Int J Adv Manuf Technol (1999) 15:38–48

The International Journal of
Advanced
Manufacturing
Technology

Generating Alternative Interpretations of Machining Features

J. Y. Lee* and K. Kim

Department of Industrial Engineering, Pohang University of Science and Technology, Pohang, South Korea

One of the major difficulties in extracting machining features is the lack of a systematic methodology to generate alternative ways of manufacturing a machined part. Most of the early research in feature extraction and process planning has not considered this aspect, and has focused on the generation of a single interpretation. In this paper, we propose a feature-based approach to generating alternative interpretations of machining features from a feature-based design model. The proposed approach simplifies the generation of alternative machining feature models by using information on feature which is captured and maintained during feature-based modelling and machining feature extraction. A set of machining features is incrementally extracted during the feature-based design process of a machined part. A feature conversion process converts each design feature into a machining feature or a set of machining features by using information on the geometry and the feature. Using reorientation, reduction, and/or splitting operations, alternative models are generated from the sets of extracted machining features. During the execution of each operation, unpromising models are pruned by using criteria such as minimising the number of accessibility directions. The machining features and their precedence relationships are represented in a STEP-based machining feature graph for the purpose of data exchange.

Keywords: Alternative interpretation; Feature-based design; Feature extraction; STEP

1. Introduction

Recently, the concept of features for design and manufacturing applications has received much attention [1]. Features can be defined from different viewpoints. Design features are the shapes related to a part's function, its design intent, or the model construction methodology, whereas machining features are the shapes associated with distinctive machining operations. For machining applications, a design model needs to be interpreted in terms of machining features. This process is called *feature recognition* or *feature extraction*. Depending on whether the input design model contains feature information or not, there are two major approaches to feature extraction:

1. The geometry-based approach.
2. The feature-based approach.

In the geometry-based approach, machining features are recognised directly from a geometric design model [2–13]. On the other hand, in the feature-based approach, they are converted from a feature-based design model [14–19].

A set of extracted machining features is often called a machining feature model or interpretation of a part. Usually, a part can be represented by more than one interpretation. These alternative interpretations correspond to different manufacturing ways to machine the part [11–13,20,21]. Sakurai and Chin [11] and Tseng and Joshi [12] proposed a cell-based decomposition approach to generating alternative models. The volume to be removed (delta volume) is decomposed into cells by extending and intersecting all of its surfaces or halfspaces. A subset of these cells is then combined into a machining feature. In this way, cell composition is repeated until all the cells of the delta volume are consumed. As a result of cell composition, the delta volume is completely decomposed into a set of machining features, which is taken as an interpretation. Alternative interpretations can be generated by changing the composition sequence of the cells. Gupta [20] and Gupta and Nau [21] viewed an interpretation as a feature cover of the delta volume. They computed alternative interpretations from an initial feature model by using the feature covering methodology. Han [13] proposed a procedure to compute a satisfactory interpretation and to generate alternative interpretations on request from a process planner. However, loss of design information and computational inefficiency have been major problems in generating alternative feature models.

In this paper, we propose a feature-based approach to generating alternative interpretations of machining features from a feature-based design model. The proposed approach simplifies the generation of alternative machining feature models by using information on a feature which is captured and maintained

---

Correspondence and offprint requests to: Dr Kwangsoo Kim, Department of Industrial Engineering, Pohang University of Science and Technology, IE2195, San 31, Hyoja-dong, Pohang 790–784, South Korea. E-mail: kskim@postech.ac.kr

*Current address: Computer Software Technology Lab, ETRI, Taejon 305 350, South Korea.

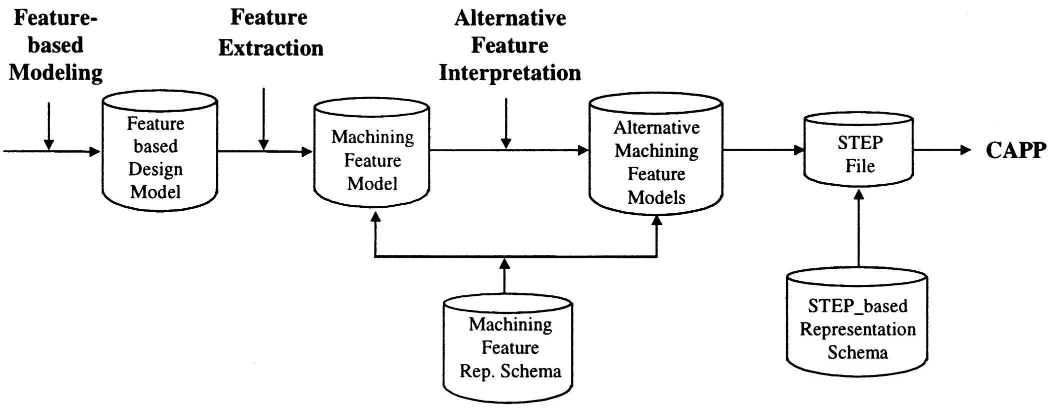

Fig. 1. A schematic diagram for generating alternative feature models.

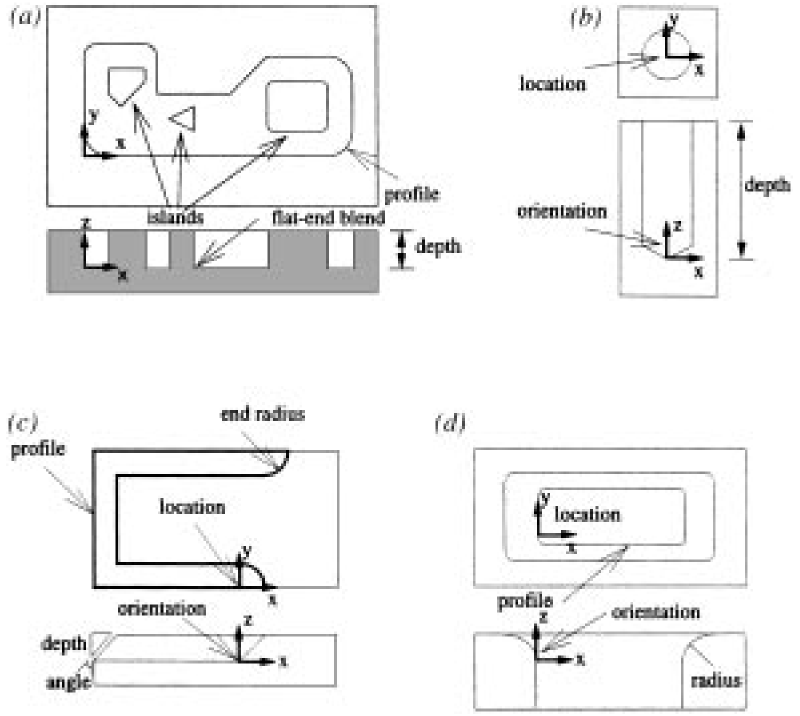

Fig. 2. Machining features: (*a*) pocket, (*b*) hole, (*c*) chamfer, and (*d*) round.

during feature-based modelling and machining feature extraction. A schematic diagram for generating alternative feature models is shown in Fig. 1. A set of machining features is incrementally extracted during the feature-based design process of a machined part. The feature conversion process converts each design feature into a machining feature or a set of machining features by using information on the geometry and feature. Using reorientation, reduction, and/or splitting operations, alternative models are generated from the sets of extracted machining features. During the execution of each operation, unpromising models are pruned by using criteria such as minimising the number of accessibility directions. The machining features and their precedence relationships are represented in a STEP-based machining feature graph for the purpose of data exchange.

