Contribution à l'analyse et à la réalisation d'un système de CAO à base de caractéristiques de forme

Catherine Poinsignon

THÈSE

présentée à

l'UNIVERSITÉ de METZ

Pour l'obtention du grade de

DOCTEUR DE L'UNIVERSITÉ DE METZ

Spécialité : INFORMATIQUE

par

Catherine POINSIGNON

CONTRIBUTION À L'ANALYSE ET À LA RÉALISATION D'UN SYSTÈME DE CAO À BASE DE CARACTÉRISTIQUES DE FORME

Soutenance à Metz le 19 décembre 1997

Composition du jury :

Directeur de thèse :	Yvon GARDAN	Professeur à l'Université de Metz
Rapporteurs :	Umberto CUGINI	Professeur à l'Université de Parme
	René SOËNEN	Professeur à l'Université de Valenciennes
Examinateurs :	Jean-Pierre JUNG	Professeur à l'Université de Metz
	Roland MARANZANA	Professeur à l'École de Technologie Supérieure de Montréal
	Christian MINICH	Maître de conférences à l'Université de Metz
	Denis VANDORPE	Professeur à l'Université de Lyon I

LABORATOIRE DE RECHERCHE EN INFORMATIQUE DE METZ

À mes parents

« Il n'est point de connaissance, eut-elle été cent fois confirmée pas l'expérience, qui ne soit un jour remise en question par des connaissances nouvelles, plus approchées d'une vérité fuyante, jamais tout à fait atteinte. »

VERCORS, Ce que je crois

En préambule à ce manuscrit de thèse, il me semble indispensable de remercier sincèrement un certain nombre de personnes, sans qui je n'aurais pas pu mener à bien ce travail.

C'est bien évidement à mon directeur de thèse, monsieur Yvon Gardan, que j'adresse mes premiers remerciements, pour la confiance qu'il m'a accordée et parce qu'il m'a guidée tout au long de la préparation de cette thèse.

Je tiens ensuite à remercier mes rapporteurs, messieurs Umberto Cugini et René Soënen, car ils ont accepté d'étudier et de juger mon travail, ainsi que les autres membres du jury : Jean-Pierre Jung, Roland Maranzana, Christian Minich et Denis Vandorpe.

Merci surtout à Christian, à qui revient ma plus sincère gratitude et qui le mérite sans doute le plus pour sa patience, sa disponibilité, ses conseils, ses corrections, ses critiques... Son soutien permanent a permis de me donner confiance en moi à chaque instant.

A Yann, merci pour ses précieux conseils prodigués lors de l'analyse et de la conception orientée objets, jusqu'à la programmation en C++. Merci aussi pour le visualiseur qu'il a mis à ma disposition et m'a permis d'intégrer à la maquette que j'ai développée.

Je remercie également Estelle, grâce à qui j'ai pu disposer d'un module d'évaluation d'opérations booléennes. Elle a bien voulu consacrer du temps à adapter son programme pour me fournir des renseignements supplémentaires dont j'avais besoin et pour traiter certains cas particuliers spécifiques.

Par crainte d'oublier quelqu'un, je ne citerai plus aucun nom ; je remercie simplement l'ensemble des membres du LRIM. J'adresse tout de même ma sympathie à mes collègues de bureau et à tous ceux qui font partie du noyau de l'équipe REGAIN, avec qui j'ai eu davantage l'occasion de travailler.

Enfin, mes pensées les plus chaleureuses s'adressent à mes parents, grâce à qui j'ai pu mener à bien ces nombreuses années d'études et qui m'ont toujours soutenue, particulièrement pendant la préparation de cette thèse.

TABLE DES MATIÈRES

INTRODUCTION

La modélisation représente une partie très importante des systèmes de Conception et Fabrication Assistées par Ordinateur. Les objectifs de la modélisation, dont le modèle géométrique n'est qu'une partie, sont d'une part de représenter les propriétés d'un objet, d'autre part d'exploiter le modèle, par exemple pour la visualisation, la fabrication...

Les systèmes de CFAO ont longtemps été basés sur des modeleurs géométriques (description de la forme uniquement). Or bien d'autres informations peuvent s'avérer utiles, tant pour la conception, que pour des applications telles que l'analyse, la fabrication... Une nouvelle approche élargit la connaissance du système vers des informations complémentaires à la forme ; il s'agit de la modélisation produit. Selon les auteurs, ces informations sont connues sous la dénomination d'entités, de *features* ou de caractéristiques ; c'est ce dernier terme que nous adopterons dans la suite. Parmi les diverses catégories de caractéristiques, nous nous intéresserons plus particulièrement aux caractéristiques de forme, qui permettent de déterminer la forme (géométrie, topologie) et d'y associer des informations complémentaires sur la sémantique ou la fonction.

Le concept de modélisation produit fait actuellement l'objet de beaucoup de réflexions. Il est d'abord apparu comme le moyen d'atteindre l'intégration de toutes les fonctions de la production (*Computer Integrated Manufacturing*). On a cependant reproché au CIM d'être trop séquentiel et de retarder les remises en cause de la conception. On s'oriente maintenant vers l'ingénierie simultanée (*concurrent engineering*) qui consiste à valider régulièrement la conception en la soumettant aux applications par une collaboration de l'ensemble des intervenants dans le processus de conception / fabrication (analyse de la "fabricabilité", de la résistance à la déformation, à la chaleur...). On espère donc pouvoir transformer automatiquement le modèle de conception en modèles d'applications, ce qui garantirait la validité de la conception et favoriserait

l'ingénierie simultanée. La modélisation produit apparaît ainsi comme un support indispensable de l'ingénierie simultanée.

Les travaux présentés dans ce mémoire s'inscrivent dans le cadre de la modélisation produit et consistent plus précisément à proposer un système de modélisation basé sur les caractéristiques de forme. Le système proposé remplit deux fonctions principales :

- l'insertion de caractéristiques dans la pièce en cours de conception et leur maintenance pendant toute l'évolution du produit (modification, suppression) ;
- la création par l'opérateur de nouvelles caractéristiques génériques et leur insertion dans la bibliothèque de caractéristiques de forme.

Le premier chapitre est une étude bibliographique comparative d'un certain nombre de systèmes de modélisation à base de caractéristiques. Cette étude a permis de recenser les problèmes qui y sont liés et a mis en évidence ceux qui sont mal résolus. Les critères de comparaison ont été les suivants :

- la nature de caractéristiques prises en compte ;
- le modèle développé pour les représenter ;
- l'existence d'une bibliothèque de caractéristiques et son architecture ;
- les autres traitements relatifs aux caractéristiques (extraction, génération de modèles d'applications) ;
- la validation des caractéristiques et du modèle de conception ;
- l'approche fonctionnelle.

Dans le deuxième chapitre, nous proposons une structure de données permettant de modéliser les caractéristiques de forme. Nous ferons donc un inventaire de toutes les informations à prendre en compte, pour que la sémantique de la caractéristique soit la plus riche possible. Nous verrons ensuite comment les caractéristiques ainsi modélisées pourront être utilisées dans le modèle de conception. Pour cela, nous détaillerons les étapes de la création d'une caractéristique et de son insertion dans le modèle de conception ; c'est ce que nous appellerons l'instanciation. Enfin, nous étudierons comment exploiter la structure de données proposée pour extraire des caractéristiques de forme d'un modèle géométrique et transformer le modèle de conception en modèles d'applications.

Le troisième chapitre est consacré à la gestion des modifications dans le modèle de conception. Nous proposerons pour cela une méthode qui évite de réévaluer tout l'historique de construction de la scène.

Le quatrième chapitre est dédié à l'enrichissement par l'utilisateur de la panoplie de caractéristiques de forme, disponibles dans le logiciel de CAO. Nous proposons pour cela des outils interactifs et conviviaux, permettant à l'utilisateur d'intégrer de nouvelles caractéristiques sans avoir recours au langage de programmation.

Le dernier chapitre traite de la mise en œuvre informatique du système de CAO. Une méthodologie orientée objets a été suivie pour réaliser l'analyse et la conception du système, ce qui a abouti à une identification d'un grand nombre de classes. Nous donnerons et justifierons aussi dans ce chapitre quelques choix faits lors de la programmation.

Chapitre 1 - LA MODÉLISATION À BASE DE CARACTÉRISTIQUES : ÉTUDE BIBLIOGRAPHIQUE

1. INTRODUCTION

Cette étude bibliographique fait un bilan comparatif de quelques systèmes de modélisation à base de caractéristiques décrits dans divers articles, dont certains ont été étudiés en détail dans [GMP 95]. Elle a débuté par la recherche de critères de comparaison entre les différents systèmes décrits. Certains de ces critères correspondent aux points particuliers développés dans les articles étudiés, d'autres ont été ajoutés. Un tableau comparatif selon les critères dégagés fait l'objet de l'annexe A.

Les descriptions des systèmes que nous avons étudiés ont souvent été publiées dans le cadre de recherches universitaires mais aucune publication ne fait état de l'existant dans le milieu industriel. Il est donc très difficile d'obtenir des informations sur la manière dont certaines fonctionnalités sont réalisées dans les systèmes commercialisés ; seules des démonstrations sont proposées. En ce sens, il se peut que certains aspects novateurs de systèmes industriels échappent à la synthèse qui suit.

Nous avons essayé, cependant, de dégager des propositions faites dans la bibliographie, les points qui nous semblent bien résolus, donc ce que nous retiendrons dans notre propre système. Nous mettons en évidence également les points faibles, qu'il reste à résoudre et à approfondir, ceux vers lesquels notre réflexion va être tournée.

La première partie de ce chapitre définit les caractéristiques, explique leur intérêt dans les modeleurs de CAO et la manière dont elles peuvent être exploitées. La seconde partie est l'étude comparative de différents systèmes, selon plusieurs critères.

2. CONCEPT DE CARACTÉRISTIQUES

Ce paragraphe a pour objet de préciser le sujet en donnant quelques définitions des concepts développés tout au long de ce manuscrit. L'intérêt de l'étude des caractéristiques pour la modélisation en CAO sera mis en évidence. Enfin, nous présenterons les critères de comparaison choisis pour réaliser l'étude bibliographique.

2.1. Définitions

La modélisation produit consiste à décrire une pièce, non comme un objet purement géométrique (description de la forme), mais en lui associant des informations liées à sa fonction, à sa fabrication... La tendance actuelle consiste à utiliser les caractéristiques (ou entités, ou *features*) qui permettent de décrire la forme (géométrie, topologie) et d'y attacher des informations complémentaires sur la fonction ou la sémantique. Les caractéristiques permettent la prise en compte d'aspects non géométriques et le lien entre la connaissance d'ingénierie et l'information géométrique. La définition de Shah et Mäntylä [ShM 95] est la suivante : «Par caractéristiques nous entendons les formes ou propriétés génériques d'un produit auxquelles les ingénieurs peuvent associer certains attributs et certaines connaissances (...)».

Les caractéristiques sont utilisées par le concepteur pour la description de la pièce dans le modèle de conception. Cependant, elles peuvent être spécifiques à une application donnée (préparation de la fabrication, maillage...) et dans ce cas elles sont utilisées dans les modèles d'applications. Plusieurs représentations de la même pièce peuvent donc être considérées, suivant divers points de vue. Le fait de choisir une caractéristique et de l'insérer dans le modèle sera mentionné par le terme d'instanciation. L'exemplaire de la caractéristique au sein du modèle sera donc une instance.

Selon l'information qu'elles procurent, les caractéristiques peuvent être de divers types, parmi lesquels :

- les caractéristiques de forme, décrivant des aspects locaux de la forme et leur sémantique (rainure, congé, trou...). Par extension, tout constituant de la forme finie est fréquemment considéré comme étant une caractéristique (boîte, cylindre, cône...) ;
- les caractéristiques de précision (tolérances, états de surface) ;
- les caractéristiques relatives au matériau (le matériau, ses propriétés, ses traitements spéciaux...) ;
- des informations de type filetage ou moletage, appelées des caractéristiques de texture. On pourrait les considérer comme des caractéristiques de forme mais leur faible importance dans la conception fait que l'on préfère les considérer comme des attributs ;
- des caractéristiques d'assemblage qui sont des entités capables de réaliser un assemblage entre deux pièces telles que les vis, les clavettes, les goupilles pour des assemblages rigides ou encore colonnes de guidage, des engrenages, des roulements à billes pour des assemblages articulés ;
- des caractéristiques cinématiques décrivant la manière dont les pièces interagissent (liaison rigide, pivot glissant). Notons une liaison étroite avec les caractéristiques d'assemblage et de forme. Les premières peuvent être considérées comme des attributs. Les secondes peuvent représenter des solutions technologiques ;
- des caractéristiques administratives comme la référence, l'existence en stock, le prix de revient (utiles pour la gestion de l'objet).

Cette classification ne fait toutefois pas l'unanimité. Si une face sert de référence pour le placement d'une rainure, la face et la rainure sont liées dans une relation qui fait que si l'une est modifiée, l'autre doit vraisemblablement l'être aussi. Elles méritent donc toutes deux d'apparaître comme des entités particulières ; or seule la rainure dépend de l'une des sept catégories citées (les caractéristiques de forme). En fait, la face de référence n'est pas une caractéristique en elle-même, mais elle appartient à une caractéristique. Nous conservons donc la répartition proposée ci-dessus mais en notant que, de manière générale, tout objet contraint, de quelque manière que ce soit, constitue un élément important donc une caractéristique.

Il peut être délicat quelquefois de classer une caractéristique dans une seule des catégories proposées, car il est souvent bien difficile de faire une transition nette entre deux types différents. C'est pourquoi nous admettrons qu'une même caractéristique peut appartenir à la fois à plusieurs catégories. De plus, certaines peuvent être considérées comme attributs d'autres (notamment de caractéristiques de forme).

Les caractéristiques de forme sont divisées en deux grands groupes : si elles ajoutent de la matière à la pièce, elles sont dites positives (protubérances) ; sinon, elles sont dites négatives (dépressions). On peut également distinguer les caractéristiques primaires, qui représentent la forme principale (boîte, cylindre, tore...), et les caractéristiques secondaires qui sont des altérations de la forme principale (rainure, trou, poche, bossage, congé, arrondi...). Quelques exemples de caractéristiques de forme sont représentés sur la figure 1.1.

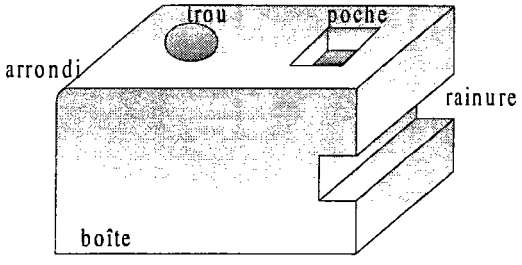


Figure 1.1 : Exemples de caractéristiques de forme.

D'autres classifications reposent sur le contexte d'utilisation des caractéristiques. Une distinction peut ainsi être faite entre les caractéristiques de conception et les caractéristiques d'application. Les premières sont utilisées directement par le concepteur pour la description de la pièce, les suivantes sont spécifiques à une application donnée, comme la préparation de la fabrication, le maillage... Par exemple, d'un point de vue fabrication, les caractéristiques représentent des formes et attributs technologiques associés aux procédés et outils de fabrication [Sha 91]. Considérons une pièce avec deux caractéristiques de forme (conception), des tenons ; la transformation en caractéristiques de fabrication (application) peut produire une rainure et deux marches (voir figure 1.2). Une seule caractéristique de conception peut donc engendrer plusieurs caractéristiques d'application. Dans cette optique, la classification de Shah et Mäntylä [ShM 95] différencie les caractéristiques de conception, de fabrication, d'inspection, d'ancrage, de coût...

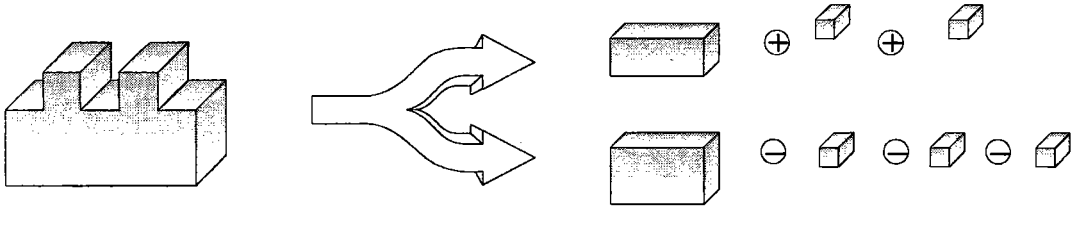


Figure 1.2 : Pièce comportant deux tenons (conception)
ou une rainure et deux marches (fabrication).

De plus, d'autres caractéristiques de conception (autres que caractéristiques de forme) peuvent également être transformées en caractéristiques d'application, en particulier les caractéristiques

de précision car on différencie les tolérances de conception des tolérances d'application. En effet, les tolérances de conception se basent avant tout sur la fonction de la pièce et indiquent dans quelles limites la pièce peut remplir cette fonction. Par contre, les tolérances de fabrication par exemple sont liées aux procédés de fabrication. La distinction entre ces deux types de tolérances est utile dans la mesure où certaines méthodes de tolérancement appliquées lors de la conception peuvent imposer un mode de fabrication, ce qui n'est pas toujours souhaitable car d'autres procédés auraient éventuellement pu donner de meilleurs résultats.

2.2. Motivations

L'utilisation des caractéristiques a d'abord été motivée par la volonté d'intégrer à la CAO la préparation de la fabrication assistée par ordinateur (CAPP = *Computer Aided Process Planning*). Les modeleurs géométriques classiques ne sont par réellement adaptés à cette intégration car ils présentent un certain nombre de déficiences [ShM 95] :

- les données disponibles dans un modeleur géométrique sont d'un niveau bas et microscopique. Or la plupart des tâches d'ingénierie nécessitent des entités macroscopiques, de niveau plus élevé. Les notions de trou, de distance, d'épaisseur ne sont par exemple pas disponibles directement dans un modeleur purement géométrique ;
- les modeleurs géométriques ne peuvent pas réellement prendre en compte l'intention de conception, c'est-à-dire la géométrie servant à satisfaire des contraintes ou des besoins fonctionnels ou d'autres notions telles que la résistance, la conductivité, la rigidité et la fabricabilité. Un modèle de plus haut niveau est nécessaire pour mémoriser ces informations dans une représentation rationnelle de la conception ;
- les modeleurs géométriques enregistrent la géométrie à un unique niveau d'abstraction, en termes d'entités géométriques dimensionnées précisément, ce qui n'est pas très approprié à l'étape de conception, mais plus utile lorsque la conception est terminée ;
- la création d'un modèle en terme d'entités de bas niveau est inefficace (car longue et fastidieuse) et elle ne permet pas de réutiliser des solutions d'ingénierie testées et sûres.

Cocquebert *et al.* [CFC 92] et De Jong *et al.* [JRH 92] confirment que les modeleurs géométriques (CSG et B-Rep) sont des techniques très performantes en ce qui concerne la géométrie (forme) mais qu'ils ne peuvent pas réellement être exploités pour la préparation de la fabrication, le contrôle de qualité, l'assemblage... Cet inconvénient est dû à l'absence

d'informations autres que géométriques, c'est pourquoi il faut pouvoir intégrer d'autres types d'information, d'un niveau sémantique plus élevé.

Les caractéristiques sont un moyen de parer à ces déficiences des modeleurs géométriques et de rassembler, en une seule entité, diverses données et méthodes de haut niveau telles que : une forme générique (topologie et/ou géométrie), des paramètres de dimensions, des paramètres contraints et des relations entre contraintes, des valeurs par défaut pour les paramètres, des méthodes de positionnement et d'ancrage, des paramètres de positionnement, des méthodes d'orientation, des paramètres d'orientation, des contraintes relatives aux dimensions, position et orientation, des tolérances, des procédures de création pour un modeleur géométrique, des algorithmes de reconnaissance, des paramètres calculés à partir d'autres caractéristiques, des règles ou procédures d'héritage, des règles ou procédure de validation, des attributs non géométriques (nombre d'exemplaires, fonctions...).

La contribution des caractéristiques dans la résolution des problèmes de la modélisation géométrique ordinaire est propre à chacun des problèmes évoqués [ShM 95] :

- le modèle produit peut être décrit en utilisant des primitives de haut niveau, au lieu des données microscopiques ; ce sont des abstractions géométriques de formes courantes. Par exemple, on manipulera des rainures, abstractions de boîtes parallélépipédiques ;
- la prise en compte de l'intention de conception est assurée par une description en termes d'entités orientées application ;
- la structure de niveau unique est remplacée par une structure multi-niveaux dans laquelle on distingue : quelques caractéristiques modélisant la forme globale de la pièce, plusieurs caractéristiques de détail modélisant des formes plus précises et la géométrie détaillée du modèle géométrique ;
- la construction fastidieuse est éliminée par une interface améliorée pour manipuler les conceptions géométriques.

La modélisation par les caractéristiques permet donc de résoudre les déficiences des modeleurs géométriques. Les caractéristiques peuvent en outre aider la fabrication ; elles sont donc un moyen de réaliser la préparation de la fabrication assistée par ordinateur (CAPP).

Selon Luby, Dixon et Simmons [LDS 86], les caractéristiques procurent un moyen de représenter explicitement l'information géométrique nécessaire pour automatiser l'évaluation de la fabricabilité, l'analyse fonctionnelle, la sélection de procédés et la préparation de la fabrication.

Pour intégrer à la fois la conception et la fabricabilité, Chen, Miller et Vemuri proposent un système composé de trois parties [CMV 91b] :

- un environnement de conception pour une définition complète de la pièce ;
- un environnement basé sur la connaissance dans lequel une pièce est décrite par des attributs génériques et des attributs spécifiques aux applications, ce qui permet d'analyser les procédés de fabrication ;
- une interface d'intégration entre ces deux environnements.

Chamberlain, Joneja et Chang soulignent que les caractéristiques de type dépression peuvent être directement associées à des procédés de fabrication ; c'est pourquoi ils s'intéressent plutôt à la façon de traiter les caractéristiques positives dans la préparation de la fabrication [CJC 93]. Comme il est moins naturel pour le concepteur de supprimer plusieurs petits volumes que d'ajouter un seul volume correspondant à la caractéristique positive, il faut transformer ces protubérances de conception en un ensemble de caractéristiques d'usinage (fabrication).

Pour Chen et LeClair, l'intérêt des caractéristiques pour la fabrication est qu'elles renseignent sur la direction d'approche de l'outil, c'est-à-dire le chemin permettant d'atteindre ce qu'ils appellent la face d'attache [ChL 94]. Ils proposent de mémoriser des règles de sélection d'outils dans une base de données. Un système d'apprentissage permet de regrouper les caractéristiques par similarité de direction d'approche et de forme d'outil. Pour usiner une caractéristique, la séquence d'outils est représentée dans un graphe acyclique orienté (DAG) où les nœuds sont des outils et les arcs des contraintes. Les DAG d'outils de toutes les caractéristiques sont combinés en un seul graphe représentant la séquence pour usiner la pièce. Cette fusion de graphes d'outils doit cependant minimiser le nombre de changements d'outils ; pour cela, les nœuds représentant un même outil sont regroupés.

Le problème majeur en conception par les contraintes d'usinage est la modélisation des contraintes et leur incorporation aux outils de conception et d'analyse. Pour Feng et Kusiac, les caractéristiques sont des objets (au sens programmation orientée objets) alors que les contraintes d'usinage sont représentées par des règles de production ou des objets [FeK 95]. Quand une caractéristique est choisie lors de la conception, les contraintes d'usinage correspondantes sont vérifiées par l'algorithme associé.

2.3. Critères de comparaison des systèmes étudiés

Pour cette étude de quelques modèles produit, nous avons choisi un certain nombre de critères de comparaison qui sont : les caractéristiques prises en compte, le modèle géométrique utilisé, l'existence d'une bibliothèque de caractéristiques, la possibilité de l'enrichir et les traitements orientés caractéristiques que l'on peut réaliser avec ces systèmes.

Nous nous intéressons tout d'abord à la structuration du modèle qui permet de représenter une pièce avec ses caractéristiques de forme. Celles-ci peuvent y être soit implicites, c'est-à-dire intégrées au modèle géométrique, soit explicites ; dans ce cas elles sont représentées dans une structure spécifique, souvent un graphe de relations. Il est également intéressant de voir comment les liaisons et interférences entre caractéristiques sont prises en compte, c'est-à-dire les relations spatiales (pour le positionnement relatif exprimé par des angles, des distances face - face, face - plan de symétrie...) et les relations d'adjacence, les contraintes liant les paramètres (par exemple le rayon d'un trou égale la moitié de la largeur d'une rainure), les intersections de caractéristiques... Les interférences peuvent être précisées par le concepteur, au moment de l'instanciation, ou bien détectées automatiquement. La gestion de ces liaisons peut être paramétrique, c'est-à-dire basée sur une résolution locale de systèmes engendrés au fur et à mesure de la construction (elle respecte donc un historique de construction) ou variationnelle c'est-à-dire basée sur une compréhension globale d'une scène construite en faisant apparaître les liens entre les objets, dans un graphe de liaisons [ChS 90].

Nous verrons ensuite que l'existence d'une bibliothèque de caractéristiques facilite la conception et assure au système de CAO une souplesse supplémentaire, car elle permet son extension et sa personnalisation par la définition de nouvelles caractéristiques par l'utilisateur. Celles-ci sont soit obtenues par une combinaison de caractéristiques primitives, soit décrites par une forme complètement inédite. La bibliothèque peut être organisée d'une manière hiérarchique, si une taxinomie de caractéristiques est proposée. L'intérêt d'une hiérarchie est de pouvoir regrouper les particularités communes à plusieurs caractéristiques à un niveau supérieur et donc de ne pas dupliquer ces informations. Par exemple, la poche, la rainure et la marche peuvent être regroupées car toutes sont créées par retrait d'un bloc parallélépipédique. Cet exemple n'est pas réaliste car il ne tient pas compte de la présence de congés de raccordement au fond des caractéristiques citées, mais il illustre simplement leur regroupement. De plus, pour la création de nouvelles caractéristiques, l'opérateur peut se servir de caractéristiques plus générales auxquelles il associerait (au niveau d'une caractéristique dérivée) des particularités supplémentaires.

En plus de la création d'une pièce, certains traitements typiquement orientés caractéristiques sont nécessaires, tels que la transformation de caractéristiques de conception en caractéristiques d'application ou encore la détection automatique de caractéristiques de forme apparues de manière implicite. Nous préciserons, dans le paragraphe 3.4, l'aptitude des différents systèmes à réaliser ces traitements.

La nécessité d'une validation des caractéristiques apparaît également dans les différents systèmes étudiés. Nous avons établi une liste des tests de validité à effectuer dans le paragraphe 3.5.

Enfin, la prise en compte de l'aspect fonctionnel dans un système de modélisation produit semble important pour compléter les caractéristiques. L'idée est de déduire une forme à partir de spécification fonctionnelle. Nous étudierons les solutions proposées au paragraphe 3.6.

Nous ne pouvons cependant pas effectuer une comparaison de tous les systèmes selon tous les critères cités ci-dessus, puisque tous n'abordent pas tous les points. Dans le comparatif qui suit, nous n'étudions donc que les aspects intéressants pour chaque système, en relevant les points positifs et en présentant les déficiences.

3. COMPARAISON DES SYSTÈMES

Nous réalisons à présent une étude comparative d'un certain nombre de systèmes de CAO décrits dans la littérature, selon les critères déterminés ci-dessus.

3.1. Les caractéristiques prises en compte

Parmi toutes les caractéristiques qui peuvent être prises en compte dans un système, les caractéristiques de forme sont les plus répandues et elles sont traitées par tous les systèmes de modélisation produit. Ensuite viennent les tolérances, qui sont souvent considérées comme des attributs des caractéristiques de forme et parfois comme des caractéristiques de précision. D'autres informations peuvent être gérées par le biais de caractéristiques, mais ne sont pas reconnues de façon unanime comme étant des caractéristiques. Les paragraphes suivants vont permettre d'établir les différents types de caractéristiques gérés par les systèmes et de proposer une représentation pour chacun d'eux.

3.1.1. Caractéristiques de forme

Les caractéristiques de forme sont des formes géométriques prédéfinies particulières qui peuvent être ajoutées à la conception d'une pièce, soit pour en modifier la forme générale, soit pour la préciser. A chaque forme est souvent associée une fonction particulière, directement liée à la géométrie. Par exemple, la forme d'une rainure implique souvent une fonction de guidage en translation. Les caractéristiques peuvent être simples ou bien composées de plusieurs autres caractéristiques, de forme ou d'un autre type (par exemple un trou taraudé est composé d'un trou, d'un filet, d'un chanfrein...). Elles sont donc l'agrégation d'entités géométriques et d'attributs tels que dimensions, type, orientation, permettant de décrire la forme et la sémantique d'une pièce.

Les caractéristiques de forme sont considérées soit comme étant des faces ou ensembles de faces [RoL 88, ReC 86], soit comme des volumes. Pour Requicha et Vandenbrande [ReV 89], les deux représentations sont utiles pour la vérification de validité. Pour Sheu et Lin [ShL 93], la représentation volumique est supérieure car les caractéristiques en sont plus faciles à manipuler. On associe alors éventuellement des entités fictives (faces de fermeture) aux objets ouverts pour en faire des volumes [CMV 91a, GoT 91]. Par exemple, une rainure peut être vue comme un ensemble de trois faces, auquel il faut rajouter trois faces supplémentaires pour obtenir une boîte (volume). Ces entités fictives disparaissent lors de l'instanciation. Elles augmentent la complexité du modèle volumique, dans la mesure où davantage d'entités topologiques sont nécessaires et où une distinction doit être faite entre les entités de la pièce et les entités de fermeture. Cependant, d'après Pratt, les caractéristiques volumiques présentent les avantages suivants [Pra 87, Pra 88] :

- les interactions entre caractéristiques sont plus faciles à gérer, car il est plus aisé de calculer des intersections entre volumes qu'entre surfaces, grâce aux opérations booléennes ;
- la suppression d'une caractéristique est plus facile à mettre en œuvre. En effet, si une caractéristique est représentée par une liste de faces, sa suppression implique la suppression de toutes ces faces, ce qui peut générer un trou dans la peau de l'objet, qu'il faut combler en prolongeant les faces restantes. En modélisant les caractéristiques par des volumes, la suppression est réalisée simplement par les opérations booléennes ;
- l'utilisation de caractéristiques volumiques résout le problème des faces partielles, quand une surface est partagée par plusieurs volumes adjacents. En effet, si l'une des deux caractéristiques qui partagent la face est détruite, c'est-à-dire si toutes ses faces sont

supprimées, la face partagée reste présente et une incohérence topologique apparaît, comme le montre la figure 1.3 ;

- il est facile de transformer les caractéristiques qui retirent de la matière en volumes à usiner (caractéristiques de fabrication). Les faces de fermeture présentes dans les volumes, à l'inverse des ensembles de faces, donne les directions d'approche de l'outil ;
- il devient plus simple de manipuler et de vérifier le résultat d'une décomposition automatique en delta volumes du volume de matière à supprimer. Les delta volumes sont des volumes usinables, tous disjoints pour éviter d'usiner plusieurs fois la même région ; les faces par lesquelles commence leur usinage et les faces qu'ils partagent sont connues.

Figure 1.3 : Suppression d'un bossage dont une face est partagée.

