model through the feature modelling functions which are also called from the user interface file.

<table>
<thead>
<tr>
<th>Class</th>
<th>Specific profile shapes available through interface</th>
<th>user defined</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boss</td>
<td><img src="image" alt="Boss Shapes" /></td>
<td><img src="image" alt="User Defined" /></td>
</tr>
<tr>
<td>Pocket</td>
<td><img src="image" alt="Pocket Shapes" /></td>
<td><img src="image" alt="User Defined" /></td>
</tr>
<tr>
<td>Hole</td>
<td><img src="image" alt="Hole Shapes" /></td>
<td><img src="image" alt="User Defined" /></td>
</tr>
<tr>
<td>Through Slot</td>
<td><img src="image" alt="Through Slot Shapes" /></td>
<td><img src="image" alt="User Defined" /></td>
</tr>
<tr>
<td>Non-through Slot</td>
<td><img src="image" alt="Non-through Slot Shapes" /></td>
<td><img src="image" alt="User Defined" /></td>
</tr>
<tr>
<td>Notch</td>
<td><img src="image" alt="Notch Shapes" /></td>
<td><img src="image" alt="User Defined" /></td>
</tr>
<tr>
<td>Step</td>
<td><img src="image" alt="Step Shapes" /></td>
<td><img src="image" alt="User Defined" /></td>
</tr>
<tr>
<td>Surface</td>
<td><img src="image" alt="Surface Shapes" /></td>
<td><img src="image" alt="User Defined" /></td>
</tr>
</tbody>
</table>

![General Profile Shapes](image)

**Figure 3: The feature/compound feature primitives in the library**
5. The Feature Modeller

The feature modeller creates a parametric feature data model for components. It also retrieves and edits the feature data model during interactive design through the user interface. The feature modeller was developed in this project and is written in C. It is organised as three packages (as seen in figure 4) namely Component, Features and Operations. The packages are computationally 'linked' with the Imaginer executable and therefore can be treated as additional functions to the standard Imaginer modeller.

The **component package** includes routines to output and retrieve component level information such as specifications (Specs.), parent-child relationships (P&C), dimensions and tolerances (D&T) and compound features (Com.ft).

The **features package** includes routines to output and retrieve feature level information such as feature parameters (ft.data) - name, class, profile type, size, surface finish; Local Coordinate System (LCS) -origin, x, y, z vectors; and face data (Face) - normal vector, type, parent face, etc.

The **operations package** includes routines to output and retrieve feature operation information such as move, delete, copy and feature validation.

The feature modelling functions are called in exactly the same way as standard Imaginer functions in the Horses network file as mentioned in section 4.
6. The Feature Processor

The feature processor is an information mapping mechanism developed in C. It reads the data from the parametric feature data model and processes it into a more complete data model suitable for subsequent process planning. The processor is not computational 'hard linked' with the Imaginer executable, therefore it can be executed separately. Examples of the functions of the feature processor include:

- Creation of a better organized data structure for the feature-based component data model.
- Derivation of face normals for every feature and the component. Normals to imaginary faces are the external access directions to the feature. Normals to the faces of the component may be used to determine set-ups of the component on a cutting or inspection machine.
- Derivation of parent-child relationships between features and their faces. For example, every imaginary face has a parent face which belongs to the stock material or another feature.
- Output of the data into different formats, e.g. a human readable text file or a file which can be read by the knowledge-based system used for the integrated process planning system [HUA 93].

![PAFEC IMAGINER USER INTERFACE](image)

**Figure 5: The Procedure for Defining the Stock Material**
7. Using the Prototype System LUT-FBDS

A simple component is modelled to demonstrate the use of the prototype feature-based design system. The prototype system is implemented as a sub-system of Imaginer and access is provided through the 'application' icon on the standard Imaginer interface as shown in figure 5.

The steps are as follows:
(a) Defining stock material. The stock material must always be defined first. The information required is the component specifications and the size of the initial stock material (a rectangular block in this example). The origin of the component coordinate system is automatically set at the middle point of the bottom face of the stock material. The stock could be a rectangular block, cylindrical or an existing component of arbitrary complexity created by Imaginer or another CAD system. Figure 5 illustrates the procedure for defining a stock material of size (10, 20, 10).

(b) Creating features on the block. When a block is generated, features can be added to it. This is done by selecting the feature's class (e.g., boss, pocket, hole, through slot, non-through slot, notch, step or surface) and profile type as shown in figure 6. When the parameters of the feature are input, its orientation can be determined by either selecting the default orientation (same as the component coordinate system) or by rotating the feature about the x, y and z axes for a given degree. Similarly, a compound feature can be created by evaluating its parameters. Figure 6 illustrates the procedure for creating a pocket of size (4, 1.5, 2). A compound feature (four holes at the corners of a square pattern) created in the similar way is also shown.