The remainder of the paper is organised as follows. Section 2 describes a feature representation scheme for machining features. Section 3 describes a method for generating alternative machining feature models. Section 4 illustrates implementation results. Section 5 presents a conclusion with some remarks.

2. Machining Feature Representation

2.1 Machining Features

Machining features considered in this paper are restricted to 3-axis milling operation features, similar to the MRSEVs [22,23]. The domain of machining features is confined to the subclasses of the linear swept features and edge-cut features

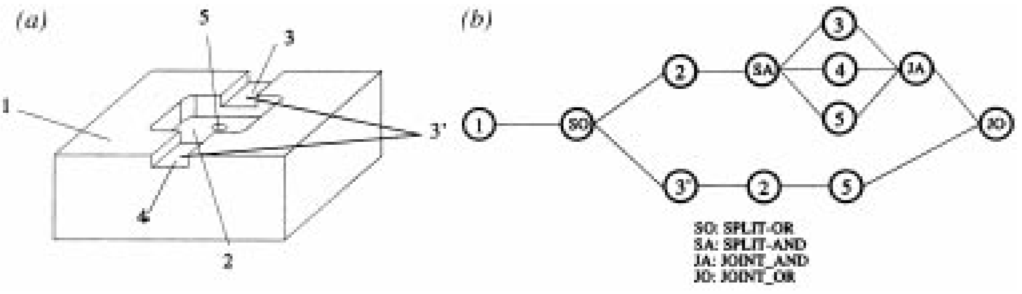

Fig. 3. Feature representation: (*a*) a designed part and (*b*) a machining feature graph.

```
SCHEMA MACHINING_FEATURE_REPRESENTATION_SCHEMA
REFERENCE FROM MACHINING_FEATURE_SCHEMA;
REFERENCE FROM MRSEVS_SCHEMA;

TYPE node = ENUMERATION OF
(machining_feature_representation_element,
 machining_feature_representation_set);

ENTITY machining_feature_activity
            id : INTEGER;
            feature : machining_feature;
END_ENTITY;

ENTITY machining_feature_representation_element
            activity_element : machining_feature_activity;
END_ENTITY;

ENTITY machining_feature_representation_set
            SUPERTYPE OF (ONEOF(sequential_activity_set,
            multiple_choice_activity_set, serial_unordered_activity_set));
            activities : LIST[1;?] OF node;
END_ENTITY;

ENTITY sequential_activity_set
            SUBTYPE OF machining_feature_representation_set;
END_ENTITY;

ENTITY multiple_choice_activity_set
            SUBTYPE OF machining_feature_representation_set;
END_ENTITY;

ENTITY serial_unordered_activity_set
    SUBTYPE OF machining_feature_representation_set;
END_ENTITY;

ENTITY machining_feature_representation
            plan : machining_feature_representation_set;
END_ENTITY;
END_SCHEMA;
```

Fig. 4. A STEP-based representation schema of machining features.

such as pocket, hole, chamfer, and round as shown in Fig. 2. In order to machine a part P from the stock S, a set of machining features must be extracted to remove the delta volume Δ which is the regularised difference between the initial stock and the part. Given a part P and a raw stock S, a set of machining features is said to be a *valid machining feature model* $M = \{M_1, M_2, \ldots, M_n\}$ if it satisfies the following properties where $-^*$, $\cup^*$, and $\cap^*$ denote regularised set difference, union, and intersection [7]:

Completeness: P can be fully decomposed when the union of all volumetric features M_i contains the delta volume Δ, or

$$\Delta \subseteq \bigcup_{M_i \in M} M_i$$

where $\Delta = S -^* P$.

Non-intrusion: $M_i \cap^* P = \emptyset$ for each M_i.

Presence: For each M_j at least one face of M_j should contact with P, or

$$\bigcup_{M_i \in M-\{M_j\}} M_i -^* \Delta \neq \emptyset \text{ and } M_j \cap P \neq \emptyset.$$

Accessibility: To remove each machining feature, a tool should be moved from the outside of the stock S into the removal volume without intersecting the part P.

2.2 Machining Feature Graph

An AND/OR graph, as shown in Fig. 3, is used to represent alternative machining feature models [24,25]. An arc in the graph represents the precedence relationship between two nodes. A node represents one of the following five different types: SPLIT-AND, SPLIT-OR, JOIN-AND, JOIN-OR, and M-FEATURE. A SPLIT-AND type node provides the basis for representing sequence alternatives in machining a part. This implies that all the paths following a SPLIT-AND type node must be executed in any sequence. A JOIN-AND type node is required to bring multiple paths back together after a SPLIT-AND type node. A SPLIT-OR type node provides the basis for representing feature alternatives in machining a particular part, which implies that only one of the machining features following a SPLIT-OR type node must be selected to be machined. A JOIN-OR type node is required to bring multiple paths back together after a SPLIT-OR type node. A machining feature is represented in an M-FEATURE type node.

2.3 Physical Representation Using STEP

To transfer the information contents extracted from CAD data to process planning, a formal scheme for representing machining features and their relationships needs to be defined. Thus, a STEP-based representation schema of machining features has been developed using the EXPRESS language as shown in Fig. 4. In the representation schema, ENTITY multiple_choice_activity_set stores machining features located between SPLIT-OR and JOIN_OR nodes, and ENTITY serial_unordered_activity_set stores machining features between SPLIT-AND and JOIN-AND nodes. The main intent of the STEP representation schema is to maintain core components of the model as generic as possible so that any process can use the same components of the model. To realise the generic property of features, the above representation is defined recursively. The machining feature representation has an attribute