En plus de la forme, les représentations des caractéristiques de forme incluent d'autres informations. Pour Roy et Lin [RoL 88], une caractéristique de forme est spécifiée par un nom (rainure, alésage...), un type (primitive ou complexe), une orientation (orthogonale ou non orthogonale), un numéro d'identification, des paramètres (variables de dimensions instanciées pour préciser la taille, relations entre divers éléments géométriques)...

Cette représentation semble incomplète car il n'y a pas a priori de représentation géométrique pour chaque caractéristique, donc les relations entre éléments géométriques sont sans doute difficiles à spécifier. De plus, des contraintes permettant de limiter le comportement de la caractéristique en cas de modification de certains attributs sont nécessaires. Des méthodes pour l'instanciation, la modification et la suppression de la caractéristique devraient également être fournies.

Les représentations de Sheu et Lin [ShL 93] et celles de Duan, Zhou et Lai [DZL 93] et de Chen et Hoffmann [ChH 95] semblent particulièrement originales. Pour [ShL 93], les caractéristiques de forme sont des volumes auxquels sont associés des contraintes et des paramètres de taille et de position. [DZL 93] et [ChH 95] représentent les caractéristiques par l'extrusion d'un contour suivant une trajectoire. Ces représentations sont décrites dans les deux paragraphes suivants. La méthode de conception par extrusion est détaillée dans [Min 91].

Pour [ShL 93], les caractéristiques de forme simples sont représentées par cinq constituants de base :

- le composant solide (B-Rep) représente le volume par une information géométrique et topologique de bas niveau ;
- les entités de mesure sont les interfaces utilisées pour attacher des paramètres au composant solide : ce sont des faces, arêtes, sommets ou encore des axes ou plans de référence ;
- un paramètre de taille est une abstraction de haut niveau d'une dimension spécifique qui contrôle une taille intrinsèque d'une caractéristique de forme ; elle fait référence à des entités de mesure qui appartiennent à la même caractéristique de forme et sa valeur correspond à un paramètre spécifique mémorisé dans le composant solide ;
- les paramètres de position sont utilisés pour représenter les relations de positionnement relatif entre caractéristiques parents et enfants (c'est-à-dire portant et porté) ; elles font référence à une entité de mesure du ou des parents et une autre du ou des enfants ;
- les contraintes sont exprimées pour restreindre le comportement d'une caractéristique de forme.

Cette représentation des caractéristiques semble tout de même limitative. En effet, la gestion des surfaces non planes notamment ne semble pas aisée, car les paramètres de taille et de position sont exprimés par des distances entre faces, ce qui implique que ces faces soient planes et parallèles. La panoplie des caractéristiques qui peuvent ainsi être prises en compte facilement semble donc réduite à des caractéristiques de géométrie relativement simple.

Dans le *Feature Solid-Modelling Tool* (FSMT) [DZL 93], les caractéristiques de forme sont vues de façon particulière. En effet, une méthode d'extrusion généralisée a été développée pour créer les différentes sortes de caractéristiques, divisées en cinq types : objets prismatiques, objets de révolution, boîte (extrusion d'un contour 2D le long d'une trajectoire fermée), surface évolutive (interpolation entre une série de coupes) et solide variant (idem mais interpolation linéaire uniquement). L'intérêt principal de la méthode d'extrusion généralisée est qu'elle met en corrélation la description de la forme et des procédés de fabrication par enlèvement de matière ; chaque motif d'extrusion correspond en effet à certaines opérations d'usinage.

Le second système dans lequel les caractéristiques sont représentées par extrusion est celui de [ChH 95]. L'originalité de ce système réside dans la manière dont les auteurs spécifient les limites des caractéristiques :

- spécification d'une dimension explicite : elle correspond à la profondeur de l'extrusion (rectiligne) ou à l'angle de révolution ;
- spécification *de - à* : la nouvelle caractéristiques s'étend d'une certaine face (désignée) à une autre ;
- spécification *de - suivant* ou *précédent - à* : l'extrusion s'étend respectivement d'une face désignée à la suivante ou à la précédente ;
- spécification *de tout - à* ou *de - à travers tout* : l'extrusion commence au début de la matière de l'objet et s'achève à la face désignée ou va de la face désignée à travers toute la matière.

Plusieurs faiblesses apparaissent dans ces systèmes basés sur l'extrusion. Contrairement à [ShL 93], [DZL 93] et [ChH 95] n'évoquent pas la manière dont peuvent être gérées les dimensions. Un certain nombre de paramètres peuvent cependant sans doute être retrouvés à partir de la trajectoire 2D. Par ailleurs, les deux systèmes étudiés ne proposent pas de méthode pour exprimer et gérer des contraintes entre caractéristiques. En outre, nous constatons que la forme de la plupart des caractéristiques usuelles peut être décrite par une extrusion, mais la généralisation à toutes les entités n'est pas immédiate, surtout pour les caractéristiques propres à une application précise, comme le montre l'exemple de la figure 1.4.

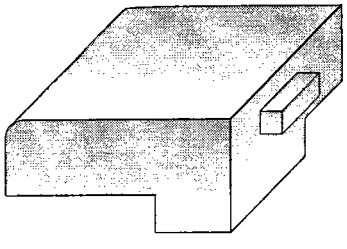


Figure 1.4 : Caractéristique ne pouvant pas être représentée par l'extrusion d'un profil

Les éléments que nous retiendrons et inclurons dans la représentation des caractéristiques de forme de notre système sont donc tout d'abord une description de la forme, par une liste de faces ou par un volume ; ce point sera discuté dans le chapitre 2. Ensuite, sur cette forme, il faut fournir un moyen de déterminer les paramètres de dimensions, position et orientation, par des distances par exemple mais pas seulement. Enfin, il faut pouvoir exprimer des contraintes qui traduisent le comportement que doit avoir la caractéristique de forme lors de son utilisation dans le modèle de conception. Ces contraintes pourront être géométriques ou exprimer des relations entre les caractéristiques.

D'autres types de caractéristiques existent et permettent de préciser davantage les caractéristiques de forme, comme les tolérances, le matériau... Elles peuvent être considérées comme des attributs des caractéristiques de forme.

3.1.2. Tolérances

Les tolérances sont des informations prises en compte par le biais de caractéristiques de précision, qui sont des attributs d'une seule caractéristique de forme (tolérance interne comme la taille) ou de plusieurs entités de caractéristiques de forme différentes (position ou localisation). Il nous a paru important de les étudier dans la mesure où elles peuvent apporter des précisions aux caractéristiques de forme.

Les tolérances, dans la norme ISO (Organisation internationale de normalisation) [Che 88], sont classées en deux catégories : les tolérances dimensionnelles et les tolérances géométriques. Les tolérances dimensionnelles permettent d'exprimer une déviation permise de la taille de l'élément tolérancé. Les tolérances géométriques sont les tolérances de forme (planéité, rectitude, cylindricité, circularité), les tolérances d'orientation (inclinaison, parallélisme, perpendicularité), les tolérances de position (localisation, coaxialité, concentricité, symétrie) et les tolérances de battement (simple ou total).

Certains auteurs utilisent cette classification standard ([RoL 88, DZL 93] et aussi [CMV 91a] qui y ajoute toutefois profil et état de surface) et d'autres proposent leur propre répartition. Ainsi dans [ShM 89], les caractéristiques de précision sont classées en quatre groupes principaux, selon la nature de l'élément sur lequel elles portent :

① sur des paramètres des caractéristiques de forme : taille (diamètre, longueur...) et forme (planéité, cylindricité...) ;

② sur des relations entre entités géométriques réelles telles que arêtes et faces (inclinaison, perpendicularité, parallélisme) ou imaginaires telles que centre de gravité et axe de symétrie (coaxialité) ;

③ sur des relations entre caractéristiques de forme : position, orientation ;

④ sur une entité géométrique de caractéristiques de forme (fini de surface).

Les caractéristiques de précision de type ① peuvent être incorporées à la liste d'attributs des caractéristiques de forme, celles de type ② et ④ doivent être attachées à des entités du modèle géométrique et celles de type ③ doivent être prises en compte au niveau des caractéristiques en fonction de leurs systèmes de référence. Cette classification particulière présente l'intérêt de

séparer les tolérances selon la nature des objets sur lesquels elles portent. Cependant, elle ne respecte pas la norme.

[RoL 88] s'appuie sur la norme ISO en séparant les tolérances en quatre catégories auxquelles correspondent quatre structures de données différentes :

- pour une tolérance dimensionnelle, un triplet avec la valeur nominale et ses variations minimum et maximum ;
- pour une tolérance de forme, le type de tolérance de forme (planéité, cylindricité,...) et la valeur de la tolérance ;
- pour une tolérance d'orientation, le type de tolérance d'orientation (inclinaison, parallélisme, perpendicularité), la valeur de la tolérance, une condition du maximum de matière (MMC = *Maximum Material Condition*) et un pointeur sur un système de référence qui peut être un système de trois plans, trois axes ou un ensemble ordonné de faces ;
- pour une tolérance de position, le type de tolérance de position, le type de l'élément tolérancé (arête, sommet,...), la valeur de la tolérance, une condition du maximum de matière (MMC) et un pointeur sur un système de référence.

Le point commun entre ces deux systèmes est une prise en compte des tolérances dimensionnelles, de forme, de position et d'orientation, ce qui est conforme à la norme ISO. À cela, [ShR 88a] ajoutent des tolérances liées aux entités virtuelles (axes et plans de symétrie) ce qui implique une modélisation de telles entités dans le système. De plus, ils permettent d'appliquer une tolérance à une partie de caractéristique (une face) en autorisant des tolérances sur les surfaces. Ces deux compléments sont sans doute des points forts de [ShR 88a] par rapport à [RoL 88].

Pour représenter les tolérances, il ne suffit cependant pas de proposer une structure de données pour chaque type de tolérance prise en compte. En effet, la spécification de tolérance implique souvent la spécification de systèmes de référence (pour établir la position et l'orientation d'une caractéristique par rapport à d'autres). De plus, [RoL 88] met en évidence la nécessité de spécifier des relations entre cinq sortes d'éléments de tolérance : sommet, arête, surface, axe et plan médian. Les trois premiers sont impliqués dans les relations d'une caractéristique avec elle-même ou entre deux caractéristiques, alors que les deux derniers éléments sont uniquement impliqués dans les relations entre deux caractéristiques distinctes.

Deux principaux types de représentation des tolérances sont décrits par [RLW 91, Jus 92] :

- la représentation par paramétrage, dans laquelle la taille et la forme d'un objet sont contrôlées par un ensemble de paramètres du modèle géométrique. On distingue le paramétrage direct, où l'utilisateur donne explicitement les paramètres du modèle en fonction des dimensions de l'objet, et le paramétrage indirect, où l'utilisateur spécifie d'abord le modèle et y attache ensuite les dimensions ;
- la représentation par *offset* calculé à partir des limites du solide idéal et détermine une zone de tolérance qui permet de contraindre les variations de taille, de forme, d'orientation et de position. L'offset est une surface qui est en tout point à la même distance de ces limites.

Le paramétrage est utilisé par [RoL 88] qui travaillent avec un référentiel et des relations entre les éléments et par [GZS 88, FaF 89] qui associent les tolérances aux caractéristiques et aux faces à l'aide d'un opérateur de positionnement. Par contre, dans [Req 83, Req 84, ReC 86], les zones de tolérances sont calculées par *offset* pour les tolérances dimensionnelles et géométriques. Enfin, [IMK 94] utilisent les deux représentations : le paramétrage pour les tolérances dimensionnelles et les *offsets* pour les tolérances géométriques.

Le type de tolérance influe donc sur le type de représentation, mais aussi sur l'instant auquel elles peuvent être représentées. D'après Chen, Miller et Vemuri [CMV 91a], les tolérances peuvent ne pas toutes être spécifiées au même moment, selon leur type. En effet, les tolérances de taille, qui sont appliquées aux dimensions tolérancées, peuvent être spécifiées durant l'instanciation des caractéristiques de forme. Les tolérances de position, qui mettent en relation des entités ou des référentiels, peuvent être spécifiées durant le placement des caractéristiques. Les tolérances de forme et d'orientation, qui s'appliquent à des entités individuelles telles que surface, arête ou axe, sont spécifiées après l'évaluation finale de la pièce.

Les tolérances sont souvent vues comme des attributs de caractéristiques de forme. Dans les propositions qui seront faites par la suite, nous ne nous attacherons donc plus aux tolérances, car nous supposerons que c'est une information qui peut être prise en compte par une couche supplémentaire dans un système de modélisation par les caractéristiques.

D'autres informations peuvent également être prises en compte au moyen de caractéristiques. Il s'agit en particulier du matériau, des contraintes, des textures, du débit...

3.1.3. Autres informations

D'autres informations devraient être prises en compte pour qu'un modeleur soit le plus complet possible et qu'il rende exploitables toutes les données de haut niveau. Il s'agit notamment de toutes les informations utiles pour la préparation de la fabrication.

Les caractéristiques de matériau sont mémorisées dans une bibliothèque, puis attachées aux caractéristiques de forme en tant qu'attributs. Les données sous-jacentes sont des types suivants [ShR 88a] :

- nom du matériau ;
- composition du matériau ;
- propriétés physiques et mécaniques ;
- traitement thermique à appliquer à la pièce entière ;
- traitement de surface.

Les informations technologiques selon [RoL 88] sont des caractéristiques spéciales, attachées aux caractéristiques de forme ou aux pièces appropriées au moyen d'attributs. Ce sont des caractéristiques géométriques complexes telles que filetage, engrenage, moletage, texture de surface... Elles ne sont pas considérées comme des caractéristiques de forme car elles sont localisées (les textures ne s'appliquent qu'à une partie de caractéristique de forme) et par souci de performance (car elles sont longues à dessiner).

3.1.4. Synthèse

La représentation des caractéristiques de forme doit avant tout permettre une description précise de la forme, par une liste de faces ou par un volume. Ensuite, il faut pouvoir associer à cette forme des informations d'un niveau sémantique supérieur telles que les paramètres de dimensions, la position et l'orientation. Enfin, il faut pouvoir exprimer des contraintes qui traduisent le comportement que doit avoir la caractéristique de forme lors de son utilisation dans le modèle de conception. Ces contraintes pourront être purement géométriques (tangence, parallélisme, distance, angle...) ou exprimer des relations entre les caractéristiques (portant - porté) ou entre paramètres de caractéristiques (équations).

3.2. Les modèles utilisés

Un modèle basé sur les caractéristiques est composé d'une couche sémantique, qui permet de mémoriser les instances de caractéristiques et leurs relations, et d'un modèle géométrique (parfois dérivé du premier). Les modèles géométriques utilisés couramment en CAO sont les suivants : le modèle par les limites ou par les frontières ou B-Rep (*Boundary Representation*) et le modèle par historique ou arbre de construction ou CSG (*Constructive Solid Geometry*) [Gar 91]. Le modèle B-Rep comprend des informations topologiques et géométriques ; il est donc aisé d'y associer des cotations ou des tolérances. La représentation CSG conserve, pour un objet donné, les opérations qui ont été mises en œuvre pour le créer (opérations booléennes, transformations géométriques). Il permet donc de conserver une partie de l'historique de construction de l'objet.

Une structure complémentaire est souvent associée au modèle géométrique et forme la couche sémantique. Elle permet une description explicite des relations entre caractéristiques (graphe), ces relations pouvant être de simples liens d'adjacence, des relations spatiales ou encore des relations de dépendance. [DoW 90] définit par exemple une relation de dépendance entre deux caractéristiques si une modification de l'une entraîne un changement du type de l'autre. Il les classe en dépendances d'existence (si la suppression d'une caractéristique entraîne la modification du type de l'autre) et dépendance de taille (si la modification d'un paramètre de l'une entraîne un changement de type de l'autre). Il propose de représenter ces dépendances par des graphes.

Les systèmes étudiés peuvent être regroupés en trois types d'organisation : un modèle hybride CSG - B-Rep (§ 3.2.1), un modèle basé sur un CSG ou un B-Rep associé à un graphe (§ 3.2.2), un modèle hybride avec un graphe (§ 3.2.3). Les liens entre l'historique et le modèle géométrique peuvent être de natures différentes ; ils seront présentés dans le paragraphe 3.2.4.

3.2.1. Modèle géométrique hybride CSG - B-Rep

La structure utilisée dans le système FSMT (*Feature Solid-Modelling Tool*) [DZL 93] est composée d'un historique de construction et d'une représentation par les limites. Dans le pseudo-CSG, chaque caractéristique, représentée par un profil 2D transformé en un volume 3D par une extrusion généralisée le long d'un guide, est représentée par un nom, une trajectoire (composée de plusieurs éléments) et une génératrice (divisée en plusieurs composants

également). Quand le modèle CSG a été évalué, un modèle B-Rep polyédrique est obtenu et des liens sont établis à partir du CSG vers chaque face du modèle B-Rep. En fait, chaque face du B-Rep est liée à un élément de trajectoire et un élément de génératrice dans la représentation CSG, comme le montre la figure 1.5. Les relations entre caractéristiques sont uniquement des positionnements relatifs et leur gestion est paramétrique. Les interférences entre caractéristiques sont résolues par le processeur d'opérations booléennes.

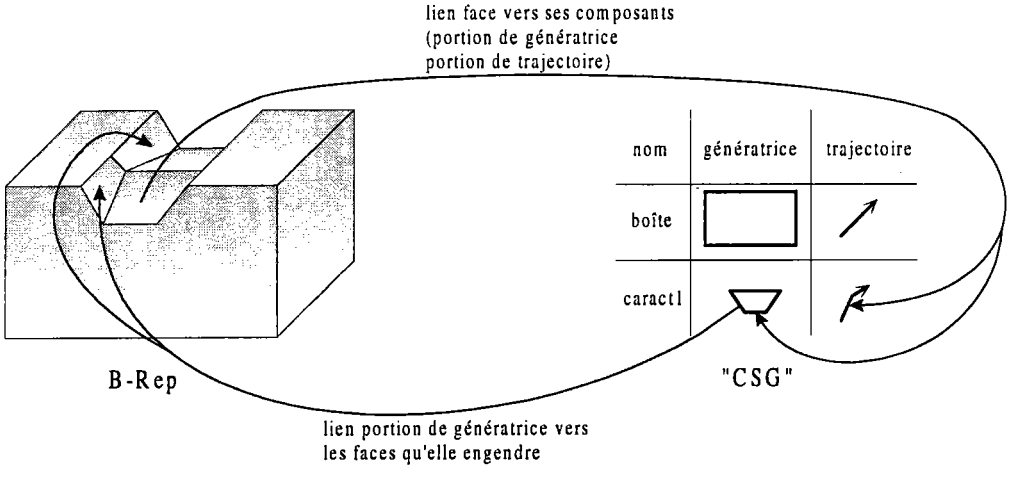


Figure 1.5 : Liens entre les représentations CSG et B-Rep dans le système FSMT [DZL 93].

Il est donc nécessaire de relier les composants de l'historique aux entités du modèle évalué. Même si deux modèles géométriques maintenus en parallèle, un B-Rep et un CSG, ne sont pas forcément nécessaires, il faut tout de même une relation qui fasse la liaison entre une caractéristique et ses composants dans la représentation évaluée.

3.2.2. Modèle CSG ou B-Rep associé à un graphe

[ReC 86] proposent une représentation des tolérances en associant un CSG à un graphe variationnel appelé *VGraph* et représenté par la figure 1.6. Cette représentation ne permet pas réellement une modélisation des caractéristiques, mais plutôt une désignation interactive des composants des caractéristiques par l'utilisateur, sans mémorisation dans le modèle. Le *VGraph* est composé de nœuds de différents types : des portions de faces identifiées par l'utilisateur, des groupes de portions de faces, des sous-ensembles d'arêtes déterminés par l'utilisateur, des groupes d'arêtes, des attributs de tolérance et des systèmes de référence. Les liens entre le *VGraph* et une représentation géométrique (B-Rep ou CSG) sont établis par les *NFaces*. Il suffit de les identifier aux faces du B-Rep ou aux faces des primitives dans un CSG. Cette structure offre divers avantages parmi lesquels :

- la possibilité d'intégrer des variations par rapport aux cotations nominales de l'arbre de construction ;
- la possibilité pour l'opérateur de facilement mettre à jour les valeurs des tolérancements au fur et à mesure de l'avancement de son projet.

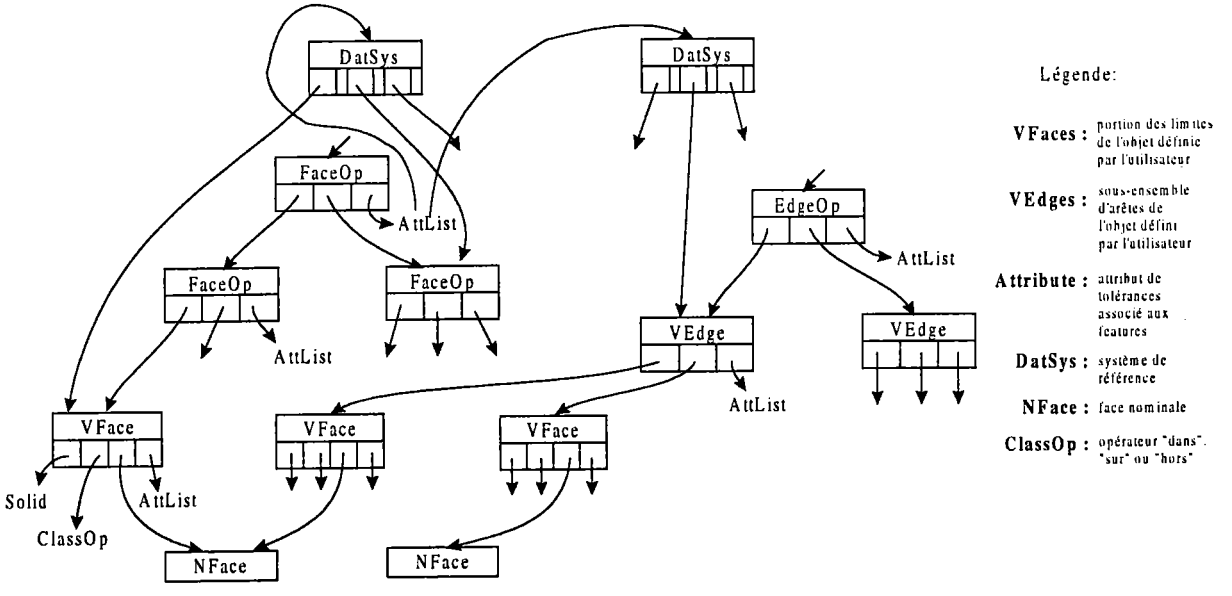


Figure 1.6 : Structure du *VGraph* [ReC 86].

Le point positif que nous pouvons dégager de cette étude est le fait de pouvoir utiliser indifféremment un B-Rep ou un CSG comme modèle géométrique associé au *VGraph*. Nous remarquons toutefois que les arêtes nées de combinaisons booléennes ne sont pas accessibles. On ne peut donc pas y associer de tolérances.

Le système de [CMV 91a] n'est pas orienté tolérances, comme le précédent. Il est composé d'une structure à trois niveaux :

- un graphe de relations de caractéristiques dont les nœuds sont soit des caractéristiques, soit des relations qui peuvent être *partie de* (relation entre une entité et ses composants), *est dans*, *est sur* ou *adjacent à* (voir figure 1.7). La relation *est dans* est similaire à une opération de soustraction ; elle établit une relation de dépendance d'existence entre une caractéristique négative et une positive, et une relation de dépendance géométrique. Une relation de dépendance d'existence implique que si la caractéristique positive est supprimée, la négative disparaît simultanément. La dépendance géométrique exprime par exemple le fait que si la hauteur d'un bloc change, la profondeur du trou traversant est automatiquement ajustée. La relation *est sur* réalise une opération d'union (de deux caractéristiques positives ou deux négatives) et établit une relation de dépendance ainsi qu'une relation spatiale entre deux caractéristiques ;

- une évaluation des limites actives (B-Rep) de chaque caractéristique qui contient des entités géométriques (faces, arêtes, sommets) ;
- une évaluation de la pièce finale par calcul et lien des B-Rep de toutes les caractéristiques.

Il est possible d'établir des relations entre dimensions par le biais de contraintes exprimées par des inégalités et des équations au moment de l'instanciation des caractéristiques. La gestion de ces liaisons est vraisemblablement variationnelle, bien que les auteurs ne donnent pas de précision à ce sujet.

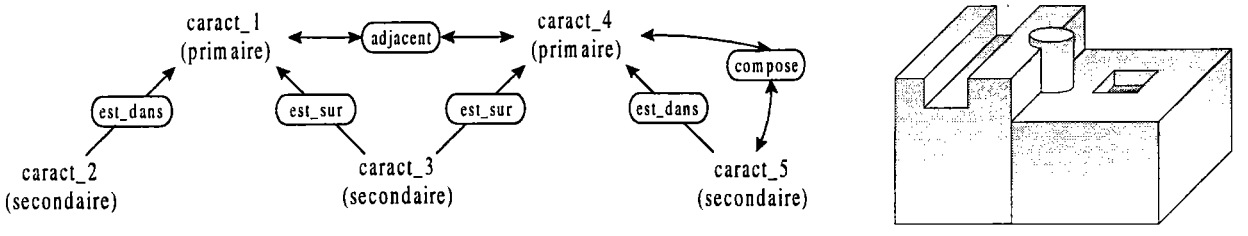


Figure 1.7 : Graphe de relations de caractéristiques [CMV 91a].

Cet exemple de structure montre bien la notion de multi-niveaux présente dans les modeleurs à base de caractéristiques. Il établit la nécessité d'un graphe de relations entre les caractéristiques, dans lequel seront présentes les relations portant - porté (*est dans*, *est sur*) ainsi que les relations de composition (*fait partie de*). Cependant, la représentation évaluée est toujours indispensable. Le point qui mérite réflexion est la nécessité d'avoir un B-Rep pour chaque caractéristique en plus du B-Rep de la pièce. Les choix pour notre propre structure seront discutés dans le chapitre 2.

Le schéma de représentation de [ShL 93] est similaire dans la mesure où une représentation B-Rep est également associée à chaque caractéristique. Les caractéristiques sont arrangées dans un graphe de dépendance (FDG = *Feature Dependency Graph*) en fonction de leur position relative (figure 1.8). Les nœuds sont de deux types : les caractéristiques de forme et les opérateurs de positionnement de caractéristiques (FPO = *Feature-Position Operator*) qui représentent les relations de positionnement relatif entre deux nœuds caractéristiques.

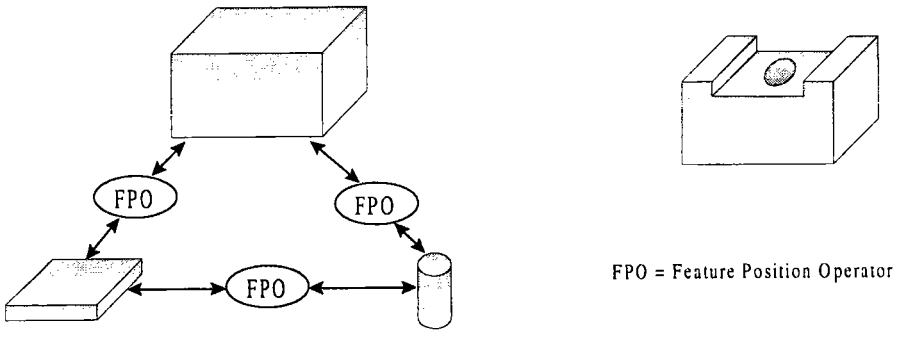
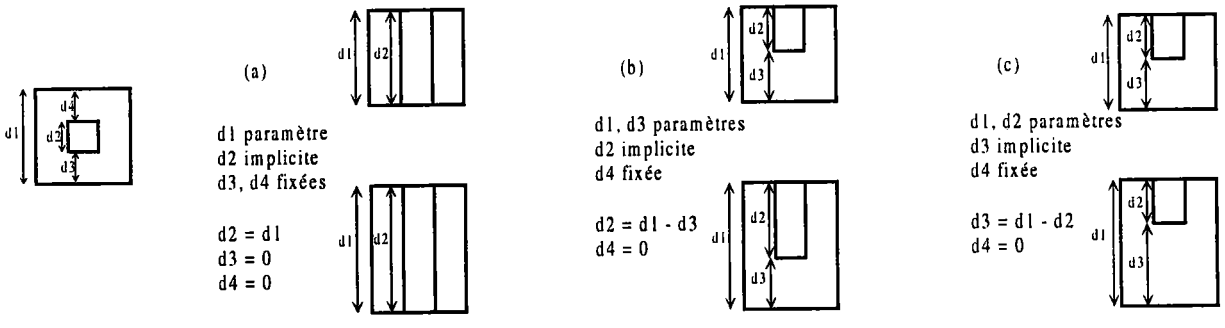


Figure 1.8 : Représentation du graphe de dépendance (FDG) [ShL 93].

À nouveau, le B-Rep final de la pièce est obtenu par évaluation du graphe FDG ; elle est nécessaire pour une visualisation ou quand des caractéristiques sont ajoutées ou supprimées du modèle FDG ou encore quand certaines dimensions sont modifiées par l'utilisateur. Les seules contraintes prises en compte sont exprimées par des distances dans une relation père - fils et dans une même direction. Sur l'exemple du trou ci-dessous, on cote des distances entre les faces inférieures et supérieures de la pièce et du trou, le trou étant *fils* de la pièce, dans la direction de la hauteur. Les contraintes sont indiquées au moment de l'instanciation (positionnement relatif). Deux types particuliers de dimensions sont utilisés dans des contraintes : les dimensions fixées, dont la valeur est déterminée avant que la caractéristique ne soit créée, et les dimensions implicites, dont la valeur peut être calculée automatiquement et dynamiquement. Les dimensions ni fixées, ni implicites sont des valeurs de conception (c'est-à-dire des paramètres).



La contrainte fixée dans le cas (a) indique que la profondeur *d2* du trou est égale à la hauteur *d1* de la pièce, c'est-à-dire que le trou est traversant et le reste lors d'une modification. Les dimensions entre les bases du trou et la pièce sont limitées (*d3* et *d4* sont égales à zéro) et la profondeur du trou est implicite (déterminée en fonction de la hauteur de la pièce : *d2* = *d1*). Dans le cas (b), la profondeur est une dimension implicite, qui est donc recalculée en fonction des autres dimensions, lors d'une modification (*d2* = *d1* - *d3*). Dans le cas (c), la profondeur du trou est une valeur de conception, elle n'est donc pas changée par la modification. La gestion des contraintes prises en compte est variationnelle.

Figure 1.9 : Exemple de comportements pour un trou [ShL 93].

Les contraintes sont des chaînes de dimensions linéaires. Des comportements différents sont attribués à la pièce par la nature des contraintes qui lui sont associées. Par exemple, si un trou est

placé dans une pièce, quand la taille de la pièce est modifiée, le comportement du trou est différent en fonction des dimensions établies, comme le montre la figure 1.9.

La notion de comportement associé à une caractéristique est prise en compte dans ce système, mais de façon très restrictive car les contraintes ne peuvent être établies que par des distances, ce qui implique que les faces soient planes et parallèles. Il faut donc enrichir cette proposition par un plus grand nombre de contraintes.