(c) Defining tolerance information. Dimensions and tolerances can be defined as shown in figure 7. For example, the second hole of the pattern (with name 'patt_holes.h2') has a positional tolerance with respect to the first hole in the pattern (named 'patt_holes.h1'). The axes of the two holes are used as the entities for tolerancing.
When the features are created and the tolerance is defined through the user interface, Brep models are generated for the features and the component. At the same time a parametric data model is generated by the feature modeller. The content of the parametric data model (simplified) for the above component is shown in figure 8.

8. Conclusions and Future Work

A feature-based product representation is not only essential for manufacturing planning activities, but also important for product design purposes. The difficulties facing the research and development community lie both with the definition of a feature information model and with the implementation of such a model in practical CAD/CAM systems.

The reported research has been targeted on the above problems. As a result, a prototype feature-based design system has been developed. The system employs in its library a set of feature primitives from which components can be designed. The system makes full use of solid modelling technologies and the parametric concept. The system also adopts a combination of design by features approach and feature recognition techniques such as geometric reasoning. A feature mapping mechanism (a feature processor) has been developed to process the generic information into application specific data such as that for process planning. Similarly, the design system can be tailored for other applications through different information mapping mechanisms.

During the implementation of the system, some problems and difficulties have been identified. For example, each feature primitive needs a set of specific codes to perform required functions and the codes will increase as the number of primitives or the number of functions increase or the complexity of the feature geometry increases. Each application needs a different data mapping mechanism which will become more complicated as the requirements of the application system increase. As product definition has not been standardised, the information processing functions cannot be generalised. During interactive design and re-design, individual features may intersect each other, and this may result in the changes in feature class, dimensions and relationships. The level of integration between the parametric feature data model and the Brep model is still relatively low. Each feature is represented in the Brep model, the parametric data model and also the processed data model and thus generates redundant data. Future research and development work should be targeted on the above problems in addition to merely extending the application areas for features.

9. Acknowledgements

This research was funded by the Application of Computers in Manufacturing Engineering (ACME) Directorate of the Science and Engineering Research Council (SERC) under Grant Number GR/035657, 'Process Capability Modelling for Design and Selection of Processing Equipment'.
Component: Exam1

Material: HS
BLK Roughness: 50.0
BLK Hardness: 120.0
BLK x_dim: 10.0
BLK y_dim: 20.0
BLK z_dim: 10.0

FT Name: patt_holes_h1
CLASS: pocket
SUB-TYPE: pchrc
DIM1: 1.0
DIM2: 0.0
DEPTH: 2.5
RADIUS: 0.8
ANGLE: 0.0
SURF FINISH: 40.0
FACE1: patt_holes_h1.f1.im
FACE2: patt_holes_h1.f1.re
FACE3: patt_holes_h1.f3.re
FACE4: patt_holes_h1.f4.re
FACE5: na
FACE6: na
LCS: x = -3.5, y = -8.5, z = 10.0
X-AXIS: 1.0, 0.0, 0.0
Y-AXIS: 0.0, 1.0, 0.0
Z-AXIS: 0.0, 0.0, 1.0

FT Name: patt_holes_h3
CLASS: pocket
SUB-TYPE: pchrc
DIM1: 0.0
DIM2: 0.0
DEPTH: 2.5
RADIUS: 0.8
ANGLE: 0.0
SURF FINISH: 40.0
FACE1: patt_holes_h3.f1.im
FACE2: patt_holes_h3.f1.re
FACE3: patt_holes_h3.f3.re
FACE4: patt_holes_h3.f4.re
FACE5: na
FACE6: na
LCS: x = -3.5, y = -8.5, z = 10.0
X-AXIS: 1.0, 0.0, 0.0
Y-AXIS: 0.0, 1.0, 0.0
Z-AXIS: 0.0, 0.0, 1.0

FT Name: patt_holes_h4
CLASS: pocket
SUB-TYPE: pchrc
DIM1: 1.0
DIM2: 0.0
DEPTH: 2.5
RADIUS: 0.8
ANGLE: 0.0
SURF FINISH: 40.0
FACE1: patt_holes_h4.f1.im
FACE2: patt_holes_h4.f1.re
FACE3: patt_holes_h4.f3.re
FACE4: patt_holes_h4.f4.re
FACE5: na
FACE6: na
LCS: x = 3.5, y = -8.5, z = 10.0
X-AXIS: 1.0, 0.0, 0.0
Y-AXIS: 0.0, 1.0, 0.0
Z-AXIS: 0.0, 0.0, 1.0

Figure 8: The parametric information for example components (simplified)
10. References


[JAM 91] James, M., Prototyping Cad, CADCAM, (July 1991)


11. Biography

Dr James Gao is currently a teaching fellow at Cranfield University. He has over eight years research experience in CAD/CAM, product modelling, feature-based design and process planning. His recent research interests include concurrent engineering, design for manufacture and the application of Information Technology in manufacturing
Dr Keith Case is a Senior Lecturer in the Department of Manufacturing Engineering at Loughborough University of Technology. He has over twenty years of research experience in the fields of Computer Aided Ergonomics Design, Computer Aided Robotic Workplace Design and Feature-based Design. Recent interests extend to the applications of Virtual Reality techniques in conceptual design and human workplace design.