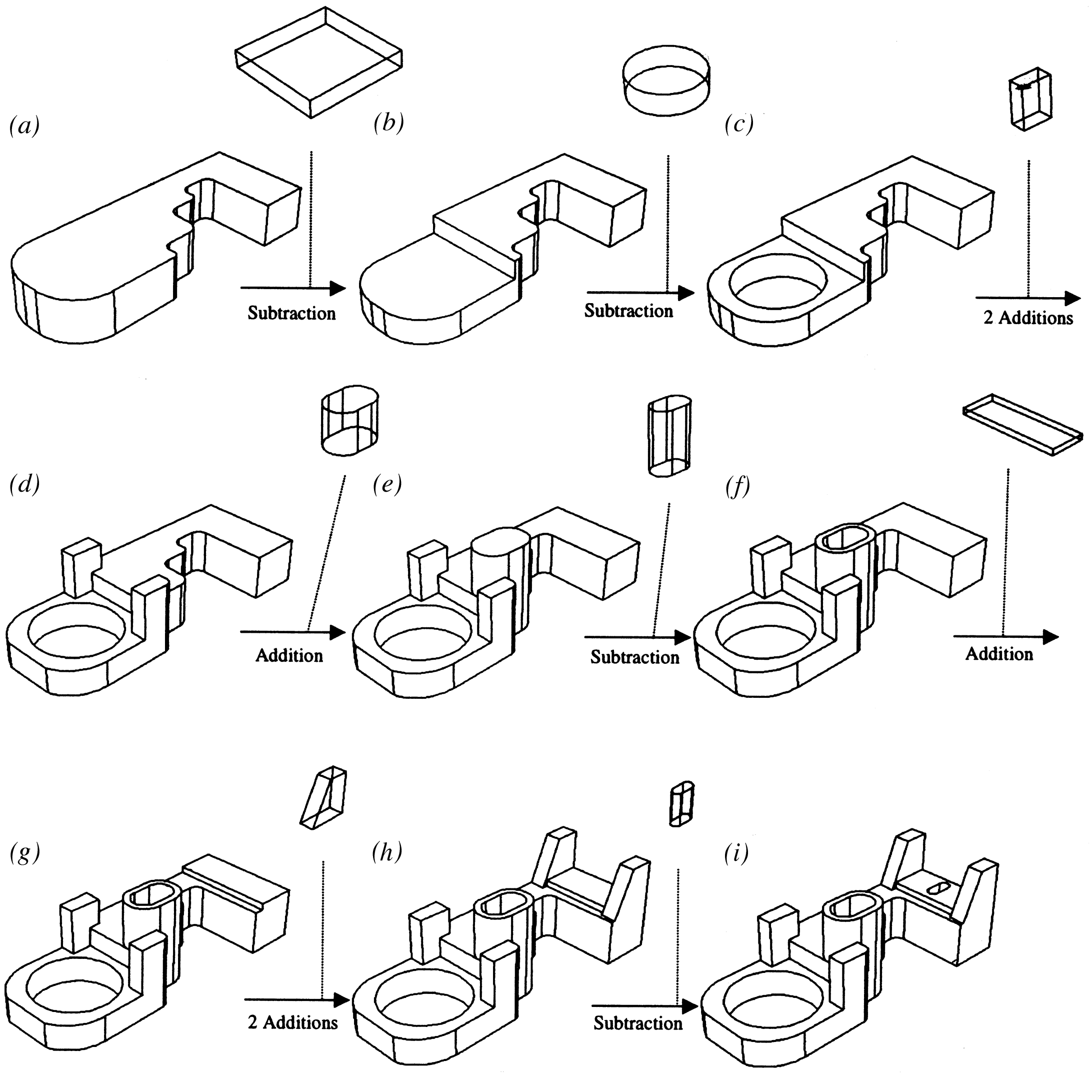

Fig. 5. Incremental feature-based modelling process.

called *activities* that references another *plan* set. Each element of the plan set references another process plan set, forming a recursive structure.

3. Generation of Alternative Machining Feature Models

3.1 An Initial Machining Feature Model

A set of machining features is extracted during modelling a part incrementally using design features [26]. Figure 5 shows a feature-based modelling process. Figure 6 shows a set of the machining features extracted to machine the part shown in Fig. 5(*i*). A machining feature model $M = \{M_1, M_2, \ldots M_n\}$ as well as feature precedence E can be represented in a *simple machining feature graph*, *S-MFG* = <*M*,*E*>, that has the following properties:

1. There is no duplicated machining feature in *M*.
2. It has no pair of SPLIT-OR and JOIN-OR nodes.

A simple machining feature graph of the extracted machining features is constructed as follows:

1. Classify all the machining features that have the same approach direction into clusters C_i.
2. For each cluster C_i, create a pair of SPLIT-AND and JOIN-AND nodes and insert all the features in C_i into that pair.
3. Create a new pair of SPLIT-AND and JOIN-AND nodes and insert all the created pairs of C_i into the new pair.

3.2 Generating Alternative Feature Models

Alternative machining feature models are generated by applying reorientation, reduction and/or splitting operations to an extracted machining feature model *M*, as shown in Fig. 7.

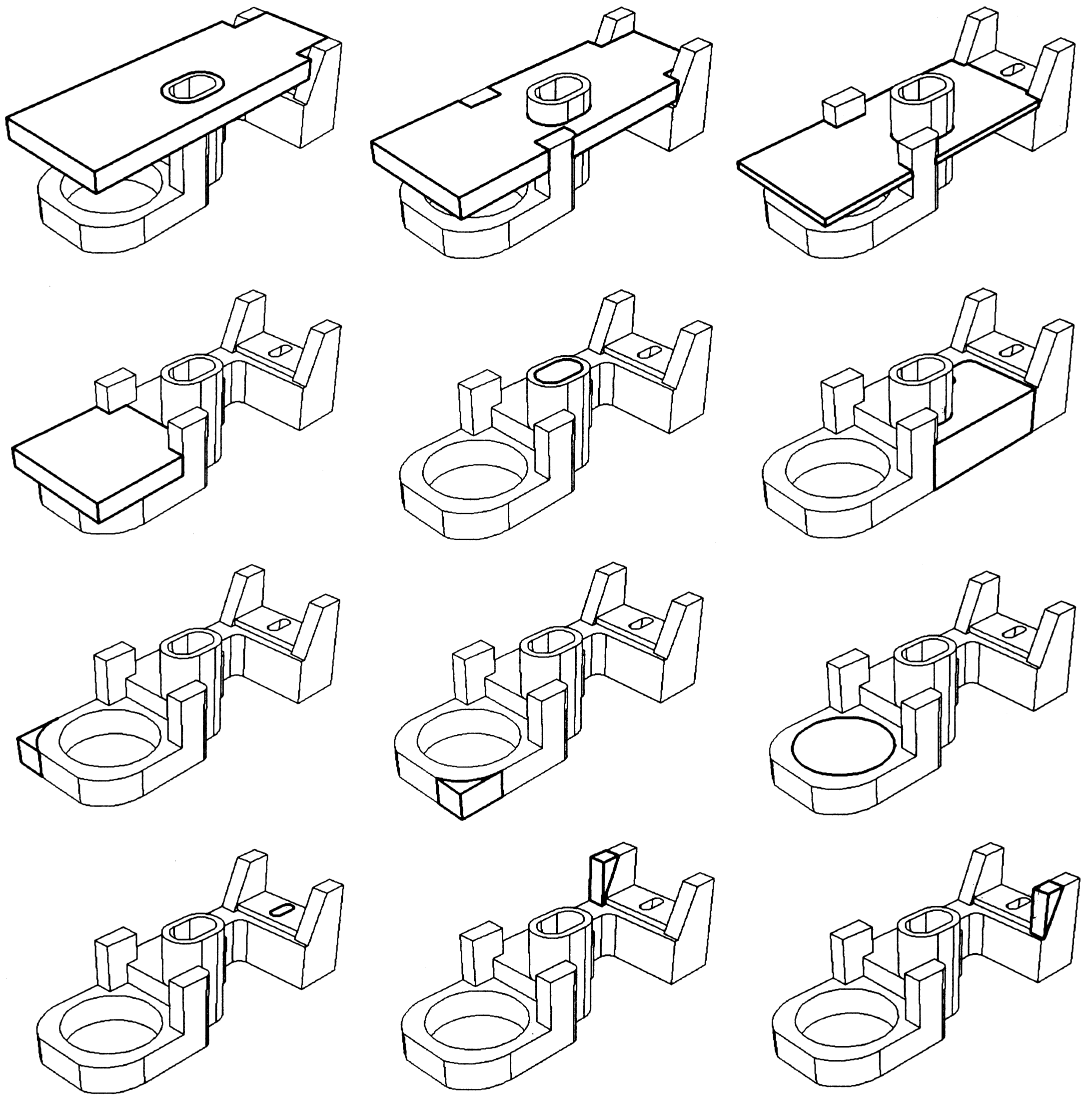

Fig. 6. Extracted machining features to machine the example part shown in Fig. 5.