Le système de [MFG 94] est composé de trois parties principales : un module de conception par les caractéristiques, un module de reconnaissance de caractéristiques et un modeleur géométrique. Plusieurs modèles sont maintenus simultanément (voir figure 1.10) :

- le modèle basé sur les caractéristiques de conception est un graphe de relations entre les caractéristiques explicites. Les nœuds correspondent à la description paramétrique des caractéristiques de conception (prédéfinies ou définies par l'utilisateur) et les arcs représentent les relations spatiales et topologiques (adjacence) entre caractéristiques ;
- le modèle intermédiaire mémorise les instances des caractéristiques de forme qui sont créées et manipulées par le module de conception ou qui sont extraites du modèle géométrique par le module de reconnaissance. C'est un graphe hiérarchique dont les nœuds sont des volumes représentés par les limites des caractéristiques et les arcs sont des relations d'adjacence entre ces volumes ;
- le modèle géométrique est une représentation B-Rep de la pièce ;
- divers modèles d'applications sont tirés du modèle intermédiaire par le module de conversion de caractéristiques de conception en caractéristiques d'application. Pour cela, on utilise des opérations de simplification et de subdivision, en appliquant des règles dépendant du contexte. Ces opérations sont précisées dans le paragraphe 3.4.2.

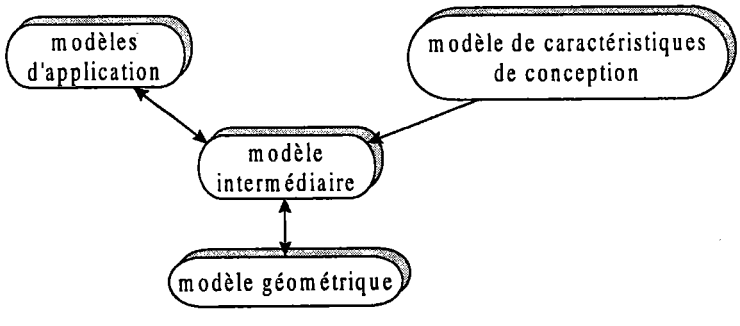


Figure 1.10 : Les différents modèles du système [MFG 94].

Les relations spatiales et topologiques des caractéristiques sont donc prises en compte dans le graphe de caractéristiques et leur gestion est paramétrique. Le volume de chaque caractéristique

de conception est inclus au modèle de la pièce en utilisant les opérations booléennes. Ce processus détecte les intersections entre les volumes des caractéristiques de conception et du modèle de la pièce et mémorise l'information pour des manipulations ultérieures. Par exemple, si une caractéristique de conception est supprimée, la pièce peut être reconstruite sans réévaluer tout le modèle.

Ce que nous retiendrons pour notre système est la conservation des relations spatiales et topologiques, ainsi que la détection et la mémorisation des interférences entre caractéristiques.

Le souci d'intégrer la conception par les caractéristiques et la reconnaissance est repris dans le système EXTDesign [LaM 93]. Le concepteur peut modéliser une pièce interactivement en utilisant les opérations de modélisation géométrique ou basées sur les caractéristiques. Deux modèles sont gérés en parallèle : un modèle géométrique et un modèle à base de caractéristiques. Grâce au module de reconnaissance des caractéristiques et au processus d'instanciation géométrique, chacune des représentations peut être éditée alors que l'autre est maintenue cohérente avec celle qui est éditée. Pendant la conception, l'utilisateur peut ainsi agir sur l'un ou l'autre des modèles.

Pour la modélisation géométrique, une représentation B-Rep est utilisée. Elle suit une approche dans laquelle deux sortes de représentations géométriques sont utilisées : les modèles polyédriques et les modèles de surfaces.

Le modèle à base de caractéristiques est composé d'une bibliothèque de caractéristiques et d'un ensemble d'instances. La bibliothèque est constituée de classes de caractéristiques qui mémorisent l'information générique commune à toutes les caractéristiques du même type. Les instances de caractéristiques mémorisent l'information spécifique, relative aux caractéristiques individuelles. Des relations parent - enfant entre les instances de caractéristiques modélisent la façon dont les caractéristiques sont positionnées les unes par rapport aux autres. Ces relations sont représentées dans une structure d'arbre dans laquelle les nœuds sont des instances de caractéristiques. Pour interfacer le modèle géométrique avec les caractéristiques, un graphe d'adjacence des faces, présentant toutefois trois particularités, est introduit (figure 1.11). Les facettes qui approchent une même face (par exemple un cylindre polygonalisé) sont réunies dans un unique nœud. Chaque face porte une référence à la caractéristique de forme qui la porte. Les arcs portent une mention "convexe", "lisse", "concave" ou "indéfini", correspondant à l'arête qu'ils modélisent.

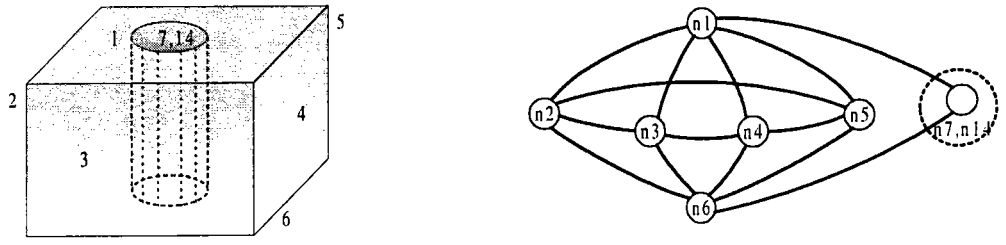


Figure 1.11 : Représentation de graphe d'adjacence (SAAG) [LaM 93].

Encore une fois, un modèle géométrique de la pièce complète est obtenu en combinant les volumes des instances de caractéristiques par les opérations booléennes, c'est-à-dire en parcourant l'arbre de la hiérarchie d'instances et en l'interprétant comme un arbre CSG.

Plusieurs des systèmes étudiés stockent les B-Rep des instances de caractéristiques et les combinent pour obtenir la pièce finale. L'obtention du B-Rep exact de l'instance n'est cependant pas détaillée. Une première solution envisageable est d'associer à chaque caractéristique un B-Rep paramétré ; il suffit donc de préciser les paramètres lors de l'instanciation. Cette solution semble bien adaptée aux caractéristiques primaires, mais pas aux secondaires. En effet, il est très difficile d'avoir un B-Rep à topologie et paramètres suffisamment général pour espérer couvrir tous les cas d'instanciation, par exemple ceux de la figure 1.12 pour une rainure.

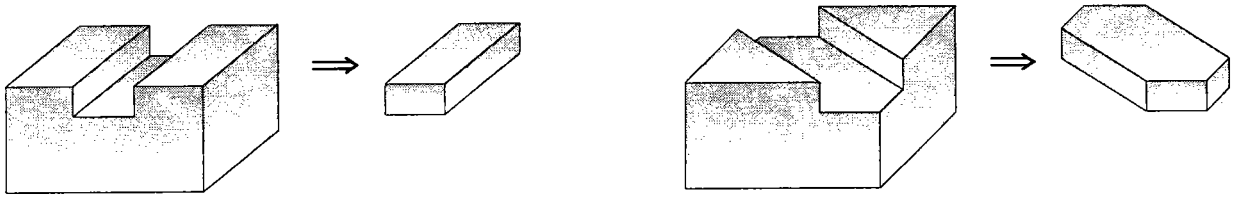


Figure 1.12 : Exemples d'instanciation de rainure.

Une seconde solution est d'avoir un B-Rep surdimensionné pour chaque caractéristique, qui permet de calculer le B-Rep correspondant exactement au volume de matière à ajouter ou à soustraire, par une opération booléenne. L'inconvénient est que le nombre de paramètres à préciser est supérieur au nombre de paramètres réellement nécessaires. Comme il nous paraît intéressant de stocker ce volume exact, nous proposons dans le chapitre 2 un moyen de le calculer en évitant les inconvénients cités.

3.2.3. Modèle hybride CSG - B-Rep avec un graphe

Le système FBMS (*Feature-Based Modelling System*) est composé de trois modeleurs distincts [ShR 88a] :

- le modeleur par les caractéristiques de forme ;

- le modeleur par les caractéristiques de précision ;
- le modeleur par les caractéristiques de matériau.

Les caractéristiques de forme sont utilisées comme la représentation fondamentale ; les caractéristiques de précision et de matériau peuvent être définies indépendamment puis reliées aux caractéristiques de forme. Le modeleur par caractéristiques de forme est structuré en quatre niveaux, représentés sur la figure 1.13 :

- un graphe de relations de caractéristiques qui maintient les liens d'adjacence et les liens père - fils entre caractéristiques ;
- la représentation de chaque caractéristique de forme c'est-à-dire des propriétés génériques et des instances. Les propriétés génériques sont : la liste des paramètres donnés par l'utilisateur, la liste des paramètres dérivés, l'arbre booléen FPV (*Feature Producing Volume*) de la caractéristique, les règles et expressions permettant de préciser un comportement... Les instances sont précisées par les valeurs des paramètres, la matrice de changement de repère pour le positionnement relatif... ;
- un arbre CSG construit avec les volumes FPV de toutes les instances du modèle associés à des opérateurs (différence, union...) ;
- l'évaluation des frontières actives (B-Rep) si nécessaire (pour procurer une réponse rapide à l'utilisateur lors d'interactions avec le modèle), représentant le niveau le plus bas de l'information géométrique et topologique.

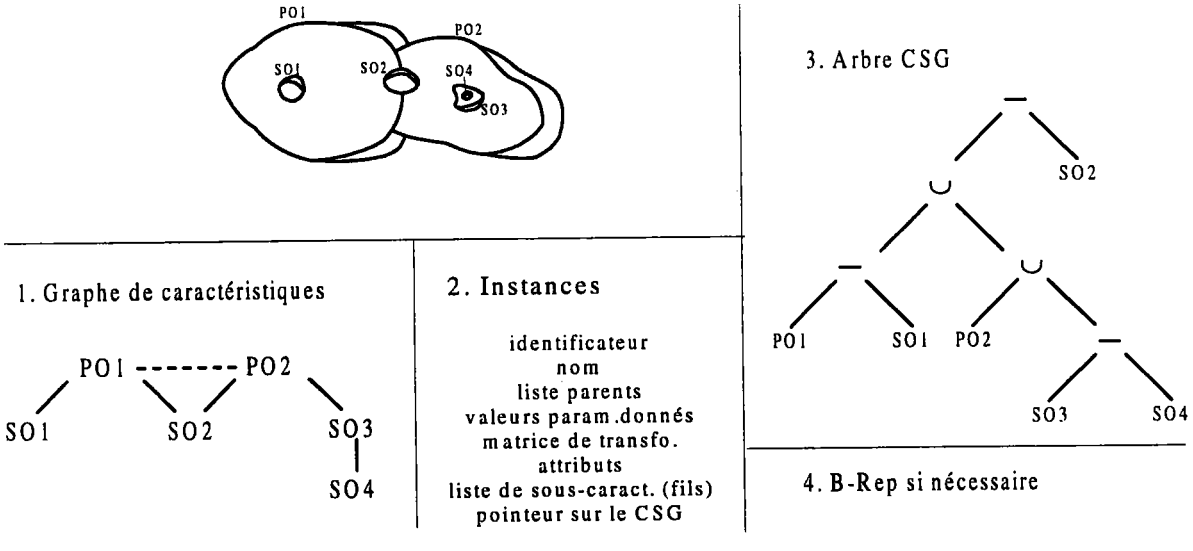


Figure 1.13 : Les quatre niveaux du modeleur par les caractéristiques de forme dans FBMS [ShR 88a].

Les liaisons pouvant être gérées par le système sont de deux types :

- les relations d'héritage de paramètres, quand un paramètre d'une caractéristique fille est exprimé en fonction d'un paramètre du père (par exemple, la profondeur d'un trou traversant est égale à la hauteur de la boîte le contenant) ;
- les relations d'adjacence (positionnement relatif) indiquées au moment de l'instanciation. La gestion de ces contraintes est certainement paramétrique.

Le modèle proposé par [GoT 91] se divise en deux niveaux : au niveau le plus haut, un graphe d'adjacence de caractéristiques et au niveau le plus bas, deux structures de données associées, une structure topologique de caractéristiques et un B-Rep. Le graphe d'adjacence de caractéristiques (FAG) est composé de nœuds qui sont des volumes positifs ou négatifs (caractéristiques de forme) et d'arcs qui sont des relations d'adjacence ou d'interaction entre ces volumes. Un exemple d'un tel graphe est donné par la figure 1.14. Il y a une relation d'interaction entre deux caractéristiques si les limites ou le volume de la première sont changés par l'insertion de la seconde.

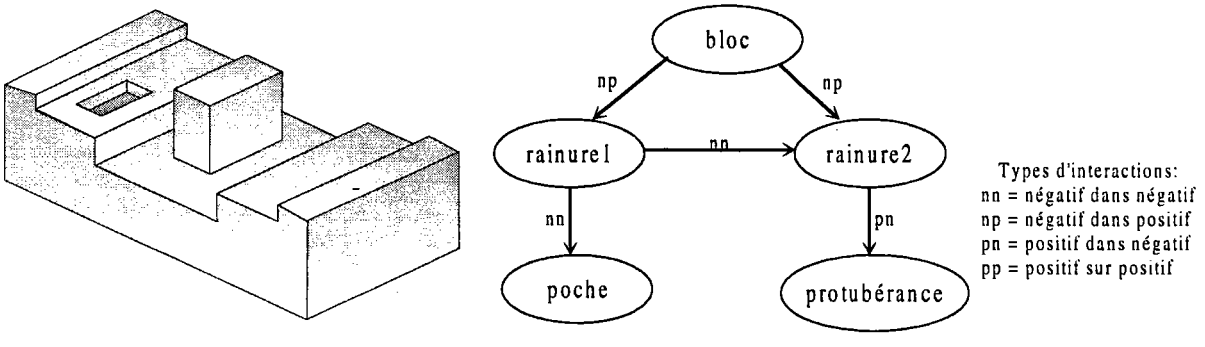


Figure 1.14 : La représentation FAG [GoT 91].

Au niveau le plus bas, chaque nœud du FAG correspond à un B-Rep et à une structure topologique de caractéristique (FTS = *Feature Topological Structure*). Le FTS est un pseudo-modèle dans lequel les caractéristiques de forme sont représentées par des volumes. Dans le B-Rep, par contre, on ne trouve que les faces des caractéristiques qui sont présentes dans l'évaluation finale de la pièce. La création des deux modèles topologiques est faite simultanément. Des liens de correspondance sont maintenus entre les entités du FTS et les entités du B-Rep. Le FTS a pour rôle de mémoriser toute l'information topologique à propos de chaque caractéristique de forme volumique.

Cette fois, les relations entre caractéristiques ne sont pas les opérations booléennes du CSG comme dans les deux approches précédentes, mais des relations d'adjacence et des interactions entre caractéristiques (dans, sur). À nouveau, une structure (le FTS) permet de conserver le

volume exact correspondant aux caractéristiques. L'intérêt de stocker le volume sera discuté dans la chapitre 2.

Nous retiendrons la nécessité de conserver un historique avec les relations entre les caractéristiques (relations portant - porté, relations spatiales et adjacences), en utilisant par exemple un graphe. De plus, il faut une représentation évaluée pour laquelle un B-Rep convient bien. Il faut également des liens entre les composants du graphe (caractéristiques) et les entités du B-Rep.

3.2.4. Liens entre le modèle de caractéristiques et le modèle géométrique

L'une des conclusions du paragraphe précédent est la nécessité de liens entre les couches historique et évaluée du modèle pour assurer leur cohérence. Ce paragraphe constitue une vue plus formelle de ces liens, déjà présentés dans les différentes approches des paragraphes précédents. Nous nous appuyons essentiellement sur les résultats de Shah et Mäntylä [ShM 95], qui discutent du type de lien selon plusieurs critères :

- la proximité du lien : statique (par un fichier intermédiaire) ou dynamique (par appel de procédures) ;
- le flot d'informations : dans un sens (caractéristiques vers géométrie ou vice versa) ou dans les deux sens ;
- l'opération de modification incrémentale ou non incrémentale : les changements peuvent être propagés à la représentation géométrique de façon incrémentale, en ajoutant, supprimant ou modifiant les éléments géométriques. Dans la solution non incrémentale, le modèle géométrique doit être réévalué dans sa totalité après chaque changement du modèle de caractéristiques ;
- le lieu du contrôle : détermine lequel des deux modèles est considéré comme le programme principal.

Les différents liens possibles sont détaillés ci-dessous, toujours selon [ShM 95].

↳ Lien statique unidirectionnel, caractéristiques vers géométrie : le concepteur travaille avec le modèle à base de caractéristiques et un accès au modèle géométrique est généré. Avec ce type de lien, des changements dans le modèle à base de caractéristiques entraînent une réévaluation complète du modèle géométrique. De plus, les fonctionnalités sont limitées : par exemple, la validation des caractéristiques ne peut pas utiliser les avantages du modeleur géométrique, utiles dans les règles de validation. Ce type de modèle est également limité dans la maintenance des

relations entre caractéristiques. Les avantages incluent la possibilité d'utiliser un format de fichier neutre tel que IGES pour réaliser la communication entre le modèle à base de caractéristiques et le modèle géométrique, ce qui rend le modèle de caractéristiques quasiment indépendant du modèle géométrique (donc plus portable).

↳ Lien statique unidirectionnel, géométrie vers caractéristiques : l'utilisateur interagit avec le modèle géométrique et ceci génère des accès vers le modèle basé sur les caractéristiques. Il est possible d'étendre le modèle géométrique avec des commandes qui génèrent des informations suffisantes pour créer des caractéristiques. Cependant, ce type de lien est rarement utilisé en pratique.

↳ Lien dynamique bidirectionnel, caractéristiques vers géométrie : l'utilisateur opère sur le modèle de caractéristiques qui génère le modèle géométrique directement par l'appel de procédures ou par une certaine forme de communication interprocessus. Le modèle géométrique possède une interface assez complète pour l'appel direct de procédures de création. La validation des caractéristiques est facilitée par les opérations booléennes. Ce type de liens simplifie également la gestion de relations entre caractéristiques.

↳ Lien dynamique bidirectionnel, géométrie vers caractéristiques : l'utilisateur interagit avec le modèle géométrique puis le modèle à base de caractéristiques est généré. Les informations localisées dans la représentation par les caractéristiques doivent être accessibles et utilisables par le modèle géométrique. Celui-ci doit supporter une architecture ouverte et extensible, qui permette de réaliser toutes les fonctionnalités de la modélisation par les caractéristiques. L'avantage est l'interface utilisateur uniforme qui couvre à la fois la modélisation géométrique et la modélisation basée sur les caractéristiques, ce qui est très intéressant dans la prise en compte de relations entre caractéristiques ou le dimensionnement et tolérancement des caractéristiques.

Le type de liens qui paraît le plus prometteur est un lien dynamique bidirectionnel, orienté des caractéristiques vers la géométrie. En effet, un lien dynamique par appel de procédures est plus immédiat donc plus efficace qu'un lien statique, qui passe par un fichier intermédiaire. Il paraît également souhaitable que le lieu du contrôle soit le modèle à base de caractéristiques, car il est d'un niveau sémantique plus élevé. Le sens du flux d'informations doit être bidirectionnel pour assurer à tout moment la cohérence entre les deux modèles et pour que chacun puisse bénéficier des avantages de l'autre. De plus, il permet une gestion incrémentale des modifications, dans la mesure où, les liens étant bidirectionnels, il est aisé de retrouver toutes les conséquences de la modification.

3.2.5. *Synthèse*

Plusieurs solutions sont donc possibles pour mettre en œuvre une structure de données capable de gérer les caractéristiques. Il ressort cependant de cette étude qu'un modèle géométrique est vraisemblablement toujours utile, voire indispensable, parallèlement au modèle à base de caractéristiques. Un bon compromis est de maintenir simultanément deux modèles : l'un permettant de conserver les informations de haut niveau (paramètres des caractéristiques, relations entre elles, contraintes, validité...) et l'autre les informations de plus bas niveau, toujours utiles (géométriques et topologiques). Des liens entre ces deux modèles sont indispensables pour assurer la cohérence permanente du système global. Le modèle géométrique sera un B-Rep plutôt qu'un CSG, puisque le principal intérêt du CSG, c'est-à-dire la conservation de l'historique, est réalisé par le modèle à base de caractéristiques. Les choix plus précis concernant le modèle à base de caractéristiques, ainsi que les liens avec le modèle géométrique, seront présentés dans le chapitre 2.

L'aptitude des différents systèmes étudiés à réaliser certains traitements sur les caractéristiques sera discutée dans le paragraphe 3.4. Le paragraphe ci-dessous explique comment les caractéristiques sont utilisables pour créer un modèle de conception. Elles sont souvent mémorisées dans une bibliothèque, tout au moins les caractéristiques de forme. Cette base de données peut être organisée de façon particulière, par exemple hiérarchique.

3.3. La bibliothèque de caractéristiques

L'intérêt de mémoriser la définition générique des caractéristiques dans une bibliothèque est de conserver un certain nombre d'informations à propos des caractéristiques, sans que ces données ne soient perdues au moment de l'instanciation. Pour cela, il faut qu'elles puissent être stockées dans une base de données, sous une forme générique, avec toutes les informations nécessaires, c'est-à-dire les données propres à chaque type de caractéristiques et les méthodes associées qui permettent leur utilisation dans le modèle de conception (instanciation, création de la forme, vérification de la validité...). La bibliothèque de caractéristiques prédéfinies est généralement extensible avec de nouvelles caractéristiques définies par l'utilisateur lui-même, selon ses besoins et sa spécialité.

La définition de nouvelles caractéristiques peut se faire de trois manières différentes :

- en combinant des caractéristiques prédéfinies simples ;

- en précisant des caractéristiques prédéfinies, par des contraintes par exemple. La queue d'aronde peut être considérée comme une précision de la rainure à trois faces ;
- en définissant une nouvelle forme, de nouveaux paramètres et de nouvelles informations associées, sans se baser sur les caractéristiques prédéfinies de la bibliothèque.

3.3.1. Existence d'un langage de définition

Pour définir de nouvelles caractéristiques, divers outils peuvent s'avérer nécessaires, comme un dialogue puissant ou encore un langage de définition. D'après [ShR 88a] en effet, le mécanisme de description des caractéristiques doit être extrêmement flexible pour permettre aux concepteurs de définir des caractéristiques de toute forme et à tout niveau, selon leurs besoins. Pour pouvoir utiliser ces représentations de caractéristiques de façon répétitive, les propriétés génériques doivent être mémorisées dans une bibliothèque de telle façon que les caractéristiques puissent être instanciées. Le système doit fournir la possibilité de définir et d'instancier des caractéristiques de forme, de précision et de matériau individuellement et d'indiquer des relations entre caractéristiques de forme et des relations intra-caractéristiques. Des langages interprétés ont été imaginés par [ShR 88a] pour écrire des règles et des expressions qui constituent en partie la définition des caractéristiques. D'une part, ces règles et expressions servent à créer un réseau d'héritage qui permet de déterminer les valeurs des paramètres hérités, calculés en fonction des paramètres du parent. D'autre part, elles permettent d'établir des règles cognitives qui garantissent la validité de l'utilisation des caractéristiques. L'ajout d'une caractéristique à la bibliothèque consiste donc à spécifier quels sont les paramètres donnés par l'utilisateur et quels sont ceux calculés en fonction du volume porteur et à donner une représentation géométrique et un ensemble de règles. Les contraintes de la caractéristique sont exprimées au moyen d'expressions dans lesquelles les variables sont remplacées par les paramètres (exemple : position-radiale * cos(angle)), il en est de même pour les règles de consistance (rayon-trou < rayon-père).

L'extension de la bibliothèque de caractéristiques dans le système IMPPACT [KrU 92] est réalisée avec un langage de description PDGL (*Part Design Graph Language*). Ce langage, détaillé dans [KRK 91], est utilisé pour une description procédurale et structurée des caractéristiques. Le système inclut un interpréteur PDGL pour créer les modèles à base de caractéristiques.

La définition des caractéristiques dans le système EXTDesign [LaM 93] est réalisée à l'aide d'un langage formel de définition de caractéristiques, qui facilite l'addition de nouvelles caractéristiques et l'édition d'anciennes. La description de la géométrie est réalisée par des opérations de modélisation géométrique classiques ; quand elles sont évaluées, un modèle géométrique de la pièce est généré.

De même [DZL 93] proposent un langage de définition de forme (SDL = *Shape Definition Language*) pour réaliser le dessin de la forme 2D, qui génère par extrusion un objet 3D. Le concepteur peut écrire la définition de la forme directement dans le langage ou bien dessiner la forme 2D interactivement ; le dessin est alors automatiquement converti dans une représentation SDL par le système.

Cette dernière approche évite la programmation et l'apprentissage d'un langage ce qui est un atout indéniable pour le système. L'intervention d'un programmeur n'est donc plus une obligation pour enrichir la bibliothèque de caractéristiques. Cependant, nous pouvons nous demander si une telle solution a la même puissance de définition que les autres, puisqu'elle se limite, semble-t-il, à une description de la forme uniquement. En particulier, il semble difficile de définir de façon précise des comportements, dans les trois dernières méthodes d'extension proposées.

3.3.2. Définition de nouvelles caractéristiques par l'utilisateur

Dans [MFG 94], deux types de caractéristiques de forme sont distinguées : les caractéristiques de conception prédéfinies, qui sont toujours disponibles et qui représentent la partie fixe de la bibliothèque, et les caractéristiques définies par l'utilisateur, qui complètent l'ensemble précédent et entraînent une gestion dynamique de la bibliothèque.

Chaque caractéristique prédéfinie (trou, poche, rainure) est décrite par :

- un nom (par exemple trou cylindrique) ;
- un identificateur (par exemple un numéro) ;
- un type de caractéristique (par exemple dépression) ;
- un type de représentation (explicite ou implicite). La définition de ces termes n'est pas précisée, on supposera qu'une représentation explicite est une description en terme de faces, arêtes, sommets, alors qu'une représentation implicite ne contient pas cette description ;

- une liste de paramètres (rayon R, profondeur P, position C...) ;
- des contraintes sur ces paramètres (R, P > 0) ;
- un ensemble d'attributs dépendant de l'application (état de surface) ;
- un ensemble de procédures nécessaires pour la création et la manipulation des caractéristiques (créer, supprimer, valider, modifier...).

Les caractéristiques définies par l'utilisateur peuvent être créées au moyen du modeleur basé sur les caractéristiques ou du modeleur géométrique, selon leur complexité. Le modeleur de caractéristiques est utilisé pour décrire des caractéristiques obtenues par composition de caractéristiques de conception prédéfinies. Dans le cas où les caractéristiques ont une forme complexe, l'utilisateur peut créer ses propres caractéristiques avec le modeleur géométrique, en concevant leur volume. L'information décrivant ce type de caractéristiques n'est pas aussi complète que celle des caractéristiques prédéfinies ; les représentations implicites ne sont pas considérées et les outils de manipulation ne peuvent pas être fournis. L'utilisateur peut seulement associer un type, un identificateur, une position et un ensemble d'attributs technologiques à la pièce.

Pour mettre les caractéristiques définies par l'utilisateur au même niveau que les caractéristiques prédéfinies, une solution est de définir les comportements de façon automatique. Ceci est cependant un problème complexe, qui sera abordé plus en détail dans le chapitre 3.

De nombreux autres systèmes autorisent également l'utilisateur à définir ses propres caractéristiques [CuD 88, DLN 90, ShR 88b...]. En particulier, le système de [ShR 88b] fonctionne en deux modes :

- le mode *modeling* qui permet d'instancier des caractéristiques génériques, d'établir des relations entre les instances et de créer des liens avec le modeleur géométrique ;
- le mode *set-up* pour la définition de nouvelles caractéristiques génériques.

Pour [CMV 91a], les caractéristiques génériques sont créées dans la bibliothèque en utilisant les étapes suivantes : création de la forme (souvent par extrusion), spécification des dimensions, spécification des types de dimensions (données par l'utilisateur à l'instanciation ou déduites de la caractéristique porteuse), spécification d'un repère local (origine et axes) et création d'une classe dans la hiérarchie. La création d'une classe suggère une approche orientée objets, qui est bien adaptée à la notion de hiérarchie. [YaW 92] proposent également un système orienté objets.

Pour [ShL 93], les caractéristiques de forme généralement utilisées sont déterminées par cinq constituants principaux (composant solide, repère local, taille, position et contraintes) et mémorisées dans une bibliothèque de caractéristiques. Les étapes de la création semblent les mêmes que celles de [CMV 91a]. Une dernière étape est toutefois ajoutée : la spécification de contraintes, exprimées par des chaînes de dimensions linéaires.

Enfin dans le système FSMT [DZL 93], une caractéristique est une forme paramétrée qui consiste en une description des attributs géométriques et une description des méthodes de transformation orientées application, qui seront détaillées au paragraphe 3.4.2. La géométrie d'une caractéristique 3D est déterminée par l'extrusion généralisée d'un profil 2D le long d'un guide. Le profil est d'abord décrit puis les dimensions et les contraintes géométriques sont déterminées et finalement le motif d'extrusion (prisme, révolution, boîte, surface évolutive, solide variant) est spécifié.

D'après [SSJ 94], le module de définition interactive de caractéristiques doit permettre :

- la définition interactive de la caractéristique de forme, avec aussi peu de programmation que possible ;
- la définition de la géométrie et d'éléments non géométriques tels que les contraintes, les tolérances, le matériau... ;
- la définition d'un comportement associé à la caractéristique ;
- une mémorisation de la caractéristique sous une forme qui facilite sa détection en cas d'apparition fortuite ;
- la conversion de la description de la caractéristique dans un format standard, tel que STEP par exemple.

Pour cela, la notion de graphe conceptuel est introduite. Une définition formelle est donnée par Sowa [Sow 84]. Un graphe conceptuel est un graphe biparti, connexe et fini, qui a les propriétés suivantes :

- les nœuds sont de deux types : les concepts (représentés par des boîtes dans la figure 1.15) et les relations conceptuelles (représentées par des cercles) ;

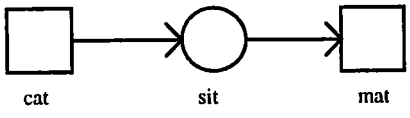


Figure 1.15 : Exemple de graphe conceptuel [Sow 84].

- chaque relation conceptuelle possède un ou plusieurs arcs liés à des concepts ;

- si une relation a *n* arcs, elle est dite n-aire ;
- un concept peut former un graphe conceptuel, mais tous les arcs d'une relation conceptuelle doivent obligatoirement être reliés à un concept.

La définition des graphes conceptuels a été adaptée à son utilisation dans FROOM par rapport à la définition originale (un exemple est donné par la figure 1.16) [SSK 94] :

- les concepts sont des assemblages, des composants, des caractéristiques ou des faces ;
- les relations utilisées dans FROOM sont uniquement unaires, binaires ou ternaires ;
- les arcs sont la plupart du temps non orientés car les relations entre concepts dénotent souvent une connexion dans les deux sens.