3.2.1 Reorientation

For each feature M_i in the machining feature model M, the reorientation operation is performed to find a new feature M_i' that has the same feature type as M_i in a different approach direction. A machining feature can be machined along several feasible approach directions, which can be determined by extending the feature by an infinitesimal amount along a set of directions. If the extended volume does not intersect the part P, then it is accessible along that direction. For example, a through hole can be accessible in both axis directions. Since it is assumed that a machining feature is associated with only one approach direction, a feature with different approach directions is converted to several different features for each direction.

If such a feature M_i' exists as shown in Fig. 7(*c*), a new machining feature model $M' = M - \{M_i\} \cup \{M_i'\}$ is generated. Then, the manufacturability of the new model is analysed. The manufacturability depends on many factors, but one of the major factors is the number of set-ups. Reducing the number of set-ups will not only reduce the time needed for machining, but also result in better machining tolerances. In this paper, only the number of set-ups is considered in the manufacturability analysis. If the number of set-ups in M' is larger than that in M, M' is discarded as an unpromising feature model as shown in Fig. 7(*d*). Otherwise, it is saved as M^i (a new alternative model of M). The operation continues until all the features in M are evaluated. The following procedure describes the reorientation operation in detail.

PROCEDURE reorientation (M_i a set of M_i')

INPUT: M_i

OUTPUT: a set of M_i

1. Find a set of all M_i' such that
   - (*a*) the approach direction of M_i' is different from that of M_i.
   - (*b*) M_i' must be accessible such that $ext(M_i') \cap^* P = \emptyset$ where $ext(M_i')$ is the extended volume beyond M_i' along the approach direction of M_i'.

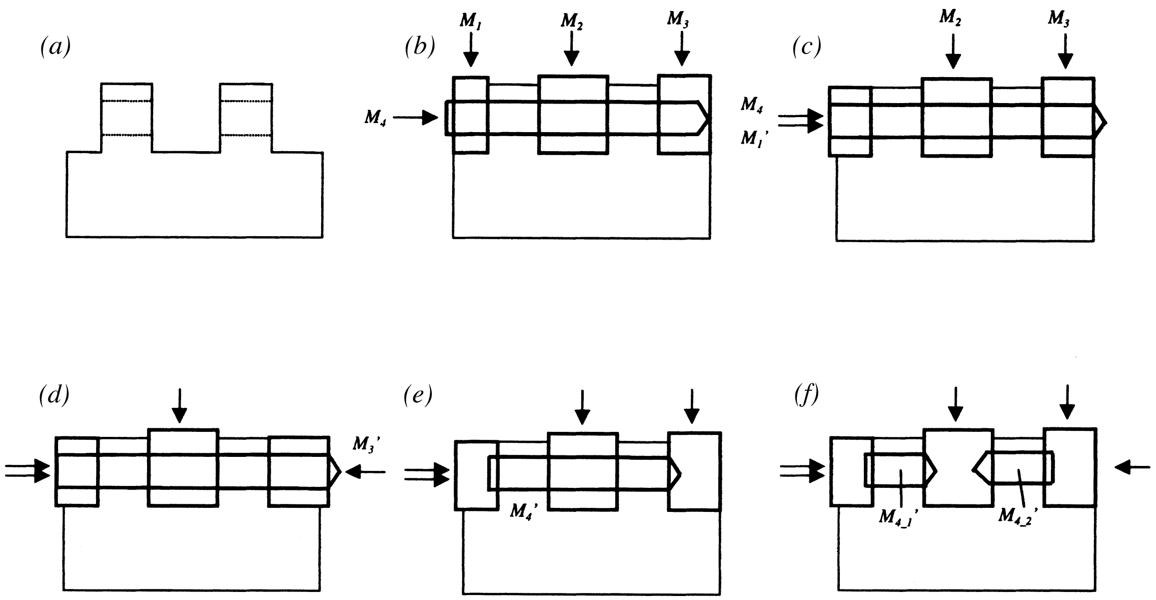

Fig. 7. Alternative generation operations: (*a*) a designed part, (*b*) an initial machining feature model, (*c*) M_1' is generated from M_1 reorientation, (*d*) an unpromising reorientation operation, (*e*) M_4' is generated from M_4 reduction by M_1', and (*f*) M_{4_1}' and M_{4_2}' are generated from M_4' splitting by M_2.

```
2. If no such $M_i'$ exists, exit with a NULL set of $M_i'$.
3. Return a set of $M_i'$.
END_PROCEDURE
```

3.2.2 Reduction

When the reorientation operation ends, the reduction operation is applied to each machining feature model M^k in the set of alternative feature models, $A = \{M^1, M^2, \ldots, M^n\}$. After finding each feature M_j in M^k, intersecting with the feature M_i, the reduction operation tries to find a feature M_i' by $M_i' = M_i {}^{-*} M_j$. If there exists such a feature, a new feature model $M^{k'} = M^k - M_i\} \cup [M_i'\}$ is generated as shown in Fig. 7(*e*). Details are described in the following procedure.

```
PROCEDURE reduction ($M_i$, $M_j$, $M_i'$)
INPUT: $M_i$ and $M_j$
OUTPUT: $M_i'$
1. If $M_i \cap^* M_j = \emptyset$ exit.
2. Find $M_i'$ such that
   (a) $erv(M_i') \subset erv(M_i)$ where $erv()$ is an effective removal volume.
   (b) $erv(M_i') \cup^* P = \emptyset$
3. If no such $M_i'$ exists, exit.
4. Return $M_i'$.
END—PROCEDURE
```

3.2.3 Splitting

Finally, the splitting operation is applied to the set of alternative models A. After finding each intersecting feature M_j in M^k, the splitting operation is performed to split M_i into two features M_{i1}' and M_{i2}' where $\{M_{i1}', M_{i2}'\} = M_i -^* M_j$. If such features M_{i1}' and M_{i2}' exist, a new feature model $M^{k'} = M^k - \{M_i\} \cup \{M_{i1}', M_{i2}'\}$ is generated as shown in Fig. 7(*f*).

```
PROCEDURE split ($M_i$, $M_j$, $M_{i1}'$, $M_{i2}'$)
INPUT: $M_i$ and $M_j$
OUTPUT: $M_{i1}'$, $M_{i2}'$

1. Find features $M_{i1}'$ and $M_{i2}'$ such that
   (a) both $erv(M_{i1}')$ and $erv(M_{i2}') \subset erv(M_i)$ where $erv()$ is an effective removal volume.
   (b) $erv(M_{i1}') \cap^* P = \emptyset$ and $erv(M_{i2}') \cap^* P = \emptyset$
2. Return $M_{i1}'$, and $M_{i2}'$
END_PROCEDURE
```

3.3 Merging Machining Feature Graphs

Each alternative feature model M^i in A can be represented in a simple machining feature graph as explained earlier. Consequently, all the simple machining feature graphs *S-MFG*s must be combined into a combined machining feature graph *C-MFG* to represent all the alternative ways for machining a part. A *C-MFG* can be defined as follows: $C\text{-}MFG = S\text{-}MFG_1 \oplus S\text{-}MFG_2 \cdots S\text{-}MFG_{n-1} \oplus S\text{-}MFG_n$ where $\oplus$ is a merging operator. As shown in Fig. 8(*a*), a *C-MFG* is constructed using the following two types of merging operations.

3.3.1 Merging two S-MFGs

The merging of two simple machining feature graphs $S\text{-}MFG_1$ and $S\text{-}MFG_2$ can be constructed as follows.

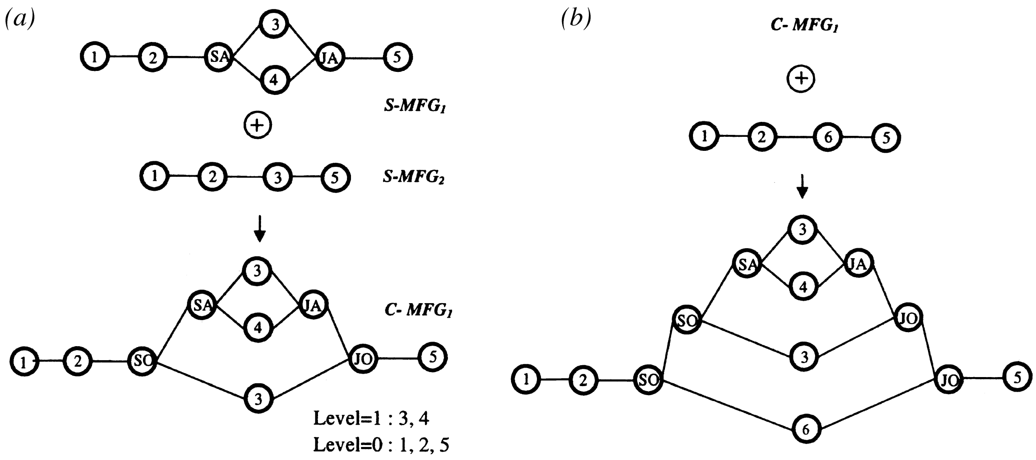

Fig. 8. Graph merging operations.

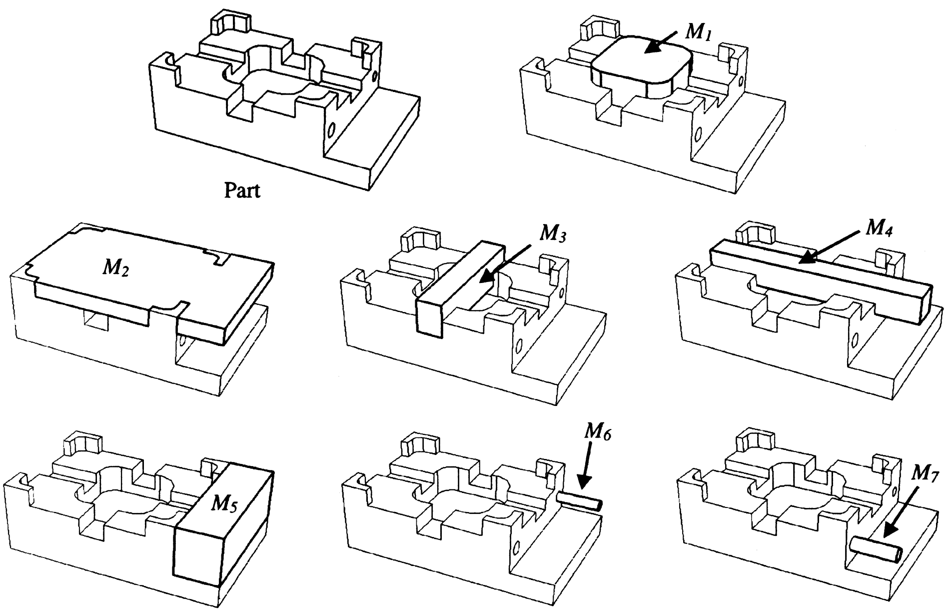

Fig. 9. A machining feature model of a designed part.

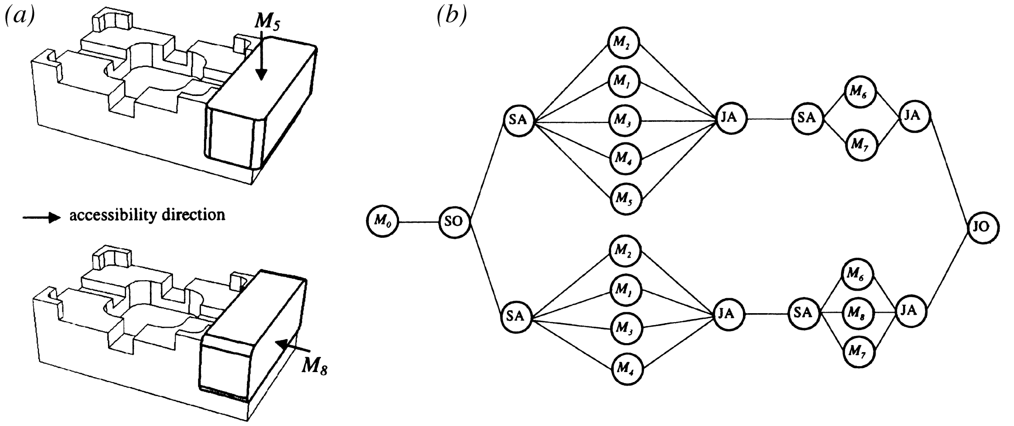

Fig. 10. Reorientation operation: (*a*) reorientation of a pocket and (*b*) a machining feature graph.