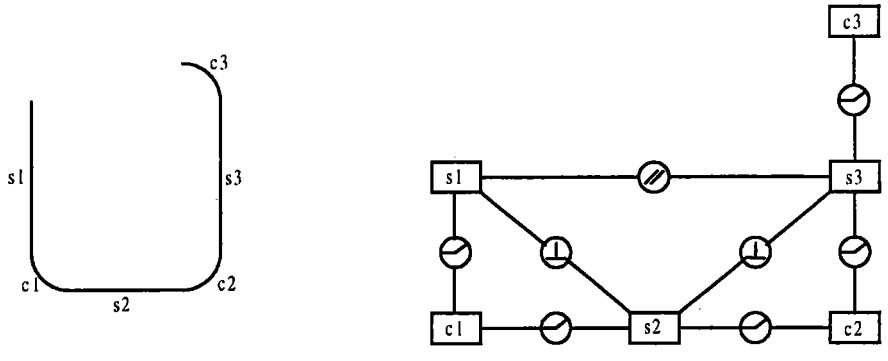


Figure 1.16 : Exemple de graphe conceptuel de FROOM [SSJ 94].

Dans l'approche de [SSJ 94], trois moyens de définitions de nouvelles caractéristiques sont envisageables :

- définir une nouvelle caractéristique par esquisse : le système détecte automatiquement les informations géométriques, topologiques et les contraintes, et le graphe conceptuel correspondant est généré automatiquement ;
- définir ou modifier une nouvelle caractéristique par édition du graphe conceptuel : l'opérateur agit sur le graphe et la géométrie est déduite automatiquement ;
- définir une caractéristique en désignant interactivement les entités d'un solide qui en font partie : le graphe est alors déduit automatiquement.

En résumé, les étapes de la création de nouvelles caractéristiques se font de différentes façons selon les systèmes, mais suivent toujours le même principe de définir la forme d'abord, puis de spécifier divers paramètres sur cette forme (dimensions, repère local...) et enfin éventuellement d'y attacher des contraintes.

3.3.3. *Organisation de la bibliothèque*

La réalisation des fonctions d'ajout de caractéristiques impose la connaissance de la structure de la bibliothèque. Nous précisons dans la suite les structures proposées dans la littérature, en remarquant qu'une organisation hiérarchique est quasiment systématique.

La hiérarchie de classes proposée par [CMV 91a] permet d'établir une relation de généralisation entre chaque super-classe et ses sous-classes, où les attributs des classes de niveau le plus spécialisé héritent des classes du niveau le plus général. La hiérarchie est représentée par la figure 1.17. Les caractéristiques prises en compte sont de deux types : une caractéristique de forme est utilisée pour décrire la forme d'une pièce alors qu'une caractéristique de fabrication (*process feature*) est une entité géométrique qui est significative pour une certaine application (surface usinée...). Les caractéristiques de forme sont soit primaires si elles représentent la forme principale de la pièce, soit secondaires si elle modifient la forme de la pièce. Une caractéristique secondaire est soit négative (dépression), soit positive (protubérance), selon qu'elle enlève ou ajoute de la matière à la pièce.

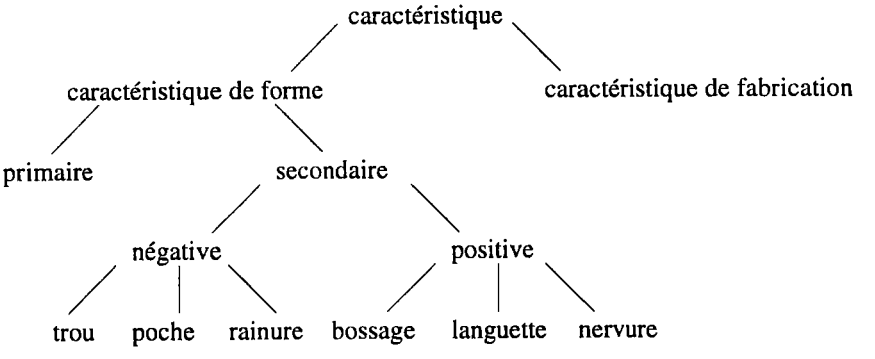


Figure 1.17 : Hiérarchie des classes de caractéristiques [CMV 91a].

L'originalité de cette classification est la distinction entre caractéristiques de forme et caractéristiques de fabrication. Elles sont au même niveau dans la hiérarchie alors qu'elles n'ont pas le même degré d'abstraction puisqu'une caractéristique de fabrication peut être une surface, donc un composant (une face) d'une caractéristique de conception telle que la rainure par exemple. Cette séparation n'est pas justifiée dans l'article, mais s'explique peut-être par l'importance des renseignements qu'une caractéristique apporte respectivement en conception et en fabrication.

Dans le système EXTDesign [LaM 93], les classes de caractéristiques sont également organisées en une taxinomie, qui permet l'héritage de l'information commune. En effet, la taxinomie est formée en utilisant la relation *est un*. La racine de la taxinomie est la classe

caractéristique ; les sous-classes immédiates sont les blocs, les transitions, les caractéristiques de base, les surfaces et les conteneurs.

En ce qui concerne l'organisation de la bibliothèque dans FSMT [DZL 93], les composants sont divisés en cinq catégories : arbre, disque, surface gauche, boîte et équerre. Des caractéristiques appartiennent à une même catégorie si leur fonction et leur géométrie sont similaires. Elles peuvent également être classées par rapport à leur procédé de fabrication. Des caractéristiques de catégories différentes peuvent donc appartenir à la même famille de caractéristiques de fabrication. D'un point de vue fabrication, certaines formes sont subordonnées à d'autres et sont donc traitées comme des sous-caractéristiques. Par exemple, les vis et les molettes ont une surface cylindrique et sont considérées comme sous-caractéristiques de la surface cylindrique. L'utilisateur peut donc tirer parti de cette structure hiérarchique pour gérer les processus de fabrication compliqués.

Pour [ShM 95], une classification des caractéristiques est utile pour les raisons suivantes :

- si les classes de caractéristiques peuvent être triées en familles avec des propriétés spécifiques, les mécanismes de généralisation peuvent être développés au sein de chaque famille, au lieu de méthodes spécialisées pour toutes les classes de caractéristiques ;
- les familles de caractéristiques peuvent utiliser une terminologie commune au travers des applications, qui peut même évoluer vers une interface plus universelle et faciliter ainsi l'extension de l'ensemble des caractéristiques disponibles ;
- des familles universelles peuvent être utiles pour développer des standards d'échange de données.

Nous remarquons que le premier point correspond à l'héritage et le deuxième au polymorphisme. Une approche orientée objets est donc souhaitable pour l'organisation de la bibliothèque de caractéristiques. Les taxinomies peuvent être basées sur les catégories de produit, sur les applications ou sur la forme des caractéristiques, comme dans le tableau ci-dessous :

Type de produit	Application	Forme
Tôlerie	Conception	Prismatique
Panneaux composés	Analyse par éléments finis	Rotationnelles
Caractéristiques usinées	Préparation de la fabrication	Plates
Moulages par injection	Inspection	Section uniforme

Aucune taxinomie universelle n'existe actuellement. Certaines classifications sont uniquement basées sur la forme, comme Part 48 de STEP (*Standard of Exchange of Product Data*) qui considère les caractéristiques en trois types : volumes, transitions et caractéristiques à motif [Dun 92]. Les caractéristiques volumiques sont elles-mêmes classées en six types de base : passage, dépressions, vides (complètement dans la matière), protubérances, connecteurs et indépendants. Une classification très élaborée a été développée pour CAM-I [BGS 85] pour la préparation de la fabrication. Certaines taxinomies enfin ne couvrent qu'une sorte de produit, par exemple en tôlerie [ShB 86].

3.3.4. Synthèse

Il ressort de cette étude de l'existence de bibliothèques de caractéristiques qu'un système doit fournir un moyen de mémoriser les caractéristiques génériques dans une structure de données adéquate. L'intérêt d'organiser cette base de données en une taxinomie permettant l'héritage de certaines propriétés a été mis en évidence. Une approche orientée objet est donc adaptée à un tel agencement.

La structure des caractéristiques génériques doit également permettre à l'utilisateur d'enrichir la bibliothèque en définissant ses propres caractéristiques, de la façon la plus conviviale possible, éventuellement grâce à un langage de définition. L'organisation hiérarchique présente un avantage incontestable pour cela puisque lorsqu'une nouvelle caractéristique est ajoutée dans une famille, la caractéristique se trouvant juste au dessus dans la hiérarchie donne connaissance de toutes les propriétés à spécifier ; la définition de la nouvelle caractéristique peut donc être guidée. Ainsi, nous proposerons un squelette de définition d'une nouvelle caractéristique, que l'utilisateur n'aura qu'à compléter.

3.4. Les autres traitements relatifs aux caractéristiques

Nous avons vu différentes solutions proposées pour mémoriser les caractéristiques dans une bibliothèque et éventuellement pour enrichir cette bibliothèque par des caractéristiques définies par l'utilisateur. Des traitements typiquement orientés caractéristiques sont parfois prévus dans les systèmes de conception ; ils exploitent la richesse sémantique des caractéristiques. Nous allons voir d'abord ceux qui permettent la détection automatique de caractéristiques de forme, apparues au cours de la conception. Puis nous étudierons les systèmes qui prévoient la transformation de caractéristiques de conception en caractéristiques d'application. Le but est d'arriver à une déduction automatique d'un modèle d'application.

3.4.1. Extraction de caractéristiques de forme

La nécessité d'une reconnaissance de caractéristiques par les applications demeure dans la mesure où les caractéristiques sont spécifiques aux applications et où les concepteurs ne pensent pas forcément toujours en termes de caractéristiques. De plus, un modèle basé sur les caractéristiques d'une pièce peut être modifié par une intervention sur le modèle géométrique, utilisant par exemple des opérations booléennes. Ceci signifie que le concepteur peut réaliser des modifications non seulement sur le modèle à base de caractéristiques, mais aussi sur le modèle géométrique, ces changements étant propagés dans l'autre modèle.

Le système de [MFG 94] permet l'extraction de caractéristiques. La méthode utilisée pour mener à bien cette opération est la suivante : le module de reconnaissance de caractéristiques de forme identifie des caractéristiques en analysant la représentation par les limites (B-Rep) de l'objet, les classifie en protubérances ou dépressions et les représente comme des caractéristiques volumiques dans le modèle intermédiaire (décrit au paragraphe 3.2.2). Le processus débute par la classification des arêtes et contours en concaves ou convexes. Ensuite, les ensembles de faces appartenant aux caractéristiques sont identifiés. Ces ensembles de faces sont alors complétés par des faces de fermeture pour obtenir des représentations volumiques positives ou négatives. Finalement, ces volumes sont intégrés au modèle intermédiaire. Le module de reconnaissance peut être utilisé pour détecter soit toutes les caractéristiques qui ne sont pas créées par le modeleur de caractéristiques et qui affectent l'objet, soit les caractéristiques résultant de l'interaction de deux autres caractéristiques.

Le module de reconnaissance de caractéristiques inclus dans EXTDesign [LaM 93] est basé sur une technique hybride dans laquelle plusieurs méthodes de reconnaissance peuvent être

utilisées ; la méthode adéquate est spécifiée dans la définition des caractéristiques, selon la catégorie à laquelle elles appartiennent. Les caractéristiques sont en effet classées en caractéristiques de base, simple, conteneur et combinaison. Pour accélérer la recherche, elles sont organisées dans un arbre : au premier niveau, elles sont classées suivant le nombre de nœuds, au deuxième, suivant le nombre d'arcs, au troisième, suivant la topologie du nœud et les feuilles représentent la topologie de la caractéristique. La recherche est alors accomplie en quatre étapes :

1) pré-traitement : le graphe d'adjacence du modèle géométrique est généré ;
2) recherche de caractéristiques de base : le graphe entier est apparié avec les représentations des caractéristiques de base. Les nœuds correspondant à ces dernières sont supprimés, ce qui divise le graphe initial en plusieurs graphes partiels ;
3) recherche de caractéristiques simples et conteneurs dans les graphes partiels par appariement ;
4) recherche de combinaison de caractéristiques dans les graphes partiels qui ne correspondent à aucune caractéristique simple.

Quand le processus de reconnaissance trouve un graphe partiel ou sous-graphe du graphe complet qui correspond à un type de caractéristique, il crée une structure de données qui relie la caractéristique à une portion du modèle géométrique (création d'une caractéristique générique). Après la reconnaissance, les caractéristiques relatives aux structures de données créées sont instanciées. Quand le modèle géométrique est modifié, le modèle à base de caractéristiques doit être mis à jour. Le contexte de reconnaissance précédent est sauvegardé et peut être utilisé quand la reconnaissance suivante est réalisée (après la modification) ; c'est pourquoi on parle de reconnaissance incrémentale. Le module de reconnaissance compare le nouveau graphe de recherche, correspondant à la représentation du modèle géométrique courant de la pièce, à l'ancien graphe. Si de nouvelles entités ont été ajoutées ou supprimées, elles sont détectées automatiquement et la recherche ne s'effectue que sur la portion du graphe qui a subi la modification.

3.4.2. Transformation en caractéristiques d'application

La façon dont une pièce est vue en termes de caractéristiques varie d'applications en applications (conception, fabrication, inspection). La dérivation d'un modèle à base de caractéristiques spécifique à une tâche dans un autre modèle à base de caractéristiques est connue sous le terme de *mapping* de caractéristiques, de transformation de caractéristiques ou de transmutation de caractéristiques. Il n'existe pas de procédé standard pour la transformation car

les relations entre modèles de caractéristiques varient. Quelques observations générales sont cependant faites dans [ShM 95] :

- les modèles à base de caractéristiques diffèrent dans leur niveau de détails : le même objet peut être traité par un modèle tridimensionnel en conception mais par un modèle bidimensionnel en éléments finis ;
- certains modèles à base de caractéristiques ne contiennent pas la description complète de la géométrie de la pièce : les caractéristiques d'inspection ne peuvent généralement pas être utilisées pour générer la géométrie de toute la pièce ;
- la même caractéristique peut être paramétrée de différentes façons dans différentes applications ;
- souvent, les caractéristiques d'une application sont obtenues à partir des caractéristiques d'une autre en regroupant différemment les entités du modèle géométrique (faces, arêtes, sommets) ;
- une application peut voir des portions solides d'une pièce comme des caractéristiques (moulage par injection) alors qu'une autre voit les absences de matière comme des caractéristiques (usinage) ;
- beaucoup de caractéristiques ont des attributs spécifiques aux applications, qui ne sont pas intéressants pour d'autres applications : les tolérances géométriques sont utiles pour la conception et la fabrication mais pas pour l'analyse de résistance à l'effort ;
- certains modèles de caractéristiques utilisent plus d'un modèle géométrique : la préparation de la fabrication nécessite un modèle de la pièce, un modèle du brut et plusieurs modèles de fixation et d'outillage.

La transformation automatique d'un modèle à base de caractéristiques en un autre est souhaitable mais représente un large éventail de problèmes et très peu de travaux ont été réalisés dans ce sens. L'extraction de caractéristiques, qui a fait l'objet du paragraphe précédent, est sans doute bien utile pour aboutir à une transformation d'un modèle en un autre. C'est d'ailleurs ce que proposent [ReV 89] : la représentation CSG est convertie en caractéristiques de fabrication (ou autre application) par un module de reconnaissance.

Dans [SBH 88], un schéma de transformation de caractéristiques est fourni pour extraire et reformuler les données nécessaires aux applications. Les paramètres sont classés en trois groupes :

- paramètres disponibles explicitement dans la base de données de caractéristiques ; ils sont indépendants des autres données dans le modèle ;

- paramètres qui peuvent être calculés à partir de ceux qui sont explicites ; ils sont implicites mais invariants par rapport au modèle ;
- paramètres qui dépendent d'une relation entre caractéristiques, c'est-à-dire qui doivent être calculés avec des paramètres de type 1 et 2 de plusieurs caractéristiques ; ils sont dépendants du modèle.

Le mécanisme de transformation doit extraire les trois types d'information. Un des besoins pour créer des structures génériques est un langage interprété qui peut être utilisé pour écrire des procédures d'extraction spécifiques.

Dans [DZL 93], deux applications sont considérées : l'analyse et la fabrication. Dans le premier cas, un maillage 3D peut être généré de façon similaire à la génération de la caractéristique. Un maillage 2D est d'abord généré sur la forme 2D de la caractéristique et les éléments 3D sont obtenus par extrusion du maillage 2D. La facilité d'obtention d'un maillage est un avantage de l'extrusion généralisée. Pour la préparation à la fabrication et la programmation numérique, chaque approche d'extrusion (prismatique, révolution, boîte, surface évolutive, solide variant) peut être reliée à un type d'opération d'usinage (perçage, forage,...). Au dire des auteurs, cette particularité, avec la similarité entre le chemin de l'outil et la trajectoire d'extrusion, rend la programmation numérique plus facile.

Le module de conversion en caractéristiques d'application de [MFG 94] a pour principale fonction de transformer le modèle intermédiaire en un modèle basé sur les caractéristiques dépendant de l'application, en appliquant des règles dépendant du contexte. Ces règles ne sont pas exprimées par du code dans le système mais sont données par l'utilisateur via une technique d'apprentissage par l'exemple. Le système extrait automatiquement le graphe d'adjacence de caractéristiques de l'exemple et analyse le modèle de l'objet en cherchant des sous-graphes qui correspondent au graphe de l'exemple. L'utilisateur précise alors le type d'opération à réaliser sur chaque sous-graphe reconnu : une opération de regroupement (ou simplification) pour générer des caractéristiques composées (la complexité de la représentation sera donc réduite) ou une opération de raffinage (ou subdivision) pour les caractéristiques qui font partie de caractéristiques complexes.

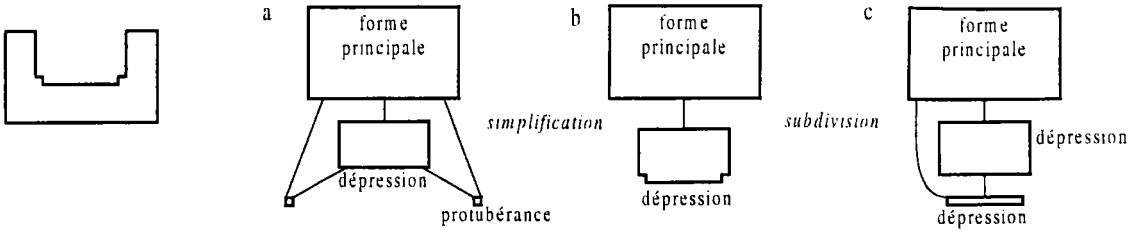


Figure 1.18 : Mise à jour du graphe de caractéristiques (a)
par opérations de simplification (b) et subdivision (c) [MFG 94].

Après ce processus de conversion, qui est basé sur la forme mais guidé par l'application, la description géométrique satisfait aux exigences de l'application et le modèle est complété avec une information spécifique pour devenir un modèle basé sur les caractéristiques d'application. L'exemple de la figure 1.18 montre que la représentation peut être adaptée à une tâche d'assemblage par opération de simplification sur le graphe (b) ou à une application d'usinage par opération de subdivision (c).

Comme les caractéristiques dépendent du type de produit, de l'application et du niveau d'abstraction, un espace de caractéristiques est présenté dans [ShM 95] comme étant un domaine spécifié par ces trois attributs. La totalité des informations relatives à toutes les sortes de produit, dans tous ses aspects, pendant tout son cycle de vie, et pour toute application détermine un domaine appelé hyper-espace de caractéristiques. Les régions de cet espace peuvent se chevaucher et correspondent à une sémantique identique. Une autre relation entre espaces est celle composée de différentes variations du même élément (3 languettes ⇔ 2 rainures). Lorsque deux sous-espaces sont totalement disjoints, il n'y a pas de transformation possible. Plusieurs situations similaires en transformation de modèles de caractéristiques entre espaces conjugués ont été reconnues et définies comme des classes de *mapping* :

- *mapping* un-à-un (trous, poches, rainures, nervures) ;
- reparamétrisation variante : quand les caractéristiques de conception correspondant aux caractéristiques de préparation de la fabrication ont été paramétrées ou positionnées de façon différente ;
- *mapping* de spécialisation : quand les niveaux de spécialisation de deux ensembles de caractéristiques sont différents ;
- *mapping* de généralisation : plusieurs caractéristiques sémantiquement différentes sont transformées en une unique entité, plus abstraite ;
- agrégation discrète : m caractéristiques (m>1) sont combinées en une unique caractéristique ;

- décomposition discrète : une caractéristique est décomposée en *m* caractéristiques (*m*>1) ;
- *mapping* conjugué : les caractéristiques sont décomposées en leurs faces constituantes puis les faces sont regroupées pour former de nouvelles caractéristiques à partir de portions de plusieurs caractéristiques différentes.

Le problème le plus courant est la transformation des caractéristiques de conception en caractéristiques de fabrication. Les techniques suivantes sont actuellement élaborées, mais encore expérimentales :

- méthodes heuristiques [CJC 93, WCJ 93, Hsi 90] : utilisation de règles de transformation pré-spécifiées, mais elles sont difficiles à énumérer (explosion combinatoire) et ne s'appliquent pas à des caractéristiques définies par l'utilisateur (à cause de leurs connaissances cachées) ;
- structure de niveau intermédiaire [SrC 92, Gad 94, SBH 88] : les caractéristiques de chaque domaine sont utilisées pour regrouper certaines entités et relations topologiques et géométriques, car les relations géométriques intéressantes ne sont pas les mêmes suivant les applications. La géométrie intermédiaire consiste en relations géométriques abstraites entre les faces et les axes des caractéristiques ;
- *mapping* par décomposition cellulaire [VaR 94, KaN 92, GNR 94, SSS 94, BiT 93, GoT 91] : en *mapping* conjugué, les volumes sont combinés en différents groupes pour créer des caractéristiques pour différentes applications. C'est pourquoi, un modèle qui consiste en une décomposition cellulaire peut faciliter le *mapping* de caractéristiques. Des méthodes basées sur la décomposition en cellules volumiques ont été proposées pour extraire les caractéristiques d'usinage ; elles comprennent quatre phases principales : déterminer le volume à supprimer par usinage comme étant le brut moins la pièce finie ; partitionner chaque volume de matière à supprimer en sous-volumes élémentaires ; combiner les volumes élémentaires en caractéristiques usinables ; faire correspondre les caractéristiques usinables à des opérations d'usinage ;
- *mapping* basé sur les graphes [FGP 92, FPS 93] : dans les modèles géométriques basés sur les graphes, les caractéristiques sont des sous-graphes isomorphes à des sous-graphes stéréotypés. Le *mapping* de caractéristiques sur un modèle basé sur un graphe implique l'abandon d'une partie d'un graphe au profit d'une autre.

3.4.3. Synthèse

Le système de modélisation doit permettre les traitements sur les caractéristiques. Les caractéristiques pourraient ainsi disposer d'informations permettant leur reconnaissance automatique dans un modèle quelconque, ou encore leur transformation dans un modèle d'application.

Les principales méthodes d'extraction sont basées sur la représentation B-Rep (recherche des protubérances et dépressions) ou sur les graphes (appariements).

Les principales techniques de transformation du modèle de conception en modèles dédiés à la fabrication sont les méthodes heuristiques (avec des règles de transformation), l'utilisation d'un niveau intermédiaire (par regroupement d'entités et de relations topologiques et géométriques), la décomposition cellulaire (suivie de la combinaison des volumes élémentaires) et le *mapping* basé sur les graphes.

En plus de ces traitements, le système doit prendre en compte l'intention de conception (validité des caractéristiques, comportement, contraintes...). Le paragraphe suivant aborde la manière dont l'intention de conception peut être garantie par le biais des caractéristiques.

3.5. Validation des caractéristiques

Lorsque des caractéristiques sont créées, modifiées ou supprimées, il faut vérifier que le résultat de ces opérations est valide. Dans ce paragraphe, nous comparons donc les systèmes étudiés selon le critère "validation".

Les tests de validité à réaliser sont classés en trois catégories principales [ShM 95] :

- la compatibilité entre les caractéristiques mères et filles, c'est-à-dire la compatibilité entre les caractéristiques adjacentes et la compatibilité de type avec l'entité géométrique sur laquelle la caractéristique est placée ;
- les limites des dimensions, c'est-à-dire des restrictions sur les paramètres de taille, et les limites de localisation, c'est-à-dire des restrictions sur les paramètres de position de d'orientation ;
- les interactions entre caractéristiques qui altèrent soit la forme, soit la sémantique, par rapport à la définition générique. On peut citer plus particulièrement [Sha 91] :
    - les interactions qui rendent une caractéristique non fonctionnelle, par exemple, un trou placé trop près de la paroi de son porteur ;

- les caractéristiques obtenues par combinaison de plusieurs caractéristiques ;
- les paramètres de caractéristiques rendus obsolètes. Par exemple, si l'on crée une poche à l'endroit où se trouve un trou, la hauteur du trou devrait être mise à jour ;
- une topologie non standard, c'est-à-dire non conforme à la définition. Par exemple, lorsque deux rainures se croisent, l'une peut compter six faces alors que dans la définition générique elle n'en compte que trois ;
- une caractéristique détruite par suppression d'une caractéristique plus grande ;
- une caractéristique détruite par addition d'une caractéristique plus grande ;
- une caractéristique ouverte devenant fermée ;
- une caractéristique fermée devenant ouverte ;
- toute interaction involontaire suite à une modification.

Le but est de proposer un système dans lequel chaque caractéristique est une entité active, munie d'une capacité d'auto-validation et capable d'un contrôle local de son comportement. Selon [MCO 97], une telle caractéristique est décrite par trois ensembles de propriétés : technologiques, de forme, de validation. De plus, elle est munie d'un ensemble de méthodes de validation qui contrôlent ses propriétés géométriques et ses entités topologiques à l'instanciation et pendant l'évolution du modèle. Ces méthodes de validation permettent de réaliser les trois catégories de tests mises en évidence, c'est-à-dire les compatibilités entre caractéristiques, les limites des paramètres et les interférences avec d'autres caractéristiques. Chaque méthode est un ensemble de propriétés que les entités géométriques et topologiques doivent posséder.

Il faut donc que le modeleur à base de caractéristiques soit muni de méthodes de validation. Que ces méthodes soient attachées aux caractéristiques semble un atout.

3.6. Conception fonctionnelle

L'un des buts fondamentaux qu'un système de modélisation produit de CAO doit atteindre est le respect de ce que l'on appelle l'intention de conception ou le *design intent*. Dans les premiers systèmes de modélisation par les caractéristiques, la forme est définie d'abord, puis un comportement y est associé. Ce comportement permet à l'utilisateur de définir des contraintes qu'il désire voir respecter pendant toute la durée de vie du produit. Ainsi, la nature des caractéristiques et les contraintes associées suffisaient à maintenir l'intention de conception.

Un exemple d'une telle approche est donné par Nielsen *et al.* [NDZ 91]. Ils proposent de capturer et d'utiliser les intentions de l'utilisateur de la façon suivante : la forme des pièces est saisie d'abord comme étant une configuration de caractéristiques de forme primitives. Le système capture alors et retient les informations de connectivité, sous forme de contraintes géométriques. Les intentions du concepteur concernant le dimensionnement et le positionnement relatif (liens géométriques) sont également capturées. Les relations scalaires intervenant dans la conception sont exprimées par des valeurs cibles ou des intervalles, avec un degré d'importance associé (très important, important, souhaitable) permettant de régler les conflits entre plusieurs relations. Il peut cependant subsister des conflits, auquel cas c'est à l'utilisateur de décider de la manière dont ils doivent être résolus.

L'intention de conception est donc exprimée uniquement par un ensemble de contraintes géométriques et équationnelles. Cependant, les chercheurs ont constaté que l'homme de métier a plus souvent connaissance de la fonction de la pièce à concevoir que de sa forme physique exacte. Une nouvelle tendance consiste donc à représenter l'intention de conception par un ensemble de fonctions que la pièce doit remplir, puis à en déduire la forme. Cette nouvelle approche fait l'objet du paragraphe suivant.

3.6.1. Des fonctions vers la forme

Le procédé de conception d'une pièce peut être décomposé en plusieurs phases. Tous les auteurs [WSS 92, ShM 95, BDS 96...] ne sont toutefois pas unanimes au sujet de la décomposition qui peut être plus ou moins détaillée. Cependant, nous pouvons toujours faire la distinction entre ce que nous appellerons la conception préliminaire (ou conception fonctionnelle) et la conception détaillée.

La conception préliminaire permet de passer des spécifications initiales du produit à un ensemble de fonctions qu'il doit assurer. Elle consiste donc à décomposer la fonction générale en une hiérarchie de fonctions de plus en plus simples, jusqu'à obtenir un ensemble de fonctions élémentaires. Cette étape est souvent à la charge du concepteur uniquement et il n'est guère assisté par l'ordinateur pour la réaliser.

La conception détaillée consiste, dans un premier temps, à associer des formes (caractéristiques de forme par exemple) aux fonctions de base résultant de la première étape de la conception. Une fois cette association réalisée, il reste à dimensionner le modèle, à l'analyser... De façon générale, c'est dans cette seconde phase que le système de CFAO intervient, en

fournissant au concepteur des outils pour décrire les formes auxquelles la décomposition fonctionnelle l'a mené.

Si on limite l'intention de conception à des contraintes essentiellement géométriques, comme nous le suggérions dans le paragraphe précédent, le système de CAO ne peut que proposer des solutions satisfaisant à ces contraintes. Ceci semble adapté à la génération de variantes conformes aux choix faits par l'opérateur au moment de traduire en formes physiques les fonctions isolées dans la conception préliminaire. Des travaux récents visent à confier davantage de responsabilités au système, en étendant son champ d'action à la conception préliminaire. Il s'agit donc d'assister le concepteur dès les premiers instants de la conception. Nous résumons dans la suite les différentes étapes, de la conception préliminaire aux premiers pas de la conception détaillée, en se basant sur l'algorithme de conception proposé par Kim et O'Grady [KOG 96]. Cet algorithme sert de modèle de représentation informatique.

La première étape consiste à passer des spécifications de la conception à un ensemble équivalent de requêtes fonctionnelles qui pourront être transformées en un ensemble de caractéristiques. Selon Tichkiewitch *et al.* [TCB 96], à l'instant initial de la conception, un premier cahier des charges donne les fonctions principales attendues pour le produit. Le métier de technologue consiste alors à faire l'inventaire des principes physiques permettant de répondre aux fonctions de base et à en choisir un ou plusieurs. À chaque principe retenu correspond au moins une solution qui elle-même peut faire appel à de nouvelles sous-fonctions. Les choix successifs de principes, de solutions associées, de nouvelles fonctions induites, amènent à réaliser une arborescence de fonctions et de structures correspondantes. Plusieurs fonctions pouvant être satisfaites par la même structure, ou une fonction pouvant être couverte par plusieurs éléments de structure, l'arborescence se traduit très vite en un graphe fonctionnel-structurel. L'aboutissement de ce travail de conception initiale doit être un ensemble de solutions techniques répondant parfaitement aux fonctions de chaque pièce composant le système et en particulier aux interactions entre les pièces.

Une fois que la hiérarchie de fonctions est établie, il faut l'analyser pour vérifier qu'il n'y a pas d'oubli (toutes les fonctions sont représentées) ni de redondance (plusieurs fonctions réalisent la même requête). Le système SYSFUND, présenté dans [TUY 93] et intégré à un modeleur à base de caractéristiques dans [RMU 96], procure une structure pour représenter les concepts abstraits des fonctions et du comportement par un modèle informatique. Pour cela, SYSFUND fournit un outil de modélisation pour permettre au concepteur de construire un

modèle FBS (Function-Behaviour-Sate). Une fois le réseau de comportement construit, le concepteur peut réaliser une simulation du comportement avec le système de raisonnement qualitatif. Le modèle FBS initial est comparé au résultat de la simulation, ce qui permet de détecter si des requêtes manquent ou si des phénomènes créent des effets de bords indésirables.