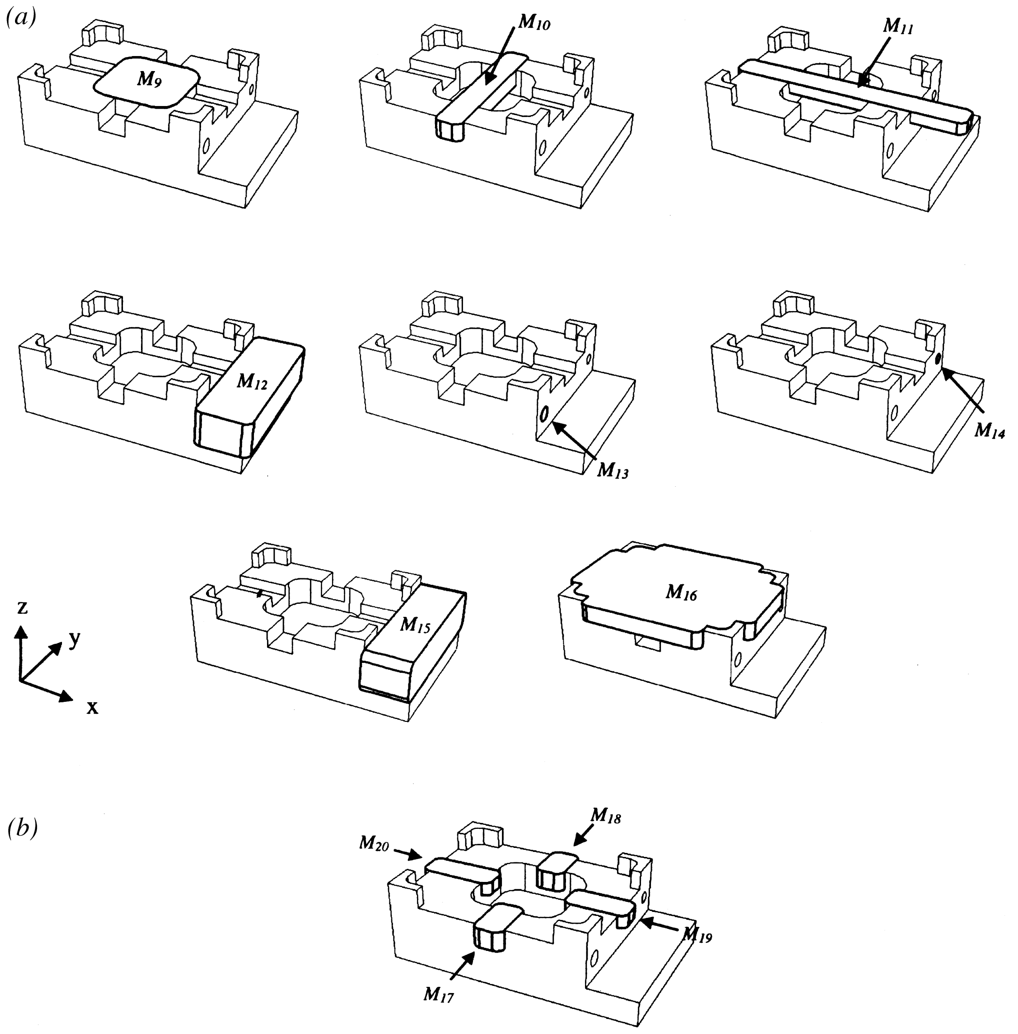

Fig. 11. Alternative interpretation: (*a*) reduction operation and (*b*) splitting operation.

1. Redefine a pair of SPLIT-AND and JOIN-AND nodes in each *S-MFG*$_i$ as a single node, so that each *S-MFG*$_i$ is sequentially ordered.
2. Find-two nodes n_1 and n_2 such that n_1 and n_2 in both *S-MFG*$_1$ and *S-MFG*$_2$.
3. Merge *S-MFG*$_1$ and *S-MFG*$_2$ at both n_1 and n_2 nodes, and insert a SPLIT-OR node after n_1 and a JOIN-OR node before n_2.
4. Repeat the above steps for the remaining parts of the two graphs until a complete *C-MFG* is generated.

3.3.2 Merging a C-MFG and a S-MFG

This operation takes similar steps as for the merging operation between two *S-MFG*s. First, all the nodes in the *C-MFG* are labelled with depth levels such that inner nodes between SPLIT-OR and JOIN-OR nodes have higher levels, as shown in Fig. 8(*a*). Initially, all the nodes in the *S-MFG* have zero levels. A new *C-MFG* can be constructed as follows.

1. Initially, *LEVEL* is set to zero.
2. Find two nodes n_1 and n_2 such that n_1 and n_2 occur in both *C-MFG* and *S-MFG*, and their levels are equal to *LEVEL*.
3. Merge *S-MFG* and *C-MFG* at n_1 and n_2, and insert a SPLIT-OR node after n_1 and a JOIN-OR node before n_2.
4. Repeat steps (2) and (3) for the remaining parts of the two graphs until no more merging operations can be carried out at the same level.
5. If merging between *S-MFG* and *C-MFG* fails, increment the level of nodes in *S-MFG* by one and set *LEVEL* = *LEVEL* + 1. Then, repeat the above steps until no graph merging is necessary (see Fig. 8(*b*)).

4. Implementation Results

The proposed approach has been implemented as a submodule of the feature-based parametric modelling system in [26]. This module has been written in C++ on an SGI Indigo2 workstation using ACIS as a solid modelling kernel.

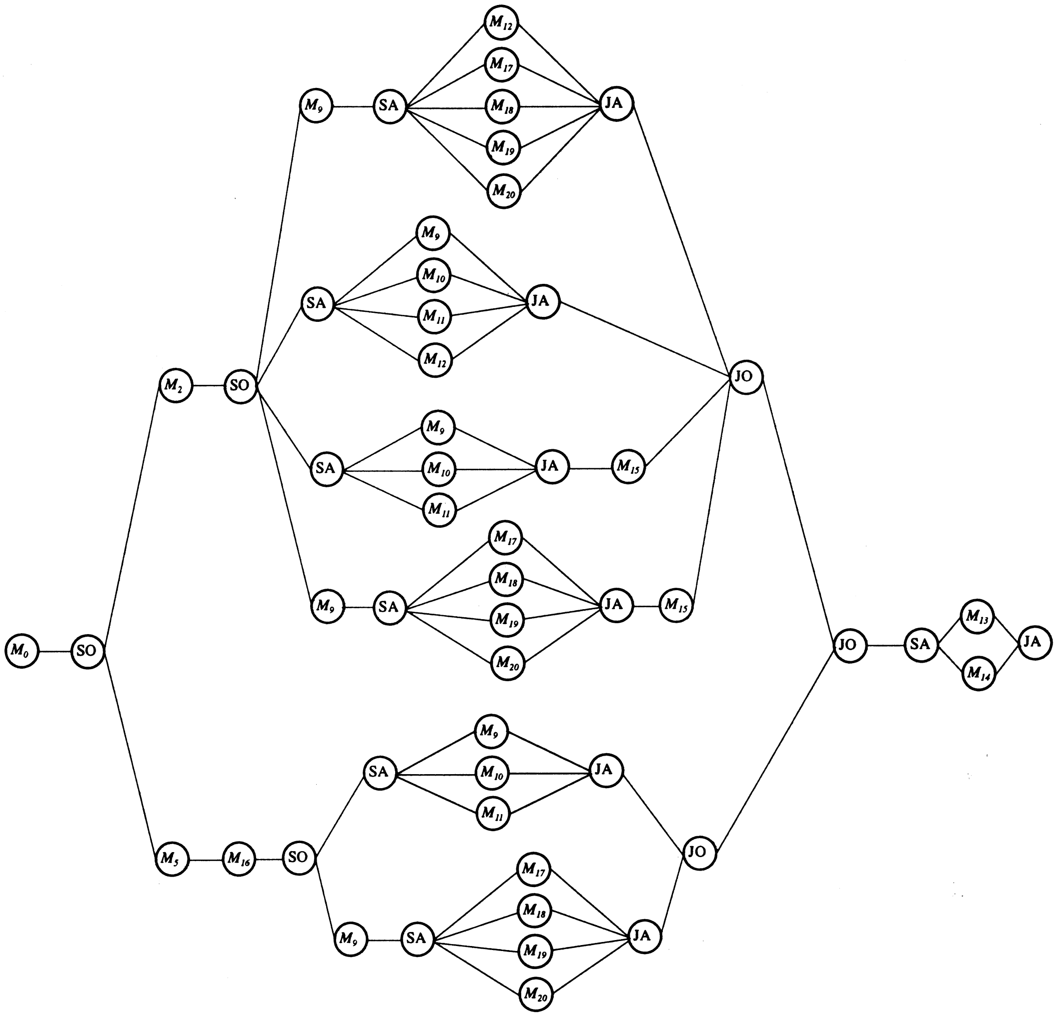

Fig. 12. A complete machining feature graph.

Figure 9 shows an initial machining feature model $M = \{M_1, M_2, \ldots, M_7\}$ to machine a part modelled by a base, 6 depressions, and 4 protrusions. Note that each machining feature is defined as a *maximum* volume [11]. By reorientation, M_8 is generated from M_5, as shown in Fig. 10. Thus, two alternative machining feature models M^1 and M^2 are generated, as shown in Fig. 10(*b*), where

$$A = \{M^1, M^2\}$$
$$M^1 = \{M_1, M_2, M_3, M_4, M_5, M_6, M_7\}$$
$$M^2 = \{M_1, M_2, M_3, M_4, M_8, M_6, M_7\}.$$

Since there are two approach directions, $[0\ 0\ 1]^T$ and $[1\ 0\ 0]^T$), either approach direction can be machined first. In this example, the set-up is ordered in the sequence of $+z$ and $+y$ approach directions for simplicity.

After reduction, the alternative model M^1 is modified into M^{1-1} and M^{1-2}, and M^2 into M^{2-1} where

$$M^{1-1} = \{M_2, M_9, M_{10}, M_{11}, M_{12}, M_{13}, M_{14}\}$$
$$M^{1-2} = \{M_5, M_{16}, M_9, M_{10}, M_{11}, M_{13}, M_{14}\}$$
$$M^{2-1} = \{M_2, M_9, M_{10}, M_{11}, M_{15}, M_{13}, M_{14}\}$$

In M^{1-1}, M_5 is reduced by M_2 into M_{12} and in M^{1-2}, M_2 is reduced by M_5 into M_{16}. As shown in Fig. 11, M_{10} is split into M_{17} and M_{18} by M_9, and M_{11} split into M_{19} and M_{20} by M_9 after the splitting operation. Thus, three more alternative interpretations are generated as follows.

$$M^4 = \{M_2, M_9, M_{12}, M_{17}, M_{18}, M_{19}, M_{20}, M_{13}, M_{14}\}$$
$$M^5 = \{M_2, M_9, M_{17}, M_{18}, M_{19}, M_{20}, M_{15}, M_{13}, M_{14}\}$$
$$M^6 = \{M_5, M_{16}, M_9, M_{17}, M_{18}, M_{19}, M_{20}, M_{13}, M_{14}\}$$

A complete machining feature graph is shown in Fig. 12. The Appendix shows a STEP physical file of the machining feature graph shown in Fig. 12.

5. Conclusion

A feature-based approach has been presented for generating alternative interpretations of machining features. An initial machining feature model is extracted from a feature-based design model, and alternative models are generated from the initial feature model by applying the proposed alternative generation operators. Since the proposed approach uses information such as design features information, nominal geometry, and functional requirements, it can generate alternative models efficiently and fast. A STEP-based feature representation scheme is used for the efficient data transfer to CAPP systems. However, there are still several issues to be studied further:

1. It would be valuable to include more complex feature types (composite features or feature groups).
2. Design rules and constraints are not yet well integrated in the system.

A process planning system based on the proposed methodology should be developed.

Acknowledgements

This research is supported in part by KRF (1996 Program).

References

1. J. J. Shah and M. Mäntylä. Parametric and Feature-based CAD/CAM: Concepts, Techniques, and Applications, John Wiley, New York, 1995.
2. S. Joshi and T. C. Chang. Graph-based heuristics for recognition of machined features from a 3D solid model, Computer-Aided Design, 20(2), pp. 58–66, 1988.
3. S. H. Chuang and M. R. Henderson. Three-dimensional shape pattern recognition using vertex classification and the vertex-edge graph, Computer-Aided Design, 22(6), pp. 377–387, 1990.
4. P. Gavankar and M. R. Henderson. Graph-based extraction of protrusions and depressions from boundary representations, Computer-Aided Design, 22(7), pp. 442–450, 1990.
5. T. C. Woo. Feature extraction by volume decomposition, in Proceedings of the Conference on CAD/CAM Technology in Mechanical Engineering, pp. 76–94, 1982.
6. Y. S. Kim. Recognition of form features using convex decomposition, Computer-Aided Design, 24(9), pp. 461–476, 1992.
7. J. H. Vandenbrande and A. A. G. Requicha. Spatial reasoning for the automatic recognition of machinable features in solid models, IEEE Transactions on Pattern Analysis and Machine Intelligence, 15(12), pp. 1269–1285, 1993.
8. W. C. Regli and D. S. Nau. Building a general approach to feature recognition of material removal shape element volumes (MRSEVs), Proceedings of 2nd Symposium Solid Modeling Foundations and CAD/CAM Applications, Montreal, Canada, pp. 293–302, 1993.
9. W. C. Regli, S. K. Gupta and D. S. Nau. Extracting alternative machining features: an algorithmic approach, Technical Report TR94–95, The University of Maryland, USA, 1994.
10. W. C. Regli. Geometric algorithms for recognition of features from solid models, PhD Thesis, The University of Maryland, USA, 1995.
11. H. Sakurai and C. Chin. Definition and recognition of volume features for process planning, in Advances in Feature Based Manufacturing, Elsevier, pp. 65–80, 1994.
12. Y. Tseng and S. B. Joshi. Recognizing multiple interpresentations of interacting machining features, Computer-Aided Design, 26(9), pp. 667–688, 1994.
13. J. Han. 3D geometric reasoning algorithms for feature recognition, PhD Thesis, The University of Southern California, USA, 1996.
14. J. J. Shah and M. T. Rogers. Expert form feature modelling shell, Computer-Aided Design, 20(9), pp. 515–524, 1988.
15. J. J. Shah. Assessment of features technology, Computer-Aided Design, 23 (5), pp. 331–343, 1991.
16. T. C. Chang. Expert process planning for manufacturing, Addison-Wesley, Reading, Massachusetts, 1990.
17. M. A. Chamberlain, A. Joneja and T. C. Chang. Protrusion-feature handling in design and manufacturing planning, Computer-Aided Design, 25(1), pp. 19–28, 1993.
18. T. Laakko and M. Mäntylä. Feature modelling by incremental feature recognition, Computer-Aided Design, 25(8), pp. 479–492, 1993.
19. T. D. Martino, B. Falcidieno, F. Giannini, S. Hassinger and J. Ovtcharova. Feature-based modelling by integrating design and recognition approaches, Computer-Aided Design, 26(8), pp. 646–653, 1994.
20. S. K. Gupta. Automated manufacturability analysis of machined parts, PhD Thesis, The University of Maryland, USA, 1994.
21. S. K. Gupta and D. S. Nau. Systematic approach to analysing the manufacturability of machined parts, Computer-Aided Design, 27(5), pp. 323–342, 1995.
22. T. R. Kramer. A library of material machining shape element volumes (MRSEVs), NISTIR 4809 NIST, Gaithersburg, Maryland, 1992.
23. T. R. Kramer. Issues concerning material machining shape element volumes (MRSEVs), International Journal of Computer Integrated Manufacturing, 7(3), pp. 139–151, 1994.
24. H. Cho. An intelligent workstation controller for computer integrated manufacturing, PhD Thesis, Texas A&M University, USA, 1993.
25. A. Derebail, H. Cho and R. Wysk. Process plan representation for shop floor control, Computer Control of Flexible Manufacturing Systems. Chapman & Hall, pp. 379–404, 1994.
26. J. Y. Lee. A knowledge-based approach to parametric feature-based modeling, PhD Thesis, POSTECH, Korea, 1998.