SYSFUND assiste le concepteur dans la tâche de création de la structure produit grâce à une base de connaissances, en suggérant les pièces qui correspondent à des entités fonctionnelles. Cependant, c'est le concepteur lui-même ou un ingénieur expert qui doit collecter à l'avance la connaissance des caractéristiques physiques possibles et les rattacher aux fonctions. Le concepteur peut donc instancier les caractéristiques physiques et ainsi transformer la hiérarchie de fonctions en un réseau de comportements.

Une dernière vérification réalisée par SYSFUND est l'analyse des redondances fonctionnelles, qui permet d'utiliser des fonctions de pièces existantes de manière légèrement différente par rapport à la conception initiale. En effet, certaines fonctions permettent de réaliser une autre fonction, en même temps que leur fonction principale (celle pour laquelle elles ont été conçues).

La deuxième étape de l'algorithme de conception de [KOG 96] prépare la conception détaillée. L'ensemble de toutes les fonctions à satisfaire est initialisé avec les fonctions issues de l'étape précédente. L'ensemble de caractéristiques est vide.

La troisième étape de [KOG 96] permet de transformer les requêtes fonctionnelles en un ensemble de caractéristiques qui sont ajoutées au modèle de conception. C'est un processus itératif qui traite chaque requête l'une après l'autre. Chaque itération est décomposée en plusieurs parties.

Il faut d'abord trouver une correspondance entre toutes les exigences fonctionnelles et les caractéristiques. S'il en existe plusieurs, il faut choisir la meilleure. Le fait que les caractéristiques de conception soient modélisées de façon à intégrer la fonction et la forme est un atout pour cette recherche de correspondance.

Enfin, les caractéristiques de base sont ajoutées au modèle de conception.

Le système de Feng *et al.* [FHK 96] permet une formalisation des fonctions que les caractéristiques réalisent, en vue d'aider la transformation des fonctions en caractéristiques. Cette transformation ne semble pas réalisable automatiquement, mais une assistance pour un domaine de conception spécifique est tout de même possible, grâce à un système expert. Le concepteur est alors guidé dans la sélection d'une caractéristique par les relations entre requêtes et fonctions qui sont modélisées.

La fin d'une itération consiste à valider le modèle de conception créé, par vérification des conditions de validité des paramètres. En cas de succès, il est transformé en représentations secondaires, composées de caractéristiques spécifiques à des applications données, pour vérifier les contraintes d'ingénierie simultanée. Le modèle de conception est ensuite soumis à une vérification de validité fonctionnelle ; on s'assure que toutes les requêtes fonctionnelles satisfaites lors du traitement des requêtes précédentes le sont encore après l'ajout des nouvelles caractéristiques. Enfin, l'ensemble des requêtes à satisfaire est mis à jour par suppression de la requête qui vient d'être traitée. [FHK 96] proposent également une phase de validation.

La quatrième et dernière étape de l'algorithme de conception de [KOG 96] est la conception détaillée. Les attributs des caractéristiques qui n'ont pas été instanciés à l'étape précédente sont spécifiés, en prenant en compte des relations du modèle de caractéristiques et diverses contraintes d'ingénierie simultanée.

Il existe diverses limitations à l'algorithme de [KOG 96]. Une intervention manuelle est encore nécessaire à plusieurs étapes et beaucoup de travail de recherche doit être réalisé avant qu'une procédure générale pour la conception automatique puisse être produite. De plus, l'obtention d'une solution en un nombre fini d'opérations n'est pas garantie, dans deux cas précis : d'une part, lorsque aucune alternative ne permet d'obtenir un modèle valide et d'autre part, lorsque satisfaire une requête fonctionnelle invalide une ou plusieurs autres requêtes déjà satisfaites (il faut alors vérifier dans les spécifications que deux fonctions ne sont pas en conflit).

La conception descendante nécessite, selon Mäntylä, de pouvoir représenter l'information à plusieurs niveaux d'abstraction [Män 90]. Elle doit, en outre, permettre la modélisation des motivations des différents intervenants, en vue d'une remise en cause des choix possibles. Il ne s'agit plus de modéliser les choix du concepteur, mais son intention (les raisons qui le conduisent à exprimer ces choix). Enfin, le modèle doit supporter une géométrie abstraite, qui représente les parties de la conception où aucun engagement pour une solution détaillée n'a été pris. Il serait donc également souhaitable que le système soit capable d'habiller automatiquement la géométrie abstraite.

Ceci permettrait sans doute de générer des alternatives innovantes, donnant alors lieu à une conception réellement créative de la part du système. Une grande expertise est toutefois indispensable pour arriver à générer des solutions complètement différentes de celles auxquelles on s'attend. Ceci semble un problème complexe, propice à de nombreuses réflexions.

3.6.2. *Synthèse*

La conception d'un produit regroupe toutes les phases qui permettent de passer progressivement du cahier des charges initial au produit réalisé. En effet, les spécifications du cahier des charge sont tout d'abord exprimées en un ensemble équivalent de requêtes fonctionnelles puis ces fonctions sont décomposées en sous-fonctions. Lorsque la décomposition est achevée, elle est analysée pour détecter d'éventuels oublis ou des redondances. Ensuite, les requêtes fonctionnelles sont transformées en caractéristiques. Enfin, il reste à valider le modèle de conception obtenu.

Il apparaît, dans les publications, que la conception par les caractéristiques constitue une bonne base pour un système de conception fonctionnelle. En effet, le fait de modéliser dans les caractéristiques à la fois la forme et la fonction est un atout indéniable dans la phase de transition entre une représentation des requêtes fonctionnelles du produit à concevoir et une solution possible par des formes fonctionnelles données. Nous développons donc notre propre système de conception à base de caractéristiques, en vue d'écrire plus tard la couche permettant d'interfacer les deux phases et de modéliser toute l'information nécessaire à une complète gestion de l'historique et de l'intention de conception.

4. CONCLUSION

Cette étude bibliographique a permis de mettre en évidence un certain nombre de points précis qui doivent être mis en œuvre dans un système de CAO / CFAO. Certaines propositions ont été retenues car jugées intéressantes, d'autres seront faites dans les chapitres suivants.

Un système de conception par les caractéristiques doit a priori permettre de prendre en compte des types de caractéristiques variés et notamment les caractéristiques de forme, de précision, de matériau... Il semble cependant que les caractéristiques de forme soient la base, elles doivent donc être étudiées en premier lieu. Les autres types de caractéristiques sont souvent considérés comme des attributs des caractéristiques de forme.

Le système doit ensuite permettre de manipuler ces caractéristiques pendant toutes les phases du cycle de vie du produit, de la conception à la fabrication. Nous retiendrons donc qu'un bon système doit permettre :

- d'instancier des caractéristiques dans le modèle de conception, en y associant éventuellement des contraintes ;
- de vérifier la validité des attributs géométriques c'est-à-dire les paramètres de position et de dimensions ;
- de disposer d'un processeur d'opérations booléennes, capable de résoudre les problèmes d'intersection entre les caractéristiques de forme ;
- de conserver un historique de construction suffisamment riche pour le maintien de l'intention de conception ;
- de modifier a posteriori les dimensions ou l'orientation des caractéristiques ;
- d'associer un attribut (une tolérance par exemple) à une face ou une partie de face;
- de pouvoir s'adapter à différents contextes d'application (*mapping*) ;
- de prévoir l'extraction automatique de caractéristiques de forme en cas d'apparition fortuite ;
- de tester la fabricabilité (direction d'approche de l'outil...).

Les propositions qui seront faites dans la suite devront répondre au mieux à toutes ces requêtes. Les caractéristiques prises en compte sont les caractéristiques de forme. L'architecture proposée est composée :

- d'une bibliothèque de caractéristiques prédéfinies, qui peut être enrichie par de nouvelles caractéristiques définies par l'utilisateur ;
- d'un modèle permettant de mémoriser les instances de caractéristiques, qui lui-même se décompose en deux couches. La première est une couche sémantique de haut niveau ; c'est un graphe de relations, qui permet de gérer les relations spatiales et topologiques entre les caractéristiques ainsi que les contraintes liant les paramètres. La seconde couche est un modèle géométrique de type B-Rep, qui renferme les informations de plus bas niveau.

La finalité d'un tel modèle est avant tout une aide à la conception, dans la mesure où il garantit le maintien de l'intention de conception en garantissant le respect des contraintes imposées par le concepteur. De plus, il devrait permettre l'intégration de la production, grâce à toutes les informations de haut niveau qu'il contient.

Chapitre 2 - UN NOUVEAU MODÈLE À BASE DE CARACTÉRISTIQUES

1. INTRODUCTION

L'étude bibliographique qui a fait l'objet du chapitre 1 a montré l'importance des caractéristiques de forme pour la modélisation produit. Même si les caractéristiques de forme constituent la base de tout système de modélisation en CAO, il est évident qu'elles ne sont pas suffisantes ; il faudrait notamment ajouter les tolérances et les caractéristiques d'assemblage. Nous débuterons toutefois nos propositions pour un modèle produit par une étude approfondie des caractéristiques de forme et une façon de les modéliser.

Avant tout, il est important de donner la définition, parmi d'autres possibles, que nous avons choisi d'adopter dans ce manuscrit :

> Une *caractéristique de forme* est une entité permettant de définir une forme (géométrie et topologie) et d'y associer des informations complémentaires sur la sémantique ou la fonction.

Une définition plus spécifique de chacune des caractéristiques manipulées reste souvent imprécise. Il semble par exemple très difficile, quelles que soient les personnes consultées, d'obtenir une définition générale de caractéristiques comme la rainure ou le trou traversant, pourtant couramment cités dans la bibliographie. Selon leur spécialité, les personnes interrogées voient la rainure sous un angle cinématique (guidage en translation) ou fabrication (par enlèvement de matière). En cela, elles confirment la nécessité d'une multi-modélisation,

permettant à chaque étape de la production d'envisager le modèle selon différents points de vue. La définition la plus complète que nous ayons recueillie de la forme de la rainure est : *dépression engendrée par une extrusion d'un profil sur la surface d'un objet, la longueur de la trajectoire étant grande devant la dimension du profil.* Il est clair que cette définition, qui ne vise pourtant qu'à décrire la géométrie, est difficilement exploitable d'un point de vue informatique. C'est vraisemblablement la raison pour laquelle les études menées sur les rainures, que ce soit en conception par les caractéristiques ou en extraction, font toujours de fortes hypothèses. En général, la rainure est constituée de trois faces planes. La figure 2.1 donne quelques exemples de rainures.

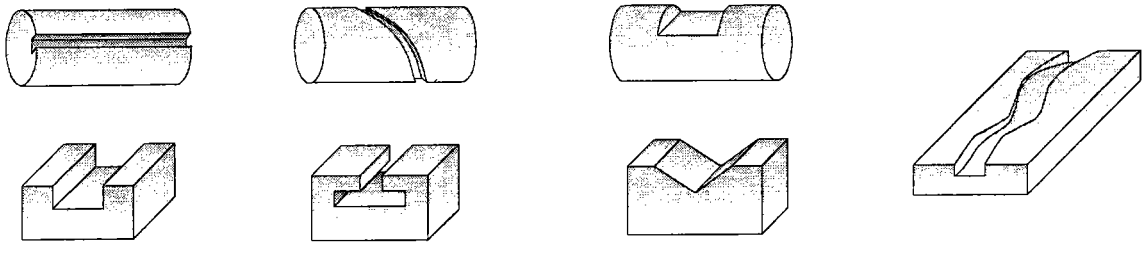


Figure 2.1 : Exemples de rainures.

Le trou traversant prête aussi à confusion. Faut-il comprendre qu'un trou traversant perce de part en part la pièce qui le porte ou, au contraire, qu'il s'arrête dès qu'il débouche sur du vide. Si une opération booléenne vient à obturer le trou, ce dernier doit-il automatiquement être prolongé jusqu'à déboucher de nouveau ? La réponse à ces questions n'est pas universelle, aussi faut-il pouvoir préciser le comportement d'une caractéristique en décrivant la manière dont elle doit réagir lors de son insertion dans le modèle ou lors des modifications qu'elle subira. Notons que la rainure et le trou n'ont été cités qu'à titre d'exemples, mais que d'autres caractéristiques posent également problème.

Ces quelques définitions étant données, nous proposons d'abord une manière de modéliser les caractéristiques, après avoir fait l'inventaire de toutes les informations qu'elles doivent regrouper. Dans une seconde partie, nous développons l'instanciation de telles caractéristiques dans un modèle de conception. La troisième partie sera consacrée aux autres traitements que l'on peut réaliser à partir d'un modèle à base de caractéristiques.

Nous ferons dans ce chapitre l'ensemble des propositions auxquelles a abouti notre réflexion. Cependant, certains points restent à développer, soit parce qu'une réflexion plus approfondie est nécessaire, soit simplement parce qu'ils n'ont pas été programmés ; ils seront signalés.

2. MODÉLISATION DES CARACTÉRISTIQUES DE FORME

Nous avons réalisé l'inventaire de toutes les informations à prendre en compte par les caractéristiques, pour aboutir à leur modélisation dans le système de CAO. Avant de parler de la modélisation proprement dite, une définition supplémentaire est toutefois nécessaire :

> On appelle *caractéristique générique* une entité qui contient toutes les informations nécessaires pour générer une instance de caractéristique de forme, dans le modèle de conception.

Une distinction est donc faite entre les informations nécessaires pour la génération de caractéristiques et les informations propres à chaque instance. En effet, les informations à mémoriser au niveau des caractéristiques génériques sont celles qui sont communes à toutes les instances de caractéristiques de même nature (par exemple toutes les rainures). Certaines informations complémentaires sont propres à chaque instance, il faut donc également pouvoir les mémoriser, mais pas au niveau générique. C'est pourquoi nous faisons une distinction entre les caractéristiques génériques et les caractéristiques d'instanciation.

Les caractéristiques génériques sont mémorisées dans une structure de données adéquate que l'on appellera bibliothèque de caractéristiques. C'est en fait une base de données qui contient toutes les caractéristiques prédéfinies disponibles, avec leurs données et leurs méthodes. Nous verrons l'architecture de cette bibliothèque plus en détail au chapitre 4, qui traite de l'intégration dans cette bibliothèque de nouvelles caractéristiques définies par l'utilisateur.

2.1. Modélisation des caractéristiques génériques

La définition des caractéristiques de forme que nous avons donnée peut être formalisée par un inventaire de toutes les informations qu'il faut leur associer, pour les modéliser de la manière la plus complète possible. Nous avons d'abord souhaité classer ces informations en données et traitements. Mais cette classification est plus liée à la manière dont sont modélisées les informations qu'à leur nature. Par exemple, la forme peut être modélisée par un B-Rep, ce qui correspond à une donnée, ou par une procédure de construction, ce qui correspond à un traitement. Dans la suite, nous décrivons donc tous les composants d'une caractéristique, sans se soucier de les classer en données ou traitements. Cette répartition sera faite dans la conclusion, lorsque le modèle retenu pour chaque composante aura été décrit.

Une nuance semblable est fréquemment faite, dans la bibliographie, entre données et comportement. Si l'on considère que le comportement est la manière dont une caractéristique réagit au sollicitations de son environnement, nous constatons que tous les composants évoqués ci-dessus, du simple paramètre à la plus compliquée des contraintes, influent sur le comportement. Nous ne ferons donc pas non plus de distinction entre données et comportement.

Les caractéristiques prises en compte ne jouent pas toutes le même rôle. En effet, certaines constituent des volumes à part entière et d'autres décrivent des aspects locaux de la forme. C'est pourquoi nous avons classé les caractéristiques de forme en deux catégories :

- les caractéristiques primaires sont des volumes de base tels que la boîte parallélépipédique, le cylindre...
- les caractéristiques secondaires décrivent des aspects locaux de la forme et leur sémantique, et permettent d'altérer les caractéristiques primaires ou des combinaisons de caractéristiques primaires (rainure, congé, trou...).

Les caractéristiques secondaires ne peuvent pas exister de façon indépendante ; elles nécessitent un support ou volume porteur, qui peut être constitué d'une caractéristique primaire par exemple. Notons que toute conception débute forcément par la création d'une caractéristique primaire. Il s'agit d'une convention que nous nous sommes imposée. Cependant, si l'utilisateur désire ne réaliser qu'une vue détaillée d'une caractéristique secondaire, sans définir un volume porteur, nous pouvons l'envisager par quelques adaptations sommaires. Nous pensons donc que cette convention ne pose pas de problème.

Les caractéristiques secondaires influent sur le volume porteur en ajoutant ou en retirant une certaine quantité de matière, ce qui correspond à une opération booléenne entre cette quantité de matière et le volume porteur (union ou différence). Cette information est importante car elle permet de déduire les opérations booléennes à réaliser lors de l'instanciation, dont les étapes seront détaillées au paragraphe 3. Les caractéristiques secondaires sont donc de deux types : elles sont positives si elles ajoutent de la matière (bossage) et négatives si elles en retirent (dépression).

Ces deux informations, la catégorie (primaire ou secondaire) et le type (positif ou négatif), peuvent apparaître par le biais d'une hiérarchie de caractéristiques plutôt que comme des attributs, comme le montre la figure 2.2. Nous mettons donc en évidence l'intérêt d'une structure orientée objets pour la mise en œuvre, qui permet grâce à l'héritage de définir cette hiérarchie.

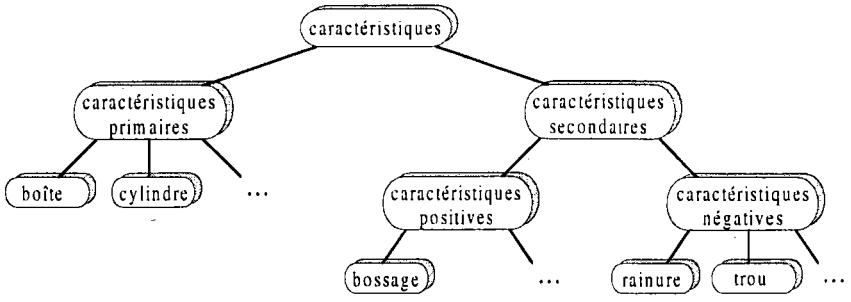


Figure 2.2 : Hiérarchie des caractéristiques.

Une fois modélisés la catégorie et le type de la caractéristique, une autre information à stocker dans la structure des caractéristiques génériques est une liste de paramètres. Nous appellerons paramètre toute quantité, numérique ou non, qui peut être soumise à des contraintes et dont la valeur doit être indiquée à l'instanciation. Un paramètre peut donc être une valeur numérique (largeur d'un bloc) ou un objet géométrique (plan de référence pour le placement d'une caractéristique). Par exemple, on peut imposer qu'une poche soit placée par rapport à une face de son porteur, à une certaine distance. Le paramétrage de la poche contient alors une surface de référence (paramètre géométrique) et une valeur pour la distance (paramètre numérique).

Au niveau générique, il faut également prévoir de conserver des contraintes génériques, permettant de définir une partie du comportement de la caractéristique. Ces contraintes peuvent s'exprimer directement par des équations. Par exemple, si le diamètre du trou doit être supérieur au quart de la hauteur dans toutes les instances, il existe une équation du type "diamètre > ¼ hauteur" dans la définition générique du trou.

Nous définissons également des contraintes génériques géométriques telles que le parallélisme, la perpendicularité, la tangence... Par exemple, les faces d'une boîte doivent toujours être parallèles deux à deux si elles sont en vis-à-vis, et perpendiculaires si elles sont adjacentes.

Un dernier type de contraintes, différent des équationnelles et des géométriques, peut être exprimé par la nature même des caractéristiques. Par exemple, le critère "débouchant" d'un trou est également considéré comme un comportement générique et est considéré à ce niveau. L'intention de l'utilisateur (*design intent*) qui est, dans ce cas, de prolonger le trou jusqu'à ce qu'il débouche de son porteur, est ainsi conservée.

Une autre partie du comportement, au niveau générique, est la description de la forme de la caractéristique. Pour une caractéristique primaire, la forme est décrite simplement par un volume de base prédéfini tel qu'une boîte ou un cylindre, par exemple. Par contre, les caractéristiques de forme secondaires étant des altérations de la surface d'un objet, nous préférons les modéliser au

niveau générique comme des demi-espaces qui engendreront, après dimensionnement et opération booléenne, la caractéristique instanciée. Par exemple, à un trou cylindrique non débouchant correspond une surface cylindrique limitée d'un côté par un disque, représentant le fond du trou, et infinie de l'autre côté, comme le montre la figure ci-dessous. Le demi-espace considéré est l'intérieur de la surface. Nous avons choisi de mémoriser la forme ainsi après une réflexion approfondie sur la description de la forme d'une caractéristique. Ce choix sera justifié en détail dans le chapitre 4.

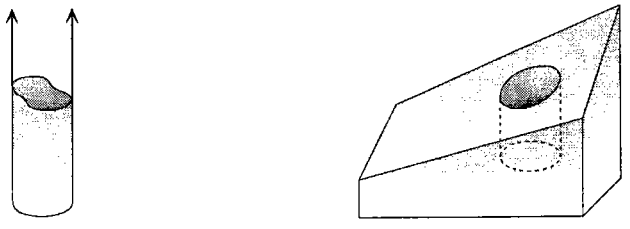


Figure 2.3 : La surface génératrice d'un trou cylindrique borgne et un exemple d'instanciation.

Quels que soient l'aspect et l'orientation du volume à percer, ceci permet de réaliser l'instanciation par une opération booléenne. On obtient en même temps le comportement de prolongement automatique de la caractéristique jusqu'à l'endroit où elle est supposée se terminer, qu'elle ajoute ou retire de la matière. Nous proposons donc de définir la forme au niveau générique par une liste de surfaces, partiellement limitées et éventuellement infinies par ailleurs, qui portent la peau de la caractéristique ; cette liste de surfaces, qui délimite deux demi-espaces dont l'un représente la caractéristique, sera appelée la surface génératrice. La génératrice peut être un volume fermé borné, notamment pour toutes les caractéristiques primaires. Pour les secondaires, la surface peut s'étendre à l'infini de certains côtés, là où l'on souhaite qu'elle se prolonge, c'est-à-dire du côté où elle débouche de son porteur si elle est négative et du côté où elle atteint ses frontières si elle est positive. La structure doit donc permettre la mémorisation de telles surfaces et la représentation des contraintes géométriques qui les lient, telles que distants, parallèles, perpendiculaires...

Les exemples de la figure 2.4 montrent que la représentation des caractéristiques de forme par leur surface génératrice semble possible pour la plupart des caractéristiques usuelles. Nous pouvons aussi citer, à titre d'exemple supplémentaire, le chanfrein, dont la génératrice est un simple plan, délimitant deux demi-espaces. C'est une caractéristique négative, puisqu'elle retire de la matière pour abattre une arête.

Nous constatons que les génératrices sont souvent les mêmes pour les caractéristiques négatives et les caractéristiques positives qui leur correspondent (rainure/languette,

poche/bossage...). L'arrondi et le congé ont également la même génératrice, une surface cylindrique qui délimite deux demi-espaces. Le demi-espace considéré est, dans les deux cas, la partie extérieure de la surface cylindrique. Il faudra cependant limiter le demi-espace, par des faces du porteur par exemple. Nous reviendrons ultérieurement sur ce problème de limitation des volumes infinis.

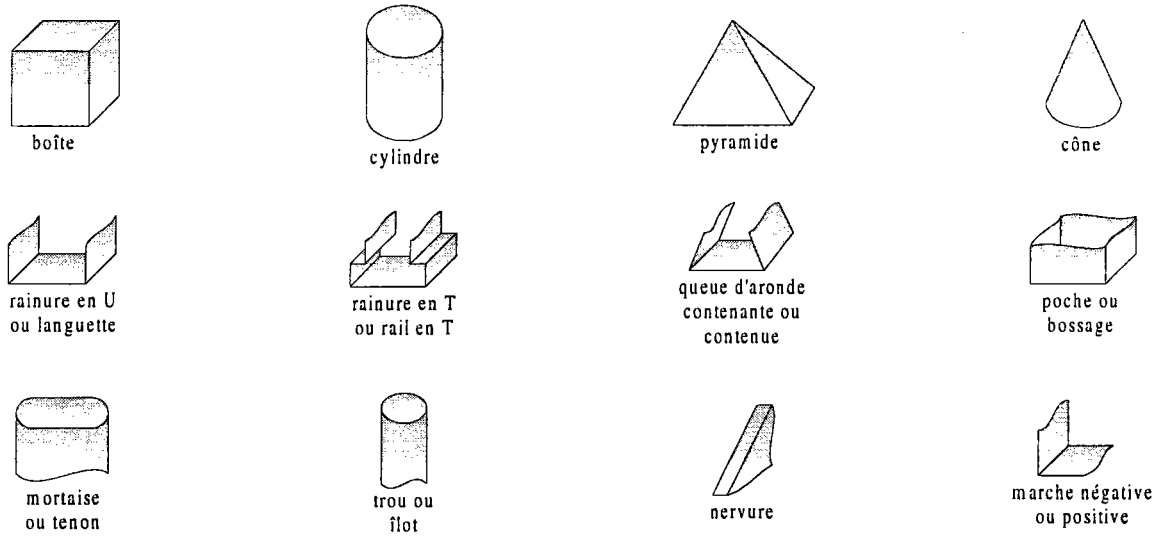


Figure 2.4 : Exemples de génératrices de caractéristiques de forme usuelles.

Pour mémoriser la génératrice d'une caractéristique, nous avons choisi de conserver, au niveau générique, sa procédure de création, qui représente la liste des instructions à effectuer pour créer la forme. Plutôt que de conserver une procédure, nous pourrions stocker une description évaluée, dans un B-Rep. Cependant, au niveau générique, les valeurs des paramètres ne sont pas connues. Il faudrait donc pouvoir déformer le B-Rep[1] de la génératrice, au moment de l'instanciation, lorsque les valeurs des paramètres sont données. Nous préférons donc mémoriser la procédure de création de la forme. Elle permet de générer, en fonction des paramètres, une représentation évaluée de la génératrice de la caractéristique qui n'est pas seulement constituée d'un B-Rep, mais aussi d'un graphe conceptuel (dont la définition a été donnée au chapitre 1, paragraphe 3.3.2), pour des raisons expliquées plus bas. Nous donnons ci-dessous un exemple sommaire de procédure de création d'une poche, créée à partir d'une boîte parallélépipédique dont la face supérieure est supprimée.

[1] Le B-Rep d'un objet est le volume représenté dans le modèle B-Rep, sous forme de faces, d'arêtes et de sommets.

```
Procédure CréationFormePoche (larg, long, rep3D, forme)
   /* Entrée :                                                          */
   /*        larg      représente la largeur de la poche               */
   /*        long      représente la longueur de la poche              */
   /*        rep3D     représente le repère local, c'est-à-dire la position */
   /* Sorite :                                                          */
   /*        forme     représente la forme de la caractéristique créée (B-Rep + graphe) */
Début
      /* Création d'une boîte dont le paramètre hauteur est un paramètre par défaut */
      CréationFormeBoîte (larg, long, HautParDef, rep3D)
      /* Identification de la face du dessus de la boîte               */
      i = FaceDuDessus ()
      /* Suppression de la face du dessus de la boîte                  */
      SuppressionFace (i)
Fin
```

Le B-Rep généré par la procédure de création est une représentation explicite, qui permet de conserver toutes les informations géométriques et topologiques relatives à la forme. Il ne permet pas, par contre, de mémoriser les informations d'un niveau sémantique plus élevé telles que les contraintes génériques internes de la caractéristique (par exemple le parallélisme de deux faces opposées dans une boîte). Il faut donc associer au B-Rep une structure permettant de stocker toutes les contraintes génériques de la génératrice de la caractéristique. Pour cela, la procédure de création génère également un graphe de contraintes ; on trouve aussi le terme graphe conceptuel dans la bibliographie.

Les nœuds de ce graphe sont des géométries (points, lignes, surfaces) correspondant à des entités topologiques réelles telles que les sommets, arêtes et faces, ou virtuelles comme les plans ou axes de référence ou de symétrie. Les nœuds ne sont pas les entités topologiques elles-mêmes, car une même géométrie peut correspondre à plusieurs entités topologiques différentes. En effet, deux faces coplanaires, par exemple, partagent la même géométrie. Si les contraintes sont appliquées aux faces par exemple, il faudrait alors les dupliquer autant de fois que de faces ayant la même géométrie, ce qui n'est pas souhaitable. En particulier, si une face est contrainte et qu'à la suite d'une opération de modélisation (instanciation de caractéristique secondaire, opération booléenne), elle est divisée en deux, les contraintes fixées doivent s'appliquer aux deux faces résultantes. Ceci est automatique si les contraintes portent sur les géométries. C'est le cas, par exemple, lors de l'instanciation d'une rainure traversante : la face porteuse de la rainure est divisée.

Les arcs qui relient les nœuds sont valués par des contraintes géométriques et éventuellement des paramètres. Les contraintes expriment, par exemple, le positionnement et l'orientation des surfaces les unes par rapport aux autres : angle, distance... Des paramètres complètent la contrainte si nécessaire : pour une contrainte de parallélisme, aucune valeur numérique n'est utile, mais pour imposer un angle entre deux faces, il faut avoir connaissance de la valeur de

l'angle. Ces paramètres sont de deux catégories. Les paramètres de dimensions relient deux nœuds (surfaces) appartenant à la même caractéristique, comme la largeur de la rainure sur la figure 2.5. Les paramètres de position ou d'orientation relient les nœuds de plusieurs caractéristiques, comme les paramètres *pos1* ou *pos2* sur la figure. À chaque paramètre est associé un nom (largeur, hauteur, profondeur...). Sachant à quel arc il se rapporte, on peut aisément retrouver à quoi il correspond (distance entre deux faces par exemple), c'est-à-dire sa sémantique. Ceci est particulièrement utile pour calculer les valeurs des paramètres d'une caractéristique mise en évidence par un mécanisme d'extraction (voir paragraphe 4.1). La nécessité de nommer les paramètres est liée à leur utilisation, rendue plus facile et conviviale, au moment de l'ajout, par l'utilisateur, de contraintes portant sur ces valeurs.

Les arcs du graphe sont orientés par un sens correspondant à la chronologie de la création, c'est-à-dire permettant, pour une distance entre deux faces par exemple, de savoir laquelle a été placée par rapport à l'autre. Cette information est fondamentale dans le cas d'une gestion paramétrique des contraintes, elle l'est moins dans l'hypothèse d'une gestion variationnelle [ChS 90]. En effet, la gestion paramétrique est basée sur une résolution locale de systèmes engendrés au fur et à mesure de la construction ; elle respecte donc un historique. La gestion variationnelle est basée sur une compréhension globale d'une scène construite en faisant apparaître les liens entre les objets, par exemple dans un graphe de liaisons [LiG 82]. Nous conservons donc l'orientation des arcs, pour ne pas être limité à un seul type de gestion des contraintes.

Figure 2.5 : Exemple de paramètres de dimensions et de position.

La présence du graphe conceptuel au niveau générique offre de multiples avantages. Il est par exemple impliqué dans les mécanismes d'extraction de caractéristiques (voir paragraphe 4.1 et [SSJ 94]). Il facilite notamment l'évaluation automatique des paramètres lors de l'extraction, puisque l'on sait exactement à quoi correspondent les paramètres : par exemple une largeur correspond à la distance entre deux faces précises.

En outre, le stockage des contraintes génériques dans le graphe contribue au maintien de l'intention de conception. En effet, si un cube est créé par l'opérateur, ses contraintes génériques seront vérifiées pendant toute l'évolution du modèle de conception, à savoir : ses faces sont

parallèles quand elles sont opposées, et perpendiculaires quand elles sont adjacentes. Le cube restera donc un cube et, de ce point de vue, l'intention du concepteur, qui est d'utiliser un cube, sera maintenue.

Enfin, le graphe conceptuel représente également le moyen de fixer un mode d'ancrage pour les caractéristiques secondaires. En effet, nous avons évoqué l'existence de paramètres géométriques, qui permettent de préciser le positionnement de la caractéristique. Ces paramètres sont des géométries représentées par des nœuds du graphe. Par exemple, si le placement est déterminé par une distance entre une face de la caractéristique et une face de référence, la valeur du paramètre de position de la caractéristique est la géométrie de la face de référence. Il suffit donc de rajouter dans le graphe une contrainte, c'est-à-dire un arc. La première extrémité de cet arc est le nœud représentant la face de la caractéristique que l'on souhaite contraindre. L'autre extrémité est le nœud donné en paramètre, qui ne sera donc précisé qu'au moment de l'instanciation, quand la face de référence sera connue.

Des liens bidirectionnels entre le graphe et le B-Rep sont nécessaires pour maintenir la cohérence entre ces deux représentations implicite (graphe) et explicite (B-Rep). En effet, quand une face est supprimée dans le B-Rep, il faut également supprimer dans le graphe la géométrie sur laquelle elle s'appuie, si elle est la seule face à utiliser cette géométrie. C'est pourquoi, il faut pouvoir connaître la géométrie porteuse d'une face donnée et également l'ensemble des faces s'appuyant sur une géométrie précise. L'exemple ci-dessous montre une rainure représentée par son B-Rep, son graphe conceptuel et les liens entre les deux.

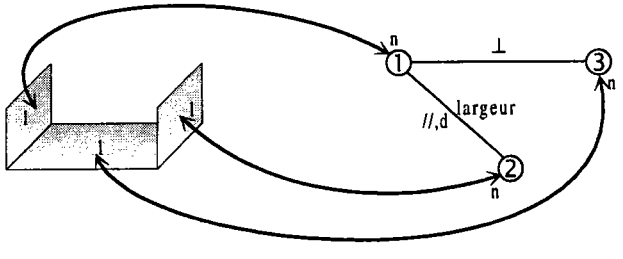


Figure 2.6 : B-Rep et graphe conceptuel de la rainure.

En conclusion, au niveau générique, nous avons choisi de ne décrire la forme de la caractéristique qu'au travers d'une procédure de création. Le graphe conceptuel et le B-Rep ne deviennent utiles qu'à l'instanciation et sont donc calculés à cet instant par cette procédure. Ils sont alors liés à la caractéristique d'instanciation. Si la procédure de création passe par exemple par la création d'un bloc, six faces sont créées dans le B-Rep. Les contraintes de parallélisme et de perpendicularité entre ces faces sont alors ajoutées au graphe conceptuel.

Il reste à prévoir au niveau générique, des moyens pour définir le reste du comportement. Nous avons vu, dans le chapitre 1, qu'il y a une différence de points de vue entre la conception et les diverses applications. Il faut donc être capable de fournir un modèle bien adapté à chacun de ces points de vue. Pour cela, il faut ajouter à la caractéristique générique des règles permettant de faciliter le *mapping*, c'est-à-dire la transformation des caractéristiques de conception en caractéristiques d'application.

De plus, il se peut que des caractéristiques apparaissent de manière non explicite, au cours de la création du modèle de conception. L'extraction automatique de telles caractéristiques doit être facilitée par des règles d'extraction. Nous remarquons que l'extraction de caractéristiques peut se révéler utile pour le *mapping*, puisqu'elle peut aussi être guidée par les points de vue.

Nous reviendrons sur ces aspects dans le paragraphe 4.

Nous avons dit en introduction que la distinction entre les données et les traitements d'une caractéristique générique ne pouvait être faite qu'après avoir proposé une représentation pour chaque composant. Ceci étant fait, nous pouvons maintenant affecter chaque composant à une de deux catégories.

Les données sont :

- un identifiant (nom, numéro par exemple) ;
- une catégorie : primaire ou secondaire ;
- un type : positif ou négatif (pour les caractéristiques secondaires) ;
- un ensemble de paramètres génériques ;
- un ensemble de contraintes génériques ;

et les méthodes sont :

- une procédure de création de la forme ;
- le reste du comportement (*mapping*, règles d'extraction...).

Une caractéristique de forme est l'agrégation de ses données et de ses traitements, ce qui correspond fidèlement à la notion d'encapsulation des langages orientés objets. C'est une des raisons pour lesquelles nous avons opté pour ce mode de programmation.

Ces informations ne sont cependant pas suffisantes pour que le modèle de conception soit complet. D'autres données, spécifiques à chaque instance, sont nécessaires. Elles seront prises en compte au niveau des caractéristiques d'instanciation, décrites dans le paragraphe suivant.

2.2. Modélisation des caractéristiques d'instanciation

Pour créer une caractéristique dans le modèle de conception, il est tout d'abord nécessaire de préciser la caractéristique générique, qui permet de générer une caractéristique d'instanciation. Les informations apportées par la caractéristique générique sont utiles, voire indispensables, à la caractéristique d'instanciation, par exemple les paramètres, les contraintes génériques, la procédure de création de la forme... Elles ne sont cependant pas suffisantes et il faut les compléter par d'autres, qui sont détaillées dans ce qui suit. Une caractéristique générique est donc un composant ou une donnée (au sens informatique) d'une caractéristique d'instanciation.

Pour une caractéristique secondaire, il faut ensuite préciser son volume porteur, c'est-à-dire le volume sur lequel elle s'appuie. Ce volume porteur est composé au minimum d'une caractéristique primaire, mais peut aussi être la combinaison de diverses caractéristiques primaires et secondaires. Une représentation informatique de la donnée "volume porteur" peut être simplement l'identifiant du volume porteur.

Quand le porteur est déterminé, la caractéristique à instancier peut y être placée de façon précise. Pour cela, la position et l'orientation sont spécifiées par le biais d'un repère local à la caractéristique d'instanciation, associé à une matrice de changement de repère, qui sera exprimée dans le repère global du monde. Pour une caractéristique secondaire, nous pourrions exprimer le changement de repère dans le repère local du porteur. Ceci imposerait donc de définir un repère local pour les volumes, comme étant par exemple le même que celui de la caractéristique primaire ayant permis de le créer. L'intérêt de préciser le repère local d'une caractéristique par rapport à son porteur est cependant limité, puisque le placement est réalisé grâce à des contraintes, comme nous le verrons au paragraphe 3.2.

Les valeurs des paramètres étant connues (fournies par la caractéristique générique), ainsi que la position et l'orientation, la procédure de création de la forme de la caractéristique générique peut être exécutée. Cette procédure génère une représentation B-Rep et un graphe conceptuel pour la caractéristique d'instanciation, ce qui a été détaillé au paragraphe précédent. D'un point de vue informatique, la procédure de création est une méthode de la caractéristique générique. Par contre, le B-Rep et le graphe sont des données de la caractéristique d'instanciation, car ils n'ont pas un rôle générateur, mais sont des représentations évaluées de la caractéristique d'instanciation.

Une autre information à conserver au niveau d'une caractéristique d'instanciation est le volume de matière exact qu'elle ajoute ou retire, que nous appellerons son volume utile. Le volume utile est obtenu à partir de la génératrice par un algorithme décrit au paragraphe 3.3. Nous remarquons que le volume de matière ajouté ou soustrait n'est pas forcément composé d'une seule composante connexe. Nous parlerons tout de même du volume utile, en admettant qu'il peut être composé de plusieurs volumes disjoints. Ce volume utile une fois calculé est stocké par son B-Rep, à la place du B-Rep de la génératrice qui n'a alors plus de raison d'être conservé. Nous pouvons faire une analogie entre ce volume utile et le FTS proposé par [GoT 91] ou le FPV de [ShR 88a] (cf. chapitre 1). De nombreux auteurs jugent en effet utile de conserver les volumes individuels de toutes les caractéristiques du modèle [CMV 91a, ShL 93, LaM 93, MFG 94].

Lorsque le volume utile est calculé, on peut déterminer un volume englobant parallélépipédique pour la caractéristique d'instanciation. Il est très intéressant de conserver cet englobant pour optimiser le calcul des intersections entre englobants et détecter ainsi les interférences entre caractéristiques. Il n'est cependant pas nécessaire de conserver le B-Rep de l'englobant. En effet, comme il est parallélépipédique et comme ses côtés sont parallèles aux axes du repère, il suffit de conserver deux points opposés, par exemple ceux ayant respectivement les valeurs minimum et maximum en x, y et z.

Une dernière information complémentaire à associer aux caractéristiques d'instanciation est la liste des contraintes d'instanciation, notamment celles définies lors du positionnement ou du dimensionnement de l'instance. On peut citer des contraintes équationnelles déterminant un paramètre en fonction d'un autre : la largeur de la languette est égale à deux fois le rayon du trou. On peut définir également des contraintes géométriques telles qu'une distance, pour placer une rainure par exemple, par rapport à son porteur. La face du fond de la rainure est distante d'une certaine valeur, donc également parallèle, d'une face donnée du porteur. Deux exemples de contraintes de placement d'une rainure sont indiqués dans la figure 2.7.

Figure 2.7 : Exemples de contraintes d'instanciation pour le placement de la rainure.

En résumé, les informations liées à une caractéristique d'instanciation sont :

- une caractéristique générique ;
- un volume porteur (pour les caractéristiques secondaires) ;
- un repère local précisant la position et l'orientation ;
- la forme exprimée par un graphe conceptuel et un B-Rep du volume utile ;
- un englobant ;
- une liste de contraintes d'instanciation.

Ces informations sont toutes nécessaires, à un moment ou à un autre dans la gestion du modèle de conception. Notons toutefois quelques redondances : l'englobant par exemple, peut être calculé à partir du volume utile et pourrait donc ne pas être conservé. Cependant, sa mémorisation est justifiée par un souci d'efficacité. En outre, le graphe conceptuel conservé dans la caractéristique d'instanciation est redondant par rapport à la procédure de création de la caractéristique générique, qui permet de l'obtenir. Nous préférons tout de même le conserver, pour faciliter l'insertion de contraintes d'instanciation.

Le paragraphe suivant valide cette proposition de modélisation des caractéristiques de forme. La structure proposée y est mise à l'épreuve pour gérer tous les problèmes liés à l'instanciation.

3. INSTANCIATION DES CARACTÉRISTIQUES

Pendant la phase de conception, l'opérateur utilise des caractéristiques fournies par le logiciel de CAO pour décrire son modèle de conception. Le premier intérêt de la modélisation basée sur les caractéristiques par rapport à une modélisation géométrique classique est de pouvoir associer des comportements et des fonctions aux caractéristiques. Ce sont des informations de haut niveau, qui ne sont pas incluses dans les modeleurs géométriques. Le second intérêt est de pouvoir instancier ces caractéristiques plusieurs fois, sans les redéfinir. Nous supposerons pour cela qu'il existe une bibliothèque de caractéristiques génériques dans laquelle chacune d'elles est définie sous la forme d'un ensemble de données et d'un ensemble de procédures régissant son comportement. La structure de la bibliothèque sera détaillée dans le chapitre 4, traitant de l'extension de la panoplie des caractéristiques fournies par le logiciel.

L'utilisation des caractéristiques passe avant tout par leur insertion dans le modèle de conception. Or, si une caractéristique correspond à une classe comme nous le suggérions au paragraphe 2.1, cette insertion peut très bien être prise en charge par le mécanisme

d'instanciation. La programmation objets apparaît donc comme un bon choix et est quasi unanimement adoptée. Nous verrons, dans le chapitre 5, qu'une méthodologie orientée objets a été suivie pour la mise en œuvre des caractéristiques génériques, des caractéristiques d'instanciation, de la bibliothèque et du modèle de conception, c'est-à-dire du modeleur à base de caractéristiques.

Pour l'opérateur, la différence entre caractéristique générique et caractéristique d'instanciation est transparente lorsqu'il veut instancier une caractéristique dans le modèle de conception. Les renseignements qu'il doit fournir au moment de l'instanciation seront indifféremment ceux concernant la caractéristique générique et ceux concernant la caractéristique d'instanciation.

L'instanciation doit, dans cet ordre :

- permettre le choix de la caractéristique ;
- permettre l'acquisition des paramètres et des contraintes ;
- calculer la forme exacte de la caractéristique (c'est-à-dire son volume utile) et dans le cas d'une caractéristique secondaire, la combiner avec son volume porteur ;
- vérifier la validité ;
- enregistrer la caractéristique dans une structure adaptée.

Nous détaillons ci-dessous chacune des étapes de l'instanciation.

3.1. Choix de la caractéristique

La première étape de l'instanciation consiste à choisir la caractéristique à créer. Dans le cadre d'une conception par les caractéristiques au sens classique, ce choix ne pose pas de problème particulier. Les caractéristiques disponibles dans la bibliothèque sont proposées soit sous forme d'une liste de noms, soit sous forme d'icônes. L'opérateur peut ainsi consulter la bibliothèque et sélectionner la caractéristique qu'il désire insérer dans le modèle de conception.

Une approche plus ambitieuse, mais qui n'est pas dans notre propos dans ce mémoire, est d'automatiser le choix de a caractéristique en s'appuyant sur des contraintes fonctionnelles fournies par l'opérateur ou déduite d'une analyse fonctionnelle menée par le système. Cette orientation constitue vraisemblablement le prochain grand enjeu de la CAO et est évoquée à la fin de ce manuscrit comme une perspective de notre travail.

Les informations concernant la catégorie (primaire ou secondaire) de la caractéristique et son type (positif ou négatif) si elle est secondaire, ne sont pas précisées explicitement par l'opérateur.

Ces informations sont des données fixes au niveau de chaque caractéristique générique, selon la place quelle occupe dans la hiérarchie de la figure 2.2 ; elles sont donc précisées implicitement par le choix d'une caractéristique.

3.2. Acquisition des paramètres et contraintes

Une fois la caractéristique choisie dans la bibliothèque, l'opérateur doit fournir les valeurs de ses paramètres. Selon le cas, ces paramètres génériques permettent de dimensionner l'instance, mais aussi parfois de la positionner et de l'orienter, si le mode d'ancrage est fixé. Cependant, même si ce n'est pas le cas, le placement de la caractéristique doit être réalisé au moment de l'instanciation, d'une autre manière que par le biais de paramètres.

Pour placer une caractéristique secondaire, une première étape consiste à déterminer sur quel objet elle doit se situer, c'est-à-dire à indiquer son volume porteur. En effet, une caractéristique secondaire ne peut pas exister de façon individuelle, elle doit obligatoirement se rattacher à un porteur. Pour une caractéristique primaire, par contre, il n'y a pas de désignation de porteur à réaliser.

Pour positionner et orienter une caractéristique, une solution serait d'exprimer son repère local par rapport au repère global du monde ou par rapport à celui du porteur si c'est une secondaire. Cependant, le fait de devoir déterminer un repère local est fastidieux pour l'opérateur. Il s'agit en effet, dans un premier temps, de placer l'origine du repère local de la caractéristique, puis d'indiquer trois angles pour fixer son orientation. La tâche pourrait être simplifiée en proposant à l'opérateur de définir simplement un certain nombre de translations et de rotations, jusqu'à ce que la caractéristique soit placée où il le souhaite. Mais cette solution n'est pas encore très conviviale. En effet, si on désire placer un cube de manière à ce que trois de ses faces fassent le même angle avec l'horizontale, il est très difficile de connaître la valeur de cet angle, pour spécifier les rotations à réaliser. En outre, il serait intéressant de pouvoir positionner une caractéristique *sur* ou *contre* une autre, donc de préciser une face de contact.

Plus généralement, la solution que nous jugeons la plus confortable pour l'opérateur est de pouvoir positionner et orienter la caractéristique en fixant un certain nombre de contraintes, qui rendront automatique et transparente la gestion du repère local. Pour cela, nous prévoyons de fournir à l'opérateur une couche de placement indépendante [Kla 96], qui permet par exemple d'identifier des faces de la caractéristique et des objets de référence et d'établir des contraintes entre elles (distance, angle...). Cependant, pour identifier des faces, il faut que ces faces soient

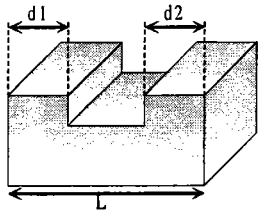
visualisées. Or le volume utile de la caractéristique n'est pas encore connu à cet instant de l'instanciation ; son B-Rep ne peut donc pas être affiché. Par contre, la surface génératrice de la caractéristique générique est connue. Mais, comme nous l'avons vu précédemment, il se peut que cette surface soit infinie. Or il n'est pas facile de visualiser une surface infinie. C'est pourquoi, au moment du placement, nous choisissons de visualiser une icône de la caractéristique, dont la forme correspond globalement à la génératrice, mais dont les surfaces sont finies. Des paramètres par défaut sont nécessaires pour donner des valeurs de dimensions et pour limiter les faces ouvertes. L'icône ne correspondant pas au volume utile, il peut sembler ne pas déboucher où il faut (caractéristique négative) ou ne pas atteindre le porteur (caractéristique positive). Mais ce n'est qu'un problème de représentation graphique temporaire, qui sera résolu par le calcul du volume utile.

En ce qui concerne les dimensions de la caractéristique, le moyen le plus simple pour les renseigner est de donner des valeurs numériques explicites. Cependant, il n'est pas toujours aisé pour l'opérateur de fournir de telles quantités, puisque ceci implique qu'il connaisse les dimensions des objets voisins et plus particulièrement du porteur pour les caractéristiques secondaires. De plus, certaines dimensions exactes ne sont pas toujours indispensables. En effet, elles peuvent quelquefois être déduites ; c'est le cas par exemple de la hauteur d'un trou traversant. Cette dimension est donc inutile, du moment que le trou débouche de part et d'autre de l'objet qui le porte, et ne doit donc pas être un paramètre du trou. Ceci justifie le fait de représenter les caractéristiques par des demi-espaces : le trou étant uniquement représenté par une surface cylindrique, son seul paramètre est le rayon. Cette représentation permet donc de minimiser le nombre de paramètres et l'opérateur n'aura donc à préciser que ceux qui sont réellement significatifs.

En outre, les dimensions ne sont pas toujours connues de manière exacte par l'opérateur. Il doit donc pouvoir en donner une valeur approchée (à la souris par exemple avec un affichage élastique) et éventuellement la préciser par la suite. De plus, il faut autoriser que certains paramètres restent non renseignés, c'est-à-dire que même une valeur approchée n'est pas indispensable. Or, pour réaliser une instanciation, tous les paramètres sont nécessaires. La solution est de prévoir des valeurs par défaut pour ces paramètres laissés vacants. Cependant, il n'est pas simple de déterminer des valeurs par défaut, en fonction du contexte. Elles sont calculées en fonction des dimensions de l'espace de travail et si possible même en fonction des dimensions des objets déjà instanciés. Ceci garantirait du même coup une certaine validité pour ces valeurs. Par exemple, si un trou est créé sans que son diamètre soit précisé, la valeur par

défaut doit être telle que le trou reste dans les limites de son porteur. Nous reviendrons sur ces problèmes de validité au paragraphe 3.1.4.

Pour éviter de donner des valeurs précises aux paramètres, qu'ils soient de dimensions, de position ou d'orientation, nous pourrions les calculer automatiquement, s'ils sont liés à d'autres quantités par des contraintes indiquées à l'instanciation ou imposées au niveau générique. Un exemple de contrainte indiquée à l'instanciation est : le diamètre du trou à créer est égal à la moitié de la largeur de telle poche. Une contrainte générique pourrait être : le diamètre d'un trou est égal à telle fraction de sa profondeur. Remarquons que les paramètres de dimensions sont parfois déduits du positionnement. En effet, si une rainure en U est placée en fixant des contraintes de distance entre ses faces latérales et les faces de sont porteur, comme l'indique la figure 2.8, sa largeur est fixée automatiquement.



largeur de la rainure *l* fixée
par les positions d1 et d2
l = L - (d1 + d2)

Figure 2.8 : Largeur de la rainure fixée par son placement.

Nous distinguons deux types de contraintes pour déterminer certains paramètres, selon leur formulation : elles sont dites déclaratives ou procédurales.

Les contraintes déclaratives correspondent à des équations faisant intervenir des paramètres d'objets (diamètre, hauteur...) ou des quantités dépendant de ces mêmes paramètres (volume, débit, distance...). De manière générale, elles sont résolues par un solveur. Cependant, l'utilisation de fonctions prédéfinies, recherchant la solution à certaines combinaisons de contraintes, peut se révéler plus efficace. En phase de dialogue, leur acquisition peut se faire sous la forme d'expressions grapho-numériques, mêlant des opérateurs numériques, des paramètres et des objets graphiques (le rayon est égal au double du paramètre "largeur" de l'objet que je montre) [GJL 95].

Les contraintes procédurales correspondent à des énoncés du type : le perçage (que l'on suppose orthogonal à la face de départ) est tel qu'il s'arrête dès qu'il débouche entièrement sur une même face plane, elle aussi perpendiculaire à son axe (figure 2.9). Ces contraintes ne peuvent pas être exprimées sous forme d'équations, leur résolution passe donc par l'exécution d'un algorithme.

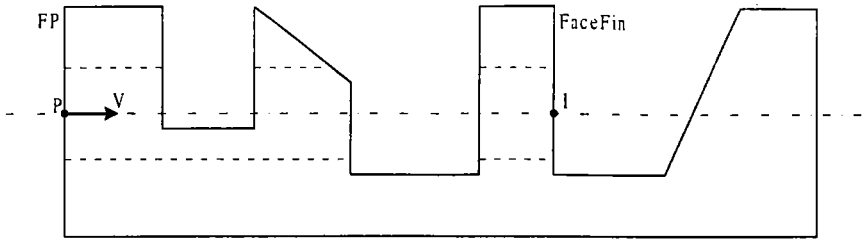


Figure 2.9 : Exemple de perçage (en coupe).

Dans le cas du trou ci-dessus, nous pouvons envisager l'algorithme suivant :

```
Soit (P,V) l'axe du trou, D son diamètre et FP la face sur laquelle on commence le perçage
(P ∈ FP) :
h = +∞ ; Normer (V) ;
Pour chaque face F du volume autre que FP faire
   Si F est plane alors
      Si Normale extérieure (F) = + V alors        /* F est parallèle à FP et est bien
                                                      orientée d'un point de vue matière */
         I = intersection de F et de la droite (P,V)
         Si I existe alors
            soit k tel que I = P + k.V             /* k = distance signée de FP à F */
            Si (k > 0) et (k < h) alors            /* F est plus proche de P que la
                                                      solution précédente */
               Si (toute arête de F est à une distance de I supérieure à D/2) alors
                                                   /* le trou ne débouche pas à la fois
                                                      sur F et une autre face */
                  FaceFin = F ; h = k ;
               Fsi
            Fsi
         Fsi
      Fsi
   Fsi
Fpour
```

3.3. Calcul de la forme

L'objectif de cette étape est de calculer la forme de la pièce après insertion de la caractéristique. Comme l'analyse bibliographique et les remarques préliminaires le suggèrent, nous souhaitons également disposer du volume utile de la caractéristique (volume de matière ajouté ou retiré par la caractéristique).

Il est évident qu'il ne faut pas une méthode d'instanciation par caractéristique. Le mécanisme que nous avons prévu ne distingue que trois sous-cas : celui des caractéristiques primaires, celui des secondaires positives et celui des secondaires négatives. Il paraît difficile d'en gérer un nombre inférieur.

Dans un premier temps, la surface génératrice de la caractéristique instanciée, correctement dimensionnée et positionnée, est obtenue par exécution de la procédure de création de la forme.

Pour une caractéristique primaire, la génératrice correspond exactement au volume utile de la caractéristique. Les deux objectifs de l'étape sont donc atteints et l'instanciation est donc terminée.

Pour une caractéristique secondaire, il reste à combiner la génératrice avec le porteur pour obtenir la forme finale et à calculer le volume utile. Dans les faits, comme le volume utile est impliqué dans la mise à jour de la forme, son calcul se fait d'abord. Nous détaillons donc ces deux étapes ci-dessous.

Le volume utile est obtenu par une opération booléenne entre la génératrice et le volume porteur. L'opération est une intersection dans le cas d'une caractéristique négative (génératrice ∩ porteur) et une différence dans le cas d'une positive (génératrice – porteur). De cette opération booléenne peuvent résulter plusieurs volumes. La génératrice se prolongeant dans certaines directions, il se peut que certains des volumes obtenus ne doivent pas être associés au volume utile. Sur la figure 2.10, seuls les volumes V1 et V2 correspondent effectivement au rainurage souhaité. Le choix du ou des volume(s) correspondant au volume utile sera vraisemblablement automatique, grâce à des règles, ou à défaut pourra être guidé par l'opérateur.

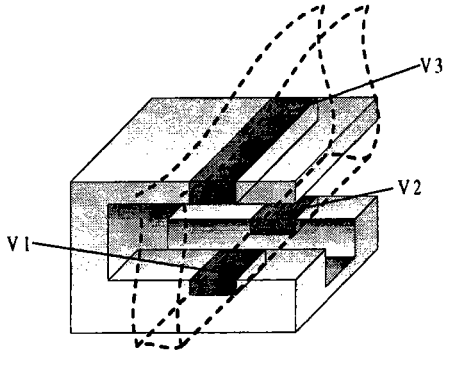


Figure 2.10 : La génératrice interfère plusieurs fois avec le porteur.

Une difficulté supplémentaire est posée par les caractéristiques positive car l'opération booléenne peut produire un volume infini. En effet, on réalise la différence entre la génératrice et le porteur, mais si la génératrice dépasse le porteur, le résultat sera un volume infini, par exemple, lorsqu'on instancie une languette, comme le montre la figure ci-dessous.

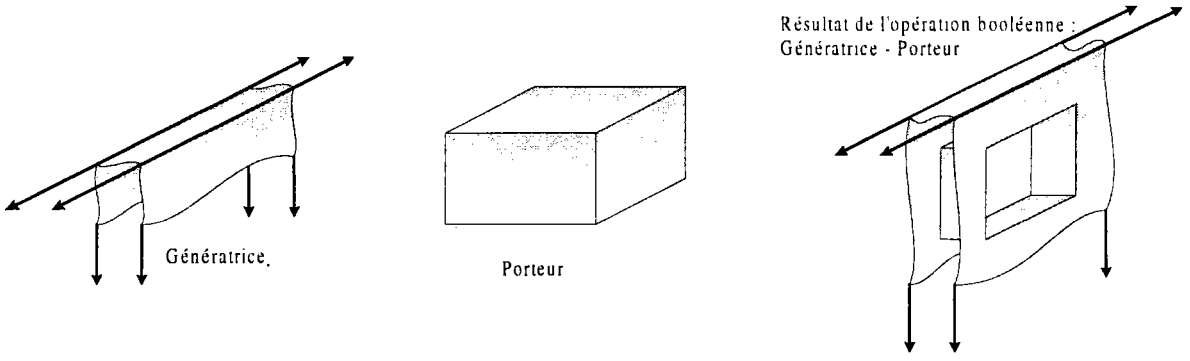


Figure 2.11 : Instanciation d'une languette.

Pour résoudre ce problème, nous proposons de limiter la génératrice de la caractéristique, par des faces de son volume porteur, ou plus exactement par les surfaces qui portent ces faces. Dans l'exemple de la languette, la génératrice peut être limitée avec les faces avant, arrière et du dessous du bloc porteur, ce qui donne le volume de la figure 2.12. Cette limitation n'est pas réalisée de façon automatique ; c'est à l'opérateur d'indiquer les faces limites. Cette méthode a été proposée par [ChH 95] et est décrite en détail dans le chapitre 1 (paragraphe 3.1.1).

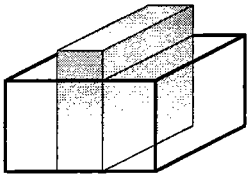


Figure 2.12 : Limitation de la génératrice par le porteur.

Lorsque la différence entre ce nouveau volume limité et le volume porteur est calculée, on obtient un volume fini, qui correspond au volume utile de la languette. Remarquons que sur cet exemple, si la génératrice n'avait été limitée que par deux faces (avant et arrière), le résultat de l'opération booléenne aurait donné deux volumes. L'un étant fini et l'autre infini, on aurait pu en éliminer un : le volume utile de la languette est le volume fini.

Le problème qui se pose alors est la mémorisation du (des) volume(s) choisi(s). En effet, si l'intervention de l'opérateur n'est pas exclue lors de la première instanciation, il est évident qu'on ne peut pas la réexiger au cours de l'évolution du modèle de conception. Nous proposons d'établir des critères de choix en fonction des faces du porteur dans lesquelles les volumes débouchent. Lors de nouvelles instanciations, les faces devront rester les mêmes. Nous pouvons également éliminer les volumes qui divisent le porteur en plusieurs composantes connexes. Ainsi, sur la figure 2.10, nous rejetterons le volume n°3. Il faut cependant s'assurer que ce critère de choix ne contredit pas les souhaits du concepteur. Une étude plus approfondie de ce problème est donc nécessaire et constitue une des perspectives de notre travail.

Le volume utile étant calculé, la forme finale est obtenue en l'ajoutant au volume porteur dans le cas d'une caractéristique positive et en le soustrayant dans le cas contraire. Nous montrons donc de nouveau que le fait de conserver le type de la caractéristique (positif ou négatif) est utile au moment de l'instanciation.

L'algorithme ci-dessous résume les étapes du calcul de la forme de la pièce après insertion d'une caractéristique secondaire (exemple de la poche).

```
// calcul de la génératrice en fonction des paramètres de dimensions et de position
Génératrice ← CréerPoche (Largeur, Longueur, RepèreLocal)

// calcul du volume utile par intersection entre la génératrice et le volume porteur
VolumeUtile ← Intersection (Porteur, Génératrice)

// évaluation de la pièce finale en soustrayant le volume utile de la poche au porteur
Pièce ← Différence (Porteur, VolumeUtile)
```

Remarquons que dans la seconde opération booléenne, les opérandes possèdent des faces superposées, souvent causes d'échec dans les algorithmes d'opérations booléennes. Celles que nous utilisons ont fait l'objet d'une formalisation rigoureuse [Per 95], qui a permis de développer un algorithme robuste et fiable, prenant en compte les faces coplanaires. La méthode est basée sur les faces des deux objets opérandes, ce qui permet de réduire la complexité à un problème bidimensionnel.