Appendix STEP Physical File

```
ISO-10303-21;
HEADER;
FILE_DESCRIPTION((”),’1’);
FILE_NAME(’STEPART’,’1997-12-16T22:49:53’,(’JYLEE’),(”),
’Pohang University of Science and Technology’,
’Industrial Engineering, CAD/CAM Lab.’,”);
FILE_SCHEMA((’CONFIG_CONTROL_DESIGN’));
ENDSEC;

DATA;
#100 = CARTESIAN_POINT(”,(-83.506909, 5.000000, 50.000000));
#200 = DIRECTION(”,(0.000000, 0.000000, 1.000000));
#300 = DIRECTION(”,(-0.707107, 0.707107, 0.000000));
#400 = AXIS_PLACEMENT_3D(”, #100, #200, #300);
#500 = CARTESIAN_POINT(”,(-83.506909, 36.705180, 50.000000));
#600 = DIRECTION(”,(0.000000, 0.000000, 1.000000));
#700 = DIRECTION(”,(-31.705180, 0.000000, 0.000000));
#800 = AXIS2_PLACEMENT_3D(”, #500, #600, #700);
#900 = CIRCLE(#800, 31.705180);
#1000 = TRIMMED_CURVE(#90, (0.000000), (90.000000), .T., .UNSPECIFIED.);
#1100 = CARTESIAN_POINT(”,(-115.212089, 36.705180, 50.000000));
#1200 = DIRECTION(”,(0.000000, 1.000000, 0.000000));
#1300 = VECTOR(#1200, 85.556000);
#1400 = LINE(#1100, #1300);
#1500 = TRIMMED_CURVE(#1400, (0.), (1.), .T., .UNSPECIFIED.);
#1600 = CARTESIAN_POINT(”,(-85.347996, 122.261180, 50.000000));
#1700 = DIRECTION(”,(0.000000, 0.000000, 1.000000));
#1800 = DIRECTION(”,(0.000000, 29.864093, 0.000000));
#1900 = AXIS2_PLACEMENT_3D(”, #1600, #1700, #1800);
#2000 = CIRCLE(#1900, 29.864093);
#2100 = TRIMMED_CURVE(#2000, (0.000000), (90.000000), .T., .UNSPECIFIED.);
#2200 = CARTESIAN_POINT(”,(-85.347996, 152.125273, 50.000000));
#2300 = DIRECTION(”,(1.000000, 0.000000, 0.000000));
#2400 = VECTOR(#2300, 79.570598);
#2500 = LINE(#2200, #2400);
#2600 = TRIMMED_CURVE(#2500, (0.), (1.), .T., .UNSPECIFIED.);
#2700 = CARTESIAN_POINT(”,(-5.777398, 110.940222, 50.000000));
#2800 = DIRECTION(”,(0.000000, 0.000000, 1.000000));
#2900 = DIRECTION(”,(41.185051, 0.000000, 0.000000));
#3000 = AXIS2_PLACEMENT_3D(”, #2700, #2800, #2900);
#3100 = CIRCLE(#3000, 41.185051);
#3200 = TRIMMED_CURVE(#3100, (0.000000), (90.000000), .T., .UNSPECIFIED.);
#3300 = CARTESIAN_POINT(”,(35.407653, 110.940222, 50.000000));
#3400 = DIRECTION(”,(0.000000, -1.000000, 0.000000));
#3500 = VECTOR(#3400, 74.171644);
#3600 = LINE(#3300, #3500);
#3700 = TRIMMED_CURVE(#3600, (0.), (1.), .T., .UNSPECIFIED.);
#3800 = CARTESIAN_POINT(”,(3.639075, 36.768578, 50.000000));
#3900 = DIRECTION(”,(0.000000, 0.000000, 1.000000));
#4000 = DIRECTION(”,(0.000000, -31.768578, 0.000000));
#4100 = AXIS2_PLACEMENT_3D(”, #3800, #3900, #4000);
#4200 = CIRCLE(#4100, 31.768578);
#4300 = TRIMMED_CURVE(#4200, (0.000000), (90.000000), .T., .UNSPECIFIED.);
#4400 = CARTESIAN_POINT(”,(3.639075, 5.000000, 50.000000));
#4500 = DIRECTION(”,(-1.000000, 0.000000, 0.000000));
#4600 = VECTOR(#4500, 87.145984);
#4700 = LINE(#4400, #4600);
#4800 = TRIMMED_CURVE(#4700, (0.), (1.), .T., .UNSPECIFIED.);
#4900 = PATH(closed_loop, (#1000, #1500, #2100, #2600, #3200, #3700, #4300, #4800));
#5000 = RECTANGULAR_CLOSED_PROFILE(#400, 147.125273, 150.619742, 31.705180, #4900);
#5100 = CARTESIAN_POINT(”,(-41.128781, 77.746456, 50.000000));
#5200 = DIRECTION(”,(0.000000, 0.000000, 1.000000));
#5300 = DIRECTION(”,(1.000000, 0.000000, 0.000000));
#5300 = AXIS2_PLACEMENT_3D(”,#5100, #5200, #5300);
#5400 = PLANE(”, #5300);
#5500 = POCKET_BOTTOM(”, ”, .TRUE., (”, ”, .TRUE., (#5400)));
#5600 = CARTESIAN_POINT(”,(0.000000, 0.000000, 0.000000));
#5700 = DIRECTION(”,(0.000000, 0.000000, 1.000000));
#5800 = DIRECTION(”,(1.000000, 0.000000, 0.000000));
#5900 = AXIS_PLACEMENT_3D(”, #5600, #5700, #5800);
#1 = POCKET(”, ”, .TRUE., #5900, #5000, #5500);
.
.
.
#50 = SERIAL_UNORDERED_ACTIVITY_SET(#12, #17, #18, #19, #20);
#51 = SERIAL_UNORDERED_ACTIVITY_SET(#9, #10, #11, #12);
#52 = SERIAL_UNORDERED_ACTIVITY_SET(#9, #10, #11);
#53 = SERIAL_UNORDERED_ACTIVITY_SET(#17, #18, #19, #20);
#54 = SERIAL_UNORDERED_ACTIVITY_SET(#9, #10, #11);
#55 = SERIAL_UNORDERED_ACTIVITY_SET(#17, #18, #19, #20);
#56 = SEQUENTIAL_ACTIVITY_SET(#9, #50);
#57 = SEQUENTIAL_ACTIVITY_SET(#52, #15);
#58 = SEQUENTIAL_ACTIVITY_SET(#9, #53, #15);
#59 = SEQUENTIAL_ACTIVITY_SET(#9, #55);
#60 = MULTIPLE_CHOICE_ACTIVITY_SET(#56, #51, #57, #58);
#61 = MULTIPLE_CHOICE_ACTIVITY_SET(#54, #59);
#62 = SEQUENTIAL_ACTIVITY_SET(#2, #60);
#63 = SEQUENTIAL_ACTIVITY_SET (#5, #16, #61);
#64 = SERIAL_UNORDERED_ACTIVITY_SET(#62, #63);
#65 = SERIAL_UNORDERED_ACTIVITY_SET (#13, #14);
#66 = SEQUENTIAL_ACTIVITY_SET(#0, #64, #65);
ENDSEC;
END-ISO-10303-21;
```