3.4. Vérification de validité

Lorsque le volume utile est calculé, l'instanciation doit déclencher, si elles existent, les fonctions qui testent la validité de la caractéristique. Si les règles de validité sont exprimées par des contraintes, il suffit de les vérifier, ce qui est du ressort du gestionnaire de contraintes. Si elles ne sont pas toutes satisfaites, il faut les résoudre et proposer une nouvelle solution. Devant l'ampleur des difficultés posées par la gestion des contraintes, nous n'avons pas voulu, dans le cadre de cette thèse, traiter le problème dans son intégralité. Actuellement, la résolution se fait de manière manuelle et la solution est alors fournie au système. L'objectif, à terme, est toutefois de pouvoir interfacer le modeleur à base de caractéristiques avec un gestionnaire de contraintes. Des travaux sur le sujet sont en cours dans notre laboratoire et une thèse est en préparation [Lei 97].

Les règles de validité peuvent également porter sur la vérification de la validité topologique. Une de ces règles pourrait être que chaque face de la génératrice doit engendrer une face au moins dans le volume qui porte la caractéristique. Si l'une manque, la caractéristique n'est pas

valide. Pour opérer cette vérification topologique, nous utilisons un processeur d'opérations booléennes développé au laboratoire [Per 95] et adapté pour fournir une liste de provenances de faces, c'est-à-dire que pour chaque face créée, il donne la liste des faces des volumes opérandes dont elle est issue.

L'adaptation du module d'opérations booléennes à notre situation nous en rend dépendant. En effet, pour réaliser la vérification que nous venons de décrire, nous ne pouvons pas utiliser un module quelconque, qui ne fournit pas cette liste de provenances. Cependant, nous avons opté pour l'adaptation afin d'exploiter au maximum les informations dont nous pourrions avoir besoin.

Un raisonnement du même type a vraisemblablement été mené pour le développement du système EXTDesign [LaM 93], bien que l'adaptation des opérations booléennes soit cette fois exploitée comme un filtre important avant une extraction incrémentale de caractéristiques. De manière générale, ce type d'optimisation est réalisable lorsque l'on développe ses propres outils. Comme le soulignent Mandorli *et al.* [MCO 97] il l'est beaucoup moins lorsque l'on utilise des composants logiciels du marché.

Si la caractéristique n'est pas valide, il existe deux solutions pour résoudre le problème :

- les paramètres sont ajustés automatiquement ou corrigés par l'opérateur si un paramètre non valide lui a été signalé ;
- la caractéristique non valide est transformée automatiquement en une caractéristique dérivée, dont il faut calculer les paramètres (dimensions, position, orientation). Par exemple, un trou borgne peut devenir débouchant, comme le montre la figure 2.13.

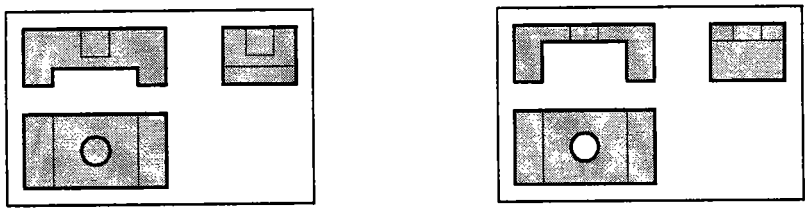


Figure 2.13 : Trou borgne devenant débouchant.

Dans le premier cas, pour ajuster automatiquement les paramètres qui ne sont pas valides, la caractéristique doit être munie des règles qui déterminent la nouvelle valeur pour le paramètre erroné, par exemple une valeur par défaut. Or, il est impossible d'avoir des valeurs par défaut pour tous les paramètres de toutes les caractéristiques au niveau générique, car ces valeurs dépendent du contexte. En effet, un paramètre donné d'une caractéristique peut être invalide simplement parce que son porteur est trop petit. Avec un porteur plus grand, la même valeur pour le paramètre en question peut être valide. Les valeurs par défaut doivent donc être calculées en

fonction de l'instanciation, au moment de la vérification de validité. Elles peuvent être déterminées en fonction d'autres dimensions, à l'aide d'équations. Quelques propositions concernant la validité des paramètres de certaines caractéristiques ont été faites dans [Poi 94].

Dans le second cas, la caractéristique doit connaître les conditions de dérivation en une autre caractéristique et posséder des règles qui fournissent les dimensions, la position et l'orientation de la caractéristique dérivée. Par exemple, si un trou borgne devient débouchant à la suite de l'instanciation d'une rainure (exemple de la figure 2.13), il faut pouvoir déterminer les paramètres du trou débouchant, en fonction de ceux du trou borgne. Ceci peut se faire à l'aide de règles du type :

Si *ConditionDérivation* alors
  caract1 dérive en **caract2**
  paramètre_21 fonction *f* de certains paramètres de **caract1**
  paramètre_22 fonction *g* de certains paramètres de **caract2**
  ...
Fsi

où *paramètre_ij* représente le $j^{ème}$ paramètre de la $i^{ème}$ caractéristique.

Il faut alors réexécuter la procédure de vérification de validité avec la nouvelle caractéristique dérivée et ses nouveaux paramètres.

Remarquons que l'on ne peut pas effectuer cette dérivation de façon automatique, sans en demander confirmation à l'opérateur. En effet, la nature de la caractéristique étant changée, il faut s'assurer que ce changement traduit bien l'intention de conception. Une difficulté se pose alors : comment gérer les contraintes qui sont associées à une caractéristique qui doit être dérivée. Il faudrait pouvoir garantir que si, par suite de nouvelles modifications dans le modèle de conception, la première caractéristique (avant dérivation) redevient valide, les contraintes qui lui étaient appliquées sont de nouveau vérifiées.

Le problème de la validité dépasse le cadre du présent travail ; les propositions faites sont donc relativement sommaires et n'ont pas été mises en œuvre. Une réflexion approfondie à ce propos est néanmoins nécessaire et fera l'objet d'une perspective de recherche.

3.5. Insertion dans le modèle de conception

À la fin de l'instanciation, quand la validité est vérifiée, il faut enregistrer la caractéristique dans une structure adaptée, que nous avons appelée le modèle de conception. On pourra ainsi la modifier plus tard ou s'en servir pour y appuyer de nouveaux objets. Le modèle de conception

doit donc permettre de mémoriser toutes les informations nécessaires à la conservation de l'intention de conception (*design intent*), mais aussi les informations utiles pour la visualisation, la cotation... Ces diverses informations sont d'un niveau sémantique différent : l'intention de conception est une information de haut niveau, les informations géométriques et topologiques sont de plus bas niveau. Ceci suggère donc une structuration du modèle en deux couches : celle de haut niveau mémorise un historique complet de la conception et celle de bas niveau est une représentation évaluée de type B-Rep. Il existe des liens entre ces deux couches qui permettent de maintenir la cohérence globale. Remarquons que cette structuration en deux couches de niveaux sémantiques différents est souvent adoptée pour un modèle produit, comme nous l'avons souligné dans le chapitre 1 (paragraphe 3.2.5).

Ainsi, dans le modèle que nous avons analysé et développé, la première couche permet de conserver l'historique de construction. Une scène est donc un ensemble de volumes, eux-mêmes constitués de composants de différentes natures. Les composants d'un volume sont soit des caractéristiques, soit des opérations booléennes, soit des transformations géométriques, ce qui rappelle les nœuds d'un arbre CSG. C'est donc une hiérarchie, même si la représentation schématique semble linéaire. Les caractéristiques peuvent être liées entre elles par des contraintes géométriques. Ces relations sont maintenues grâce aux graphes conceptuels des caractéristiques, par des arcs entre les nœuds correspondants aux faces des différentes caractéristiques. Il s'agit des contraintes indiquées à l'instanciation par l'opérateur, de celles du niveau générique et éventuellement de contraintes déduites. L'ensemble des contraintes mémorisées dans la première couche du modèle de conception permet donc de prendre en compte l'intention de conception. En effet, elles assurent le respect de ce qui a été spécifié par le concepteur, lors de la modification d'une valeur et des répercussions de cette modification.

Chaque volume est en outre représenté par un B-Rep, résultat de la combinaison de tous ses constituants ; ce B-Rep forme la seconde couche du modèle. Des liens entre les deux couches relient les nœuds des graphes aux faces correspondantes dans le B-Rep. La structure du modèle de conception est détaillée dans l'annexe B.

En conclusion, nous venons de vérifier que les caractéristiques de forme, modélisées de la manière décrite au paragraphe 2, peuvent être instanciées. Le choix de la caractéristique ne pose pas de problème particulier. La mémorisation des contraintes d'instanciation est réalisée à l'aide du graphe conceptuel. Le volume utile est calculé en réalisant une opération booléenne entre la génératrice et le porteur. Des propositions ont également été faites pour vérifier la validité de la

caractéristique instanciée. Enfin, l'insertion dans le modèle de conception est réalisée par une seconde opération booléenne, entre le volume utile et le porteur.

4. AUTRES TRAITEMENTS RELATIFS AUX CARACTÉRISTIQUES

Comme nous venons de le voir, une partie du comportement associé aux caractéristiques permet leur insertion dans le modèle de conception. La définition d'un comportement plus élaboré augmente leur intérêt lors de traitements que l'on souhaite réaliser sur le modèle à base de caractéristiques, comme nous l'avons évoqué dans le paragraphe 3.4.3 du chapitre 1. Nous faisons ici un survol des possibilités d'extension.

Nous envisageons d'abord de munir la caractéristique d'un certain nombre de règles lui permettant de s'identifier dans un environnement qui n'est pas composé uniquement de caractéristiques explicites. Ces règles sont donc des règles d'extraction "automatique".

De plus, nous avons déjà vu l'intérêt de pouvoir déduire facilement, si ce n'est automatiquement, un modèle spécifique à une application donnée, à partir du modèle de conception. Cette transformation du modèle de conception vers divers modèles d'applications est fréquemment appelée le *mapping*.

Ces deux points font l'objet des deux paragraphes suivants. Nous aborderons ensuite brièvement la possibilité de définir des motifs de caractéristiques et d'étendre la classification des caractéristiques que nous proposions en 2.1, à des caractéristiques qui ne seraient ni positives, ni négatives.

4.1. Extraction automatique de caractéristiques de forme

Dans le modèle de conception, il peut se produire une apparition de certaines caractéristiques à la suite d'opérations de modélisation géométriques (combinaison booléenne), sans qu'elles soient créées explicitement. Si l'on réalise une union entre trois boîtes adjacentes de manière à construire une rainure (voir figure 2.14), il serait commode, du point de vue utilisateur, de pouvoir ensuite intervenir sur les dimensions de la rainure. Cette dernière n'étant pas connue en tant que telle, il faut d'abord la détecter puis retrouver les valeurs de ses paramètres (dimensions, position, orientation).

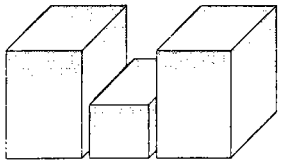


Figure 2.14 : Apparition non explicite d'une rainure.

Pour détecter la présence d'une caractéristique dans le modèle de conception, on peut exploiter le fait que les caractéristiques sont représentées par un graphe conceptuel. Une première méthode d'extraction consiste, en effet, à rechercher des occurrences du graphe conceptuel de la caractéristique dans le graphe de la pièce complète. Une seconde méthode consiste à exécuter des procédures d'extraction générées automatiquement à partir des graphes conceptuels des différentes caractéristiques. Le principe est toujours de réaliser un appariement entre les nœuds du graphe conceptuel de la caractéristique à extraire et les faces du B-Rep de la pièce. Lorsqu'une correspondance est établie, il faut vérifier que les contraintes exprimées entre les nœuds du graphe sont satisfaites dans le B-Rep. Dans l'affirmative, l'extraction est terminée, dans le cas contraire, elle a échoué.

Une fois qu'une caractéristique est détectée, il faut retrouver les valeurs de ses paramètres. Or, il n'est a priori pas simple d'établir la correspondance entre les cotes de la rainure extraite, par exemple, et les paramètres d'une rainure. Nous rappelons toutefois que certaines contraintes du graphe conceptuel sont précisées par des paramètres (voir paragraphe 2.1.2). Comme, lors de la détection, les géométries du graphe conceptuel ont été associées à des faces du B-Rep, les faces correspondantes peuvent être retrouvées à partir des contraintes. Ainsi, dans l'exemple de la rainure, le paramètre "largeur" est associé à une contrainte de distance entre deux faces. Il est donc aisé de retrouver la valeur de la largeur qui est donc égale à la distance en question.

Malgré ces quelques propositions, il n'existe aucun algorithme à la fois fiable et général d'extraction de caractéristiques de forme [Min 96]. Une gestion systématique de l'apparition non explicite de caractéristiques apparaît donc difficile. Cependant, nous pouvons citer trois idées pour améliorer les performances : diminuer le nombre d'extractions, réaliser une reconnaissance partielle en n'analysant que les endroits ayant subi des modifications, proposer un algorithme d'extraction et des règles pour atténuer l'explosion combinatoire.

Le premier point consiste à prévoir quelle caractéristique est engendrée dans une opération booléenne faisant intervenir une caractéristique donnée. Il est par exemple vraisemblable que le complément d'une languette est une rainure. Comme l'illustre la figure 2.15, ce type de déduction a cependant ses limites.

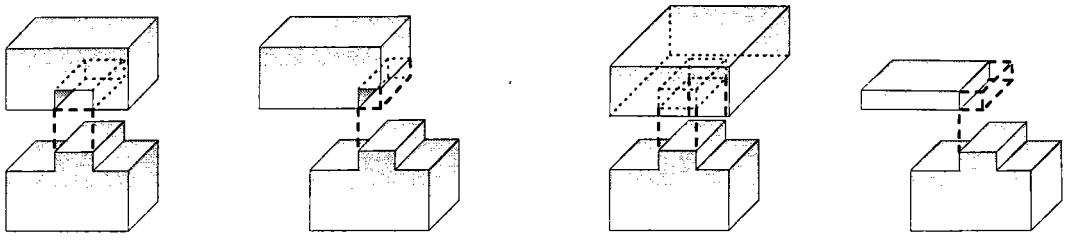


Figure 2.15 : Le complément de la languette est une rainure, une marche, une poche ou rien de particulier.

Pour améliorer les performances des algorithmes d'extraction, il ne faudrait pas traiter la totalité du modèle de conception à chaque modification du modèle géométrique, c'est-à-dire à chaque fois qu'une nouvelle extraction est nécessaire. Cette approche, connue sous l'appellation d'extraction incrémentale, a été proposée notamment par [LaM 93] et [PaK 96]. Les deux méthodes sont étudiées et comparées dans [Min 96]. Il en ressort qu'il est difficile de comparer les performances de ces méthodes, mais qu'elles sont meilleures que les approches non incrémentales. En effet, elles optimisent toutes deux l'espace de recherche, l'une en supprimant les faces des caractéristiques reconnues au cours d'une précédente extraction, l'autre en isolant par une opération booléenne les zones où l'objet a évolué.

La troisième proposition pour rendre l'extraction des caractéristiques plus ou moins automatique, est de proposer un algorithme qui exploite la connaissance de la surface génératrice de toutes les caractéristiques. En effet, la présence d'une caractéristique dans le modèle géométrique peut être détectée, si on arrive à apparier les faces de la génératrice avec un certain nombre de faces du modèle. Il faut cependant tenir compte du côté de la matière, pour faire la différence entre les caractéristiques positives et négatives dont la génératrice est semblable (par exemple rainure et languette). La figure ci-dessous donne des exemples d'appariement de trois faces du modèle avec la génératrice de la rainure.

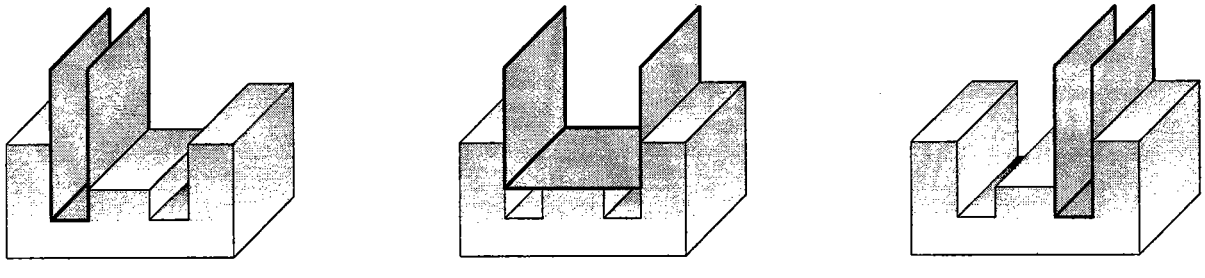


Figure 2.16 : Extraction de trois rainures.

Nous pouvons réaliser un filtre à cette recherche en examinant l'historique de construction de l'objet. En effet, la recherche peut être orientée en fonction des opérations booléennes qui ont été réalisées. Si une différence apparaît dans l'historique, on cherchera à reconnaître dans le second opérande une caractéristique négative, si ce second opérande est un volume primitif. Par contre,

si on rencontre une union, on recherchera une caractéristique positive. Cependant, ce filtre a ses limites puisqu'il n'est efficace que si l'historique correspond à l'opération réalisée lors de l'instanciation de la caractéristique. En particulier, l'historique de la figure 2.14 n'incluant pas de différence, ne permet pas la détection d'une rainure. Pour pallier ce problème, nous pourrions normaliser l'historique, en remplaçant les unions par des différences. Des travaux de normalisation d'arbres CSG en arbre DSG (*Destructive Solid Geometry*) ont été effectués dans cet esprit [PCL 90]. Nous pourrions nous en inspirer pour normaliser notre modèle sémantique. Remarquons que nous ne faisons ici que quelques propositions générales, qui n'ont pas fait l'objet d'une réflexion plus approfondie. Une étude plus détaillée est donc nécessaire.

Le principal intérêt d'exploiter la génératrice pour réaliser l'extraction est de pouvoir appliquer la même méthode à n'importe quelle caractéristique définie par une génératrice. Ce côté automatique évite l'écriture d'algorithmes d'extraction spécifiques, pour chaque caractéristique. C'est un atout indéniable, en particulier lors de l'enrichissement de la bibliothèque. En effet, la méthode évite à l'utilisateur d'avoir à programmer une procédure d'extraction, lorsqu'il définit de nouvelles caractéristiques.

La méthode d'extraction proposée se heurte toutefois au problème de la combinatoire, qui peut être très élevée. Considérons l'exemple de trois faces apparaissant dans le modèle géométrique, deux d'entre elles étant parallèles et la troisième perpendiculaire et adjacente aux deux autres. Cette configuration peut être appariée avec toutes les génératrices des caractéristiques dont la forme correspond plus ou moins à un parallélépipède. Or c'est le cas de nombreuses caractéristiques usuelles (rainure, poche, languette...). Plutôt que de fournir une seule solution, nous proposons donc de donner, en résultat de l'extraction, une liste des candidats possibles. Le choix de la solution ad hoc parmi les candidats pourra être laissé aux soins de l'opérateur. Cette combinatoire élevée peut être exploitée pour proposer des alternatives selon les points de vue. En effet, selon le contexte de l'application, une solution pourra être privilégiée par rapport aux autres. Nous reviendrons sur l'intérêt de l'extraction dans le cadre de la transformation du modèle de conception en modèle d'application au paragraphe suivant.

4.2. Génération de modèles d'application

Le modèle de conception est la maquette virtuelle issue de la conception. La structure évoquée auparavant, dans laquelle sont recensées les caractéristiques instanciées, n'en est qu'une facette. De manière idéale, il contient également l'historique complet de la conception et, plus

généralement, l'ensemble des contraintes qui lient les composants de la pièce finie. Parmi ces contraintes, il y a celles qui portent sur des paramètres de caractéristiques, les liant ainsi à d'autres caractéristiques ou à d'autres entités du modèle (faces, arêtes, axes de symétrie...).

Le modèle de conception contient donc toutes les informations nécessaires, mais il n'est pas directement exploitable par les applications, c'est-à-dire par toute activité en parallèle ou en aval de la conception. C'est pourquoi il faut pouvoir transformer, de manière aussi automatique que possible, le modèle de conception en modèles d'application, dans lesquels seules les informations utiles à une tâche donnée sont conservées.

Le fait qu'il soit possible de transformer le modèle de conception en modèles d'application est une façon de valider la conception et donc d'envisager l'ingénierie simultanée. Par exemple, il est intéressant de pouvoir obtenir le modèle de fabrication par usinage, avec calcul du brut, de l'ordonnancement des opérations, des outils nécessaires, du nombre de passes et des trajectoires d'approche et de travail.

Il n'est cependant ni réaliste, ni raisonnable d'espérer effectuer ces transformations en permanence. Ce n'est pas réaliste car elles sont coûteuses et pas toujours maîtrisées ; ce n'est pas raisonnable car la conception passe forcément par des phases non finies au cours desquelles elle n'est pas valide. Comme il semble difficile de tester automatiquement si la conception a atteint un niveau cohérent, il paraît judicieux de n'opérer les transformations qu'à la demande du concepteur ou à des moments clés déterminés par des règles prédéfinies.

Il faut aussi prévoir d'éventuels retours d'un modèle d'application vers le modèle de conception et d'en faire bénéficier les autres modèles d'applications. Si par exemple l'étude par éléments finis remet la dimension d'une rainure en cause, il faut d'une part corriger le modèle de conception et d'autre part mettre à jour les trajectoires d'outils, dans le modèle de fabrication. Les modèles d'applications doivent donc pouvoir communiquer, éventuellement via le modèle de conception.

La connaissance explicite des caractéristiques présentes dans la pièce est un avantage dans la phase de dérivation du modèle de conception en un modèle d'application.

C'est par exemple le cas dans le cadre d'une fabrication par enlèvement de matière : les caractéristiques de forme suggèrent directement une partie de la gamme d'usinage. Par exemple, la présence d'une rainure suggère le passage d'une fraise. Les tolérances et le matériau aident au choix des outils et du nombre de passes.

C'est aussi le cas lors d'un maillage par éléments finis qui doit permettre d'étudier le comportement de la pièce sous l'effort. Dans ce type de traitement, il est primordial, pour des

raisons de performances, de simplifier l'objet à mailler. La connaissance des caractéristiques et de leurs dimensions offre justement le moyen de savoir si elles ont une taille significative à l'échelle à laquelle on travaille et donc s'il faut les supprimer ou raffiner le maillage dans leur voisinage.

Nous pouvons également exploiter le fait que les caractéristiques sont représentées par un demi-espace pour la transformation des caractéristiques secondaires négatives en caractéristiques d'application, particulièrement pour l'usinage. La génératrice qui représente la caractéristique permet de déterminer à la fois la forme de l'outil et la trajectoire d'approche. Les faces de la génératrice sont considérées comme les faces à usiner. Les faces virtuelles, c'est-à-dire les faces qu'il faudrait ajouter à la génératrice pour en faire un volume, permettent de déduire une trajectoire d'approche de l'outil possible.

Nous soulignons aussi la possibilité d'utiliser l'extraction automatique de caractéristiques, pour réaliser la transformation du modèle de conception en modèle d'application. En effet, nous avons vu que cette extraction peut tenir compte du point de vue, notamment lorsqu'il y a plusieurs solutions, le contexte peut aider au choix de la solution retenue. Pour réaliser la transformation on peut donc exécuter une extraction dans un contexte d'application précis.

4.3. Motifs de caractéristiques

Parmi les difficultés survenant lors de la gestion d'un modèle à base de caractéristiques, on peut également rencontrer le problème de l'altération imprévue mais volontaire d'une ou plusieurs caractéristiques. Par exemple, il se peut que l'on répète automatiquement selon un certain motif une caractéristique donnée. Ne connaissant pas à l'avance le nombre de répétitions, l'opérateur en crée volontairement trop. Les caractéristiques excédentaires seront tôt ou tard partiellement ou totalement détruites ce qui n'implique pas forcément qu'elles doivent l'être dans la structure. En effet, l'intention de l'opérateur a été de reproduire la caractéristique aussi loin que nécessaire et s'il revient sur l'opération qui a détruit certains exemplaires, il faut pouvoir les restituer.

En fait, la situation est sensiblement la même que lors de la propagation de contraintes où, lorsqu'une valeur change, il faut établir la liste des entités affectées, la liste des contraintes qui les déterminent et satisfaire ces dernières. Les contraintes sont simplement moins explicites dans le cas présent ; lorsque l'opération qui altère un certain nombre de caractéristiques a lieu, le

système doit mémoriser l'existence d'une dépendance entre le motif de caractéristiques et l'objet mis en cause dans l'opération. Ainsi sait-on que l'objet a une influence sur le motif (et vice versa). Au moment de sa modification, le motif fera partie de la liste des entités affectées et en exploitant son historique de création, il pourra être mis à jour.

4.4. Extension de la classification proposée pour les caractéristiques

Nous rappelons que, comme de nombreux systèmes étudiés dans la bibliographie, nous avons séparé les caractéristiques positives des négatives, dans notre taxinomie. L'argument fondamental qui justifie ce choix est que la nature de la caractéristique induit le type des opérations booléennes réalisées lors de l'instanciation. La figure 2.17 rappelle cette classification.

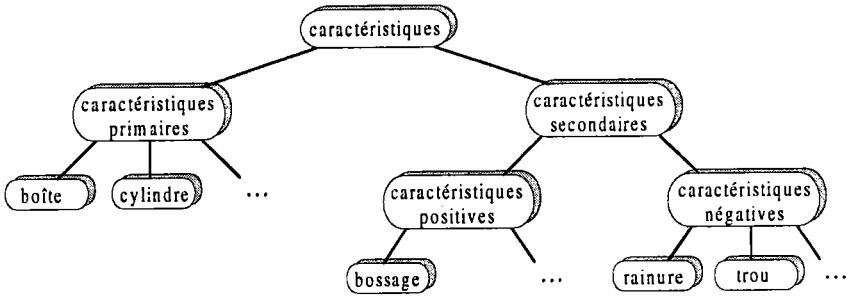


Figure 2.17 : Hiérarchie des caractéristiques.

Le problème d'une telle hiérarchie est l'impossibilité d'y placer une caractéristique qui à la fois ajoute et retire de la matière (figure 2.18). Nous pouvons le surmonter en considérant qu'une caractéristique qui n'est ni positive ni négative peut être décomposée en d'autres caractéristiques qui elles sont positives ou négatives. Dans la hiérarchie, nous définissons alors un troisième type de caractéristiques, au même niveau que positives et négatives, c'est-à-dire dans les caractéristiques secondaires, que l'on nommera caractéristiques composées. L'instanciation d'une caractéristique composée sera réalisée en plusieurs étapes. Son volume utile sera en fait la liste des volumes utiles des caractéristiques qui la composent. Par exemple, la caractéristique de la figure 2.18 est créée en instanciant d'abord une rainure puis une languette. Son volume utile sera composé du volume de la rainure et de celui de la languette.

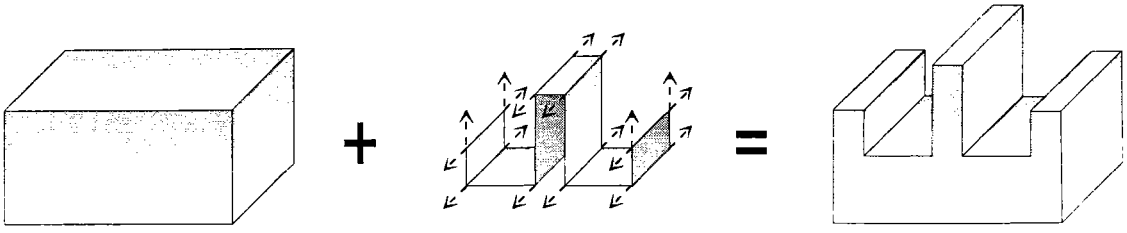


Figure 2.18 : Caractéristique ni positive, ni négative.

De même que pour l'instanciation, nous pouvons trouver une solution pour adapter tous les algorithmes permettant de définir le comportement de telles caractéristiques. En particulier, lors de l'extraction, nous proposons de rajouter une passe supplémentaire. La première étape consiste à détecter les caractéristiques simples d'abord, la rainure et la languette dans l'exemple cité. La seconde étape permet de reconstituer des caractéristiques composées à partir de toutes les caractéristiques simples extraites, en essayant de les combiner en fonction de leur position relative.

5. CONCLUSION

Dans la première partie de ce chapitre, nous avons proposé une modélisation des caractéristiques de forme qui permet de mémoriser toutes les informations nécessaires à leur définition. Toutes les caractéristiques sont représentées par la même structure, qu'elles soient prédéfinies ou définies par l'opérateur. Les principales informations qu'elles contiennent sont : la liste des paramètres, les contraintes génériques et d'instanciation, et les méthodes permettant de spécifier les comportements associés à la caractéristique tels que la description de la forme, la vérification de la validité...

La forme d'une caractéristique est déterminée par une surface génératrice qui présente plusieurs avantages, que nous n'avons pas retrouvé simultanément dans la bibliographie. Premièrement, elle uniformise la modélisation des caractéristiques. Deuxièmement, le nombre de paramètres nécessaires est minimisé par rapport à une représentation systématiquement volumique. Troisièmement, le prolongement de la caractéristique secondaire est automatique. En effet, on souhaite que la surface se prolonge jusqu'à atteindre les limites du porteur ; c'est le cas lorsque la génératrice est ouverte. Le dernier avantage est qu'elle permet le calcul du volume utile, dont la mémorisation est nécessaire pour réaliser plusieurs traitements de manière efficace :

- la préparation de la fabrication par usinage, car les volumes utiles des caractéristiques négatives correspondent aux delta-volumes, c'est-à-dire aux volumes de matière exacts à

usiner. Il faut toutefois réaliser un traitement sur les volumes utiles, lorsque les caractéristiques négatives interfèrent avec des positives. En effet, pour obtenir le volume de matière à usiner, il faut retirer les volumes utiles des caractéristiques positives, des volumes utiles des caractéristiques négatives ;

- la détection des interférences entre caractéristiques ;
- la gestion des modifications, qui est détaillée dans le chapitre suivant.

Nous distinguons deux types de caractéristiques : les caractéristiques génériques, qui contiennent les informations nécessaires pour générer des instances, et les caractéristiques d'instanciation, qui regroupent les données propres à chacune des instances du modèle de conception.

Dans une seconde partie, nous avons vérifié que cette modélisation des caractéristiques de forme était adaptée à leur utilisation dans le modèle de conception. Lors de l'instanciation, l'opérateur doit fournir un certain nombre de renseignements pour préciser la caractéristique qu'il veut insérer dans le modèle. La caractéristique d'instanciation est alors créée automatiquement avec ces renseignements, grâce à la caractéristique générique, puis insérée dans le modèle.

Enfin, une dernière partie a démontré que la modélisation des caractéristiques de forme qui a été proposée peut être exploitée pour d'autres traitements sur les caractéristiques. Il s'agit de l'extraction de caractéristiques de forme et de la transformation du modèle de conception en modèle d'application. Nous avons proposé une génération automatique de l'algorithme d'extraction, comme nous l'avions suggéré dans l'étude bibliographique. Cette génération automatique présente l'avantage de ne pas imposer à l'opérateur, lors de l'extension de la bibliothèque, de fournir des règles d'extraction pour les nouvelles caractéristiques.

Nous avons toutefois le sentiment que les principaux apports de notre approche se situent au niveau de la gestion des modifications et de l'aptitude à la description interactive de nouvelles caractéristiques par un opérateur non informaticien. Ces deux points font l'objet des chapitres 3 et 4.

Chapitre 3 - GESTION DES MODIFICATIONS

1. INTRODUCTION

Lorsque le modèle de conception est créé, l'opérateur peut revenir sur certaines décisions prises lors de l'instanciation des caractéristiques et vouloir réaliser des modifications. Elles peuvent porter sur des paramètres, sur des contraintes ou sur des caractéristiques entières (suppression). En ce qui concerne les paramètres, la modification peut concerner la valeur d'un paramètre numérique (de dimensions, position, orientation) ou bien le changement d'un paramètre géométrique (face de référence, axe...). La modification de contraintes peut être un ajout, une suppression ou bien un changement de valeur numérique ou encore d'entité concernée. Par exemple, une contrainte de distance peut être modifiée soit par la valeur de la distance, soit par la face de référence par rapport à laquelle la distance est établie.

Nous présentons tout d'abord l'ensemble des solutions que nous avons envisagées pour gérer les modifications. Nous détaillons ensuite les différentes étapes de celle que nous avons retenue. Enfin, nous justifions notre démarche, en montrant que toutes les étapes sont nécessaires et en donnant une preuve formelle de l'une de ces étapes.

2. SOLUTIONS ENVISAGÉES

Il existe plusieurs solutions pour gérer les modifications du modèle de conception. Comme il est constitué de deux couches, toute modification entraîne une mise à jour de la couche sémantique d'une part et du B-Rep d'autre part. La gestion des modifications est donc réalisée en deux étapes successives.

La première étape de la modification consiste à mettre à jour la caractéristique concernée dans la première couche, celle de plus haut niveau d'abstraction. La mémorisation des contraintes dans le modèle sémantique permet la propagation de la modification, en traitant les contraintes attachées à la caractéristique modifiée. Nous rappelons que la propagation des contraintes proprement dite est à la charge du gestionnaire de contraintes. Le modèle doit simplement permettre d'établir la liste des caractéristiques à modifier, qu'il est possible de trier dans l'ordre chronologique, mais il n'a pas à calculer les nouvelles valeurs des paramètres. En effet, elles sont mises à jour par la propagation de contraintes.

Le modèle permet ensuite de rechercher les interférences, au sens géométrique, entre les caractéristiques modifiées et les autres, car chacune dispose de son volume englobant. Nous calculons les intersections entre englobants, pour faire la liste de ce qui doit être modifié, alors que nous pourrions faire des tests d'interférence exacts, en utilisant les volumes utiles des caractéristiques, mais le gain serait minime. Se contenter des englobants représente tout au plus la mise à jour de quelques caractéristiques supplémentaires, mais le calcul d'intersection entre englobants est nettement plus efficace qu'entre deux volumes quelconques. La démarche complète de recherche des caractéristiques à mettre à jour est détaillée dans le paragraphe 3.1.

Dans un second temps, il faut répercuter la modification au niveau de la couche B-Rep, c'est-à-dire au niveau le plus bas. Il faut donc réévaluer la forme de la caractéristique modifiée et de toutes celles sur lesquelles il y a une incidence, c'est-à-dire celles qui ont été détectées à l'étape précédente.

Une première solution consiste à réexécuter en totalité l'historique des objets concernés, à chaque modification. Cette solution peut être envisageable aujourd'hui, malgré sa lourdeur, car les systèmes sont puissants et la réévaluation totale ne serait pas trop lente ; elle est d'ailleurs pratiquée dans certains systèmes, comme ProEngineer. Cependant, cette solution n'empêche pas la réévaluation de certaines parties de l'historique qui ne sont pas affectées par la modification,

notamment tout ce qui la précède dans l'ordre chronologique et qui n'interfère pas avec les caractéristiques modifiées (ni par les contraintes, ni géométriquement).

Pour l'optimiser, une seconde solution, sans doute préférable, consiste à réexécuter l'historique à partir du B-Rep dans l'état où il était avant l'instanciation de la première des caractéristiques à modifier. Pour obtenir ce B-Rep, nous proposons deux possibilités : la sauvegarde systématique du B-Rep à chacune de ses évolutions ou bien la désinstanciation à partir du B-Rep courant.

La première approche peut être réalisée de manière relativement peu coûteuse, grâce à un B-Rep incrémental qui optimise la sauvegarde de tous les états en mémorisant uniquement ce qui a changé entre deux étapes successives. Les travaux sur les B-Rep sont nombreux, mais nous ne les détaillerons pas davantage car ils dépassent le sujet de ce manuscrit. Grâce au B-Rep sauvegardé, à partir de l'étape précédant l'instanciation de la première caractéristique à modifier, il suffit de réévaluer tout ce qui suit dans l'historique, dans le même ordre que lors de la création. Cette solution est paramétrique et systématique, même inutilement systématique puisque parmi les caractéristiques reconstruites, certaines ne sont nullement affectées par la modification en question. Cette solution présente toutefois l'avantage de ne pas avoir de cas particuliers, donc une simplicité de mise en œuvre. La difficulté vient de la maintenance d'un B-Rep incrémental.

La seconde approche, pour obtenir le B-Rep d'avant l'instanciation de la première caractéristique à modifier, est d'annuler toutes les caractéristiques qui doivent être réévaluées, c'est-à-dire qui sont modifiées ou qui interfèrent avec une caractéristique modifiée. Nous évoquerons désormais cette annulation de caractéristiques instanciées en vue d'une réévaluation, par le terme de *désinstanciation*. La désinstanciation de toutes les caractéristiques à réévaluer est alors suivie de leur *réinstanciation*, c'est-à-dire d'une nouvelle instanciation. Cette solution semble plus intéressante, car la quantité de caractéristiques à réévaluer est optimisée par rapport aux autres solutions ; c'est donc celle que nous retenons et développons dans la suite. La figure 3.1 présente un récapitulatif de la procédure de modification de la couche B-Rep.

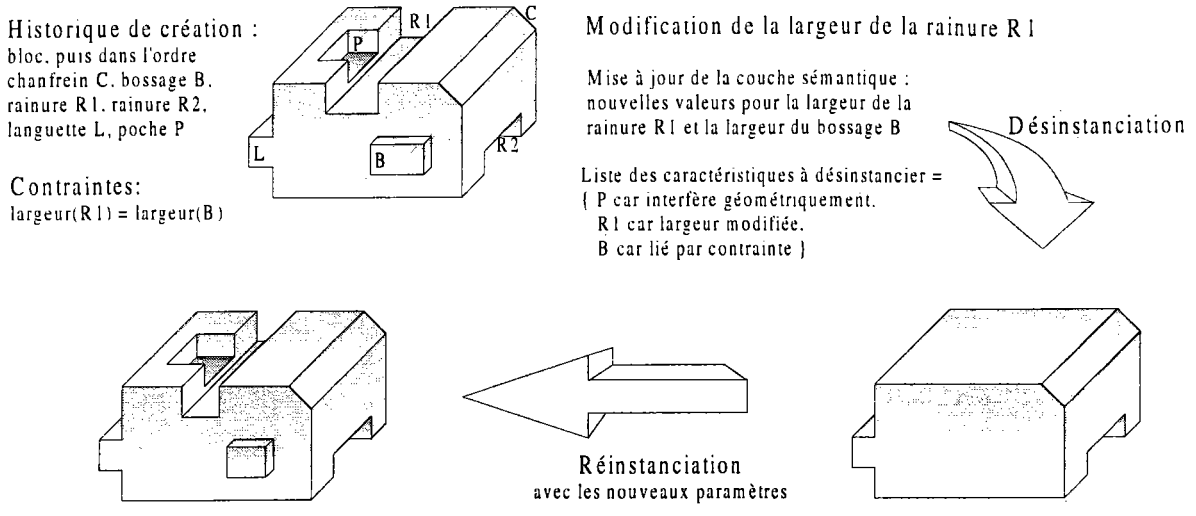


Figure 3.1 : Procédure de modification du B-Rep.

Notons que l'esprit des méthodes proposées ci-dessus est toujours de mettre à jour le B-Rep en déroulant l'historique de création, même le plus partiellement possible. Une partie du B-Rep est donc recréée. Une approche radicalement différente serait d'adapter le B-Rep aux nouveaux paramètres, en déplaçant des faces, des arêtes, des sommets, en corrigeant localement des géométries ou des topologies. Cette approche nous apparaît d'une grande complexité et impliquerait vraisemblablement de disposer d'une bibliothèque de procédures de modification de la géométrie (B-Rep), couvrant toutes les situations qui pourraient survenir. Une première difficulté est de trouver, dans le B-Rep, à quoi correspond un paramètre donné, par exemple une distance entre deux faces, et de déterminer quelles géométries modifier. S'il faut, par exemple, modifier la géométrie d'un certain nombre de sommets en calculant leurs nouvelles coordonnées, la mise à jour ne se limite nullement à leur appliquer une même translation à tous, même si la face est plane. En outre, il faut prévoir d'actualiser la topologie, car il est probable que certaines entités (faces, arêtes...) disparaissent ou apparaissent. Il s'agit donc sans doute d'une solution digne d'intérêt, si elle peut être mise en œuvre, mais elle n'entre pas dans le cadre de cette thèse.

3. DÉTAIL DE LA SOLUTION RETENUE

Après avoir donné un aperçu de toutes les solutions envisagées, nous détaillons à présent la solution retenue, qui consiste dans un premier temps à propager la modification dans la couche sémantique. Pour cela, il faut mettre à jour les paramètres modifiés et prévoir les répercussions de la modification en établissant la liste de tout ce qui doit être réévalué. Dans un second temps, il faut mettre à jour le B-Rep en désinstanciant puis en réinstanciant tout ce qui doit l'être.

3.1. Modification de la couche sémantique

Pour plus de clarté, nous appellerons dorénavant la caractéristique modifiée M. Pour désinstancier tout ce qui interfère avec M, il faut dans un premier temps détecter toutes les caractéristiques à désinstancier. La modification de la couche sémantique établit la liste de toutes ces caractéristiques et met à jour les paramètres qui doivent être changés.

Un parcours de toutes les contraintes de M est nécessaire pour retrouver les autres caractéristiques concernées par ces contraintes. Ceci est immédiat lorsqu'il s'agit de paramètres internes à des caractéristiques. Mais si une contrainte relie un paramètre de M à d'autres paramètres de plusieurs caractéristiques, toutes ces caractéristiques seront liées à M. Par exemple, une distance entre deux faces de deux caractéristiques distinctes indique une dépendance entre ces deux caractéristiques. De plus, les contraintes peuvent entraîner des modifications d'autres paramètres. En effet, si un paramètre est calculé par rapport au paramètre modifié, sa valeur changera elle aussi et entraînera donc une modification de la (des) caractéristique(s) concernée(s). Ainsi, on répercute la propagation jusqu'à ce que toutes les contraintes soient vérifiées. Cette propagation des contraintes est réalisée par une collaboration étroite entre le modèle et le gestionnaire de contraintes. En effet, c'est le modèle qui stocke les contraintes, il a donc connaissance des objets qui y sont impliqués. Il peut donc aisément fournir ces informations au gestionnaire de contraintes, qui vérifie si les contraintes sont encore satisfaites, après la modification. Si tel n'est pas le cas, le gestionnaire de contraintes doit résoudre un système de contraintes et s'il trouve une solution, la transmettre au modèle. Une interface constante entre modèle et gestionnaire de contraintes est donc nécessaire pour réaliser la propagation. Lors de la mise à jour de la couche sémantique, le modèle se charge de conserver la liste de toutes les caractéristiques ayant subi une modification, en vue de mettre à jour la couche B-Rep.

Il faut ensuite ajouter à cette liste de caractéristiques à mettre à jour dans le B-Rep, celles dont l'englobant interfère avec l'un des englobants des caractéristiques modifiées par la propagation de contraintes. En effet, les caractéristiques qui interfèrent avec celles qui sont modifiées peuvent être altérées par la modification, voire disparaître. C'est le cas, par exemple, lorsqu'une rainure est placée sur un bossage, comme le montre la figure 3.2 et que la largeur de celui-ci est diminuée. Sa réévaluation seule entraînerait la disparition de la rainure. C'est pourquoi elle doit aussi être réévaluée.

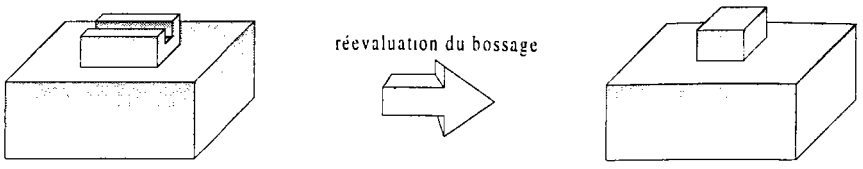


Figure 3.2 : Réévaluation du bossage uniquement.

Pour compléter la liste des caractéristiques à mettre à jour, il suffit de parcourir l'historique des objets où interviennent ces caractéristiques et de tester les intersections entre les englobants. Si celles-ci ne sont pas vides, on ajoute les caractéristiques testées à l'ensemble des caractéristiques à désinstancier (voir l'algorithme ci-dessous). Notons toutefois que plusieurs cas peuvent se produire, selon la nature des caractéristiques. En effet, si elles sont de natures différentes (l'une positive, l'autre négative), leurs englobants peuvent se recouvrir partiellement et leur intersection ne sera donc pas vide. Par contre, si elles sont de même nature, il est possible que leurs volumes utiles soient adjacents, auquel cas leurs englobants pourraient l'être également. Cette situation doit tout de même être prise en compte pour la détection des caractéristiques à mettre à jour, c'est-à-dire que deux caractéristiques adjacentes sont considérées comme interférantes.

```
Algo  Recherche_Caractéristiques_Interférant_Par_Englobant ( LC )
Début
   EnsCaractÀDésinstancier ← ∅
   Pour  toutes les caractéristiques modifiées M de la liste LC  faire
      Pour  toutes les caractéristiques Cara de l'objet contenant M  faire
         Si  Englobant (M) ∩ Englobant (Cara) ≠ ∅  alors
            EnsCaractÀDésinstancier ← EnsCaractÀDésinstancier ∪ Cara
         Fsi
      Fpour
   Fpour
Fin
```

Remarquons toutefois que cette gestion des modifications s'applique plus précisément aux caractéristiques secondaires. En effet, si une caractéristique primaire est modifiée, le volume qu'elle représente est réévalué en totalité. Dans la suite, nous nous intéresserons donc aux modifications des caractéristiques secondaires.

3.2. Modification du B-Rep

Lorsque la liste de toutes les caractéristiques à réévaluer est établie, le B-Rep peut être mis à jour. Nous rappelons que l'instanciation d'une caractéristique secondaire est réalisée par deux opérations booléennes :

- une première opération booléenne entre la génératrice de la caractéristique, obtenue par exécution de sa procédure de création, et son porteur permet de calculer le volume utile, qui est conservé dans la caractéristique instanciée. Il s'agit d'une intersection dans le cas d'une caractéristique négative (qui retire de la matière) et d'une différence dans le cas d'une caractéristique positive (qui ajoute de la matière) ;
- une seconde opération booléenne entre le porteur et le volume utile réalise l'instanciation proprement dite ; c'est une différence pour une caractéristique négative et une union pour une caractéristique positive.

Rappelons également que pour gérer les modifications, nous souhaitons obtenir un B-Rep où toutes les caractéristiques concernées par la modification ont disparu. Pour cela, nous avons établi la liste de toutes ces caractéristiques, qu'il suffit d'annuler dans le B-Rep.

La désinstanciation consiste alors simplement à effectuer l'opération booléenne inverse de l'instanciation entre le porteur et le volume utile, sachant que le porteur n'est plus le même que lors de l'instanciation. En effet, d'une part l'instanciation elle-même est intervenue sur ce porteur et d'autre part, des instanciations d'autres caractéristiques ont pu avoir lieu entre-temps. Il s'agit donc d'une union pour une caractéristique négative et d'une différence pour une caractéristique positive. La preuve de cette technique de désinstanciation par les opérations booléennes est réalisée au paragraphe 4.2.

La solution qui consiste à effectuer les désinstanciations dans l'ordre inverse de l'historique assure que le résultat obtenu après les désinstanciations est conforme aux souhaits, puisqu'il est équivalent à ce qui existait avant l'instanciation de la plus ancienne caractéristique à réévaluer, à quelques caractéristiques non interférentes près. L'exemple de la figure 3.3 montre qu'en désinstanciant des caractéristiques dans un ordre différent, on n'obtient pas un volume correct. Le problème vient du volume utile, car le porteur à partir duquel il est calculé comporte d'autres caractéristiques. Les parties communes aux génératrices appartiennent donc au volume utile de la dernière caractéristique instanciée, ce qui crée une incohérence lors des désinstanciations si elles ne sont pas réalisées dans le bon ordre.

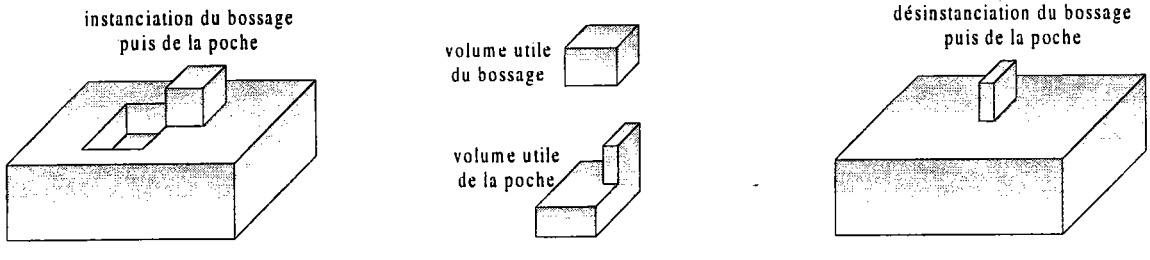


Figure 3.3 : Désinstanciation de caractéristiques dans un ordre différent de l'inverse de la chronologie.

Après avoir désinstancié toutes les caractéristiques concernées par la modification, il faut les réinstancier avec leurs nouveaux paramètres. La nouvelle génératrice de la caractéristique est obtenue en exécutant la procédure de création de la forme. Les volumes utiles sont alors réévalués, en effectuant la première opération booléenne de l'instanciation. La dernière étape de la réinstanciation est le calcul de la seconde opération booléenne, entre le volume utile et le porteur.

Nous rappelons que la procédure de création de la forme permet de générer à la fois le B-Rep de la génératrice de la caractéristique et son graphe de contraintes. De plus, les nœuds du graphe conceptuel sont des géométries correspondant aux faces de la caractéristique. Lorsque les faces sont modifiées, il faut donc mettre ces géométries à jour. La figure 3.4 schématise les liens existant entre le B-Rep et le graphe.

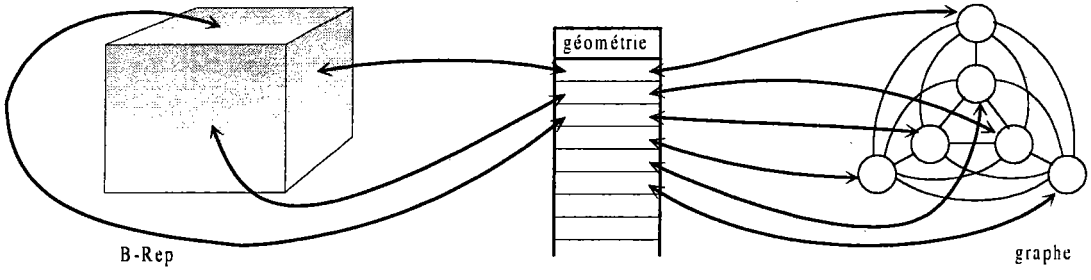
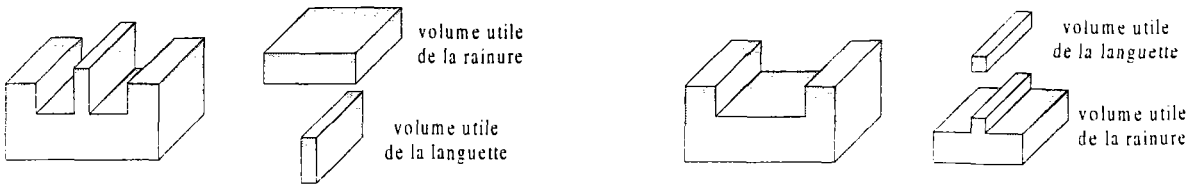


Figure 3.4 : Liens entre le B-Rep et le graphe conceptuel d'une caractéristique.

Lors d'une modification, les contraintes ne sont pas remises en cause, seul le B-Rep est changé. C'est pourquoi il est inutile de recréer le graphe au cours d'une réinstanciation. Pour éviter cela, une option est associée à la procédure de création pour distinguer la première instanciation d'une réinstanciation. Dans ce second cas, la procédure de création génère uniquement un nouveau B-Rep et met à jour les géométries correspondant aux anciennes faces, avec les équations des nouvelles.

Une remarque concerne la justification de l'ordre dans lequel les réinstanciations de toutes les caractéristiques désinstanciées sont réalisées. Elles sont faites dans l'ordre de la chronologie, ce qui garantit la conformité du résultat aux souhaits initiaux de l'opérateur. Il est très facile de s'apercevoir que le non-respect de cet ordre peut créer des erreurs. Par exemple, si deux caractéristiques interférantes ne sont plus recréées dans le même ordre, et que la première est négative et la seconde positive, on n'obtiendra pas le résultat souhaité. Si elles sont créées dans l'ordre inverse la caractéristique négative altérera forcément la caractéristique positive déjà instanciée, ce qui n'est pas souhaitable, comme le montre l'exemple de la figure 3.5.



(a) réinstanciation de la rainure puis de la languette

(b) réinstanciation de la languette puis de la rainure

Figure 3.5 : Importance de l'ordre des réinstanciations, l'ordre initial étant : bloc, rainure, languette.

4. JUSTIFICATION DE LA DÉMARCHE

Après avoir présenté en détail toutes les étapes de la démarche, nous justifions à présent qu'elles sont toutes nécessaires. Nous faisons ensuite la preuve de la désinstanciation, à l'aide d'une formalisation des opérations booléennes. Enfin, nous évoquerons quelques problèmes qui peuvent survenir lors de la réinstanciation.

4.1. Nécessité de toutes les étapes

Le fait de désinstancier une caractéristique, de la modifier puis de la réinstancier, peut avoir des conséquences sur d'autres caractéristiques à savoir celles qui y sont liées par des contraintes et celles qui interfèrent géométriquement.

Les caractéristiques qui interfèrent avec celle qui est modifiée conservent les mêmes dimensions et positionnement. Ceci suggère a priori de ne pas réévaluer leurs volumes utiles pour mettre à jour la partie du B-Rep qui les concerne. On se contenterait alors d'ajouter ou de retirer (selon les cas) une nouvelle fois leurs volumes utiles (voir la figure 3.6). Ceci permet d'éviter ainsi deux opérations booléennes, celle de la désinstanciation et celle du calcul du volume utile.

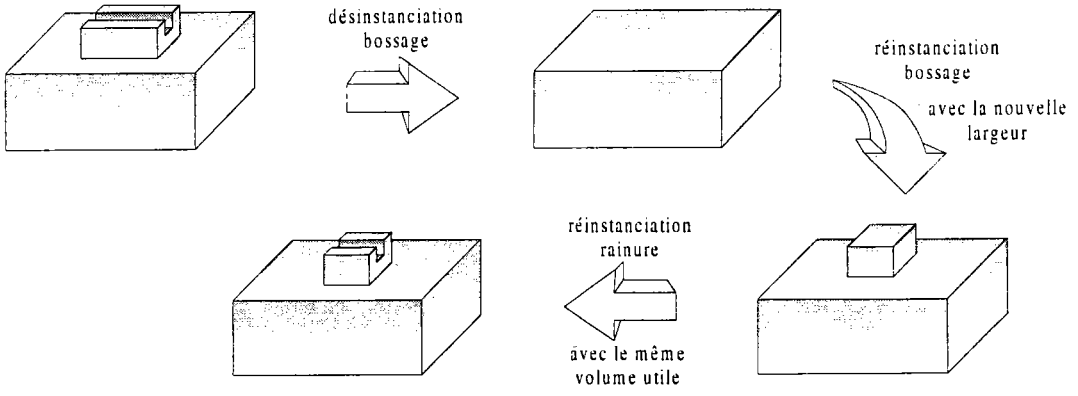


Figure 3.6 : Modification de la largeur d'un bossage.

Cependant, la réévaluation est moins simple qu'elle ne le paraît à première vue : en effet, le volume utile initial peut ne plus être correct, du côté où la génératrice de la caractéristique se prolonge automatiquement, comme le montre l'exemple ci-dessous. Nous ne pouvons donc pas faire l'économie de la première opération booléenne, qui évalue le volume utile.

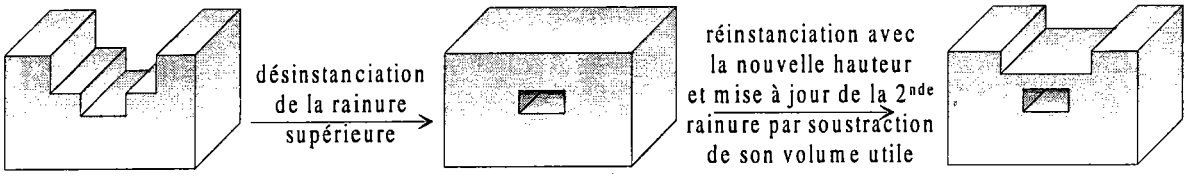


Figure 3.7 : Modification de la position d'une rainure.

Une seconde solution serait de ne toujours pas désinstancier les caractéristiques interférantes mais de recalculer leur volume utile tout de même et de les réinstancier avec le volume utile modifié. Cette solution permettrait encore d'éviter une opération booléenne, celle de la désinstanciation. Cependant, le calcul du volume utile est réalisé à partir d'un porteur où la caractéristique est déjà instanciée. Il n'est donc plus correct (et ne pourra plus être utilisé pour une désinstanciation) car il représente uniquement les interférences entre les caractéristiques. L'exemple de la figure 3.8 montre que la solution semble correcte a priori, mais le volume utile de la rainure n'est plus cohérent, car il est maintenant cylindrique.

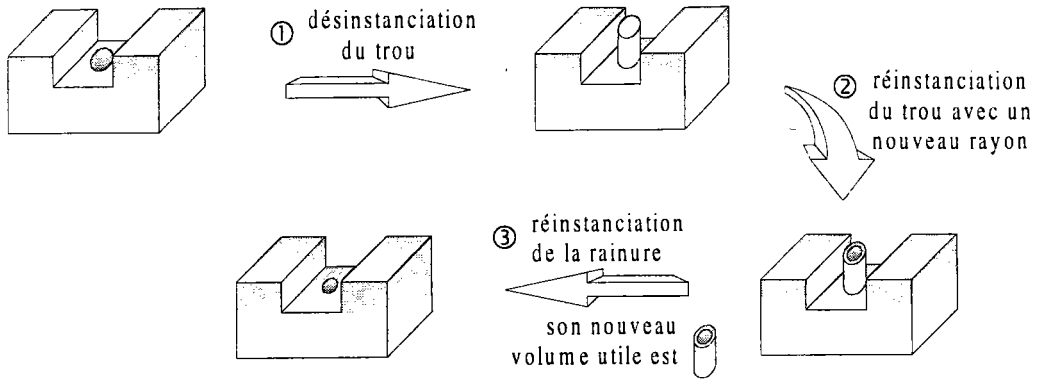


Figure 3.8 : Calcul d'un volume utile sans désinstanciation préalable : on crée le trou, la rainure puis on diminue le rayon du trou.

Toutes les étapes de la modification sont donc nécessaires : la désinstanciation, la réévaluation du volume utile et la réinstanciation de la caractéristique modifiée et de toutes celles qui interfèrent.

4.2. Preuve de la désinstanciation

Lors de l'instanciation des caractéristiques, le volume de matière exactement ajouté ou retiré (volume utile) est calculé et conservé. La désinstanciation d'une caractéristique consiste à retirer ou à ajouter ce même volume utile. Il paraît intuitif qu'en désinstanciant les caractéristiques à modifier dans l'ordre inverse de la chronologie, on obtient bien un objet dans lequel toutes les caractéristiques à modifier ont disparu. Cependant, nous proposons de transcrire cette intuition de façon plus formelle, en la démontrant. La preuve est réalisée cas par cas. Nous différencions les caractéristiques positives et négatives, car les opérations booléennes sont différentes. Nous supposerons qu'il n'y a qu'une seule caractéristique à désinstancier, car dans le cas contraire, il suffit d'appliquer la désinstanciation plusieurs fois, de façon successive.

Nous rappelons tout d'abord quelques propriétés des opérations booléennes, qui serviront dans la preuve et y seront référencées par leur numéro. Les propriétés (0) à (6) sont des propriétés établies dans la théorie des ensembles [Hog 92].

Propriétés évidentes :	$A \cup \varnothing = A$	(0.1)	$A \cap \varnothing = \varnothing$	(0.2)
	$A \cup A = A$	(0.3)	$A \cap A = A$	(0.4)
Commutativité de l'union :	$A \cup B = B \cup A$			(1)
Commutativité de l'intersection :	$A \cap B = B \cap A$			(2)
Associativité de l'union :	$A \cup (B \cup C) = (A \cup B) \cup C$			(3)

Associativité de l'intersection : $A \cap (B \cap C) = (A \cap B) \cap C$ (4)

Distributivité de l'union : $A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$ (5)

Distributivité de l'intersection : $A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$ (6)

Propriétés supplémentaires illustrées par la figure 3.9 et montrées en annexe C :

si $A \cap B = \varnothing$ alors $A - B = A$ (7)

si $A \subset B$ alors $A \cup B = B$ (8)

$A - B - C = A - C - B$ (9)

$A - (A \cap B) = A - B$ (10)

$(A \cup B) - B = A - B$ (11)

$A \cup (B - A) = A \cup B$ (12)

$(A - B) \cup (A \cap B) = A$ (13)

$(A \cup B) - (B - A) = A$ (14)

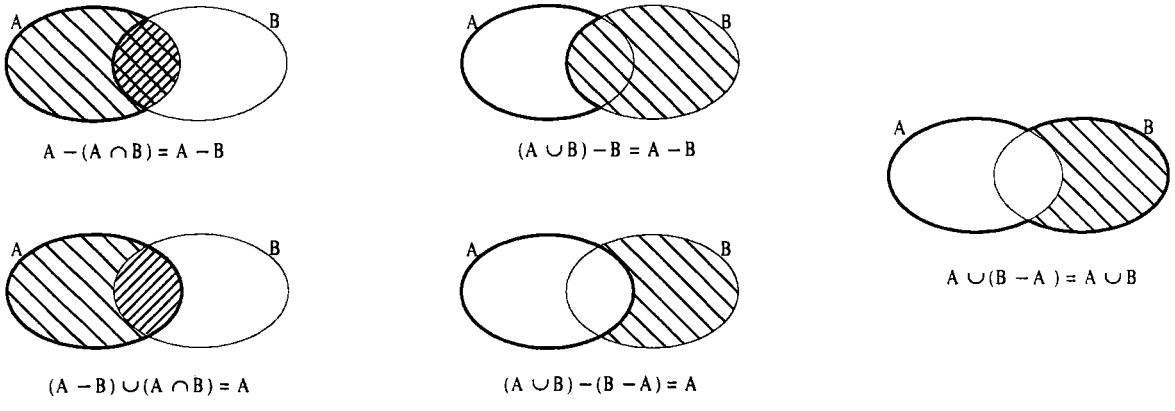


Figure 3.9 : Illustration de quelques propriétés des opérations booléennes.

Nous rappelons également qu'en l'absence de parenthèses, les priorités sont à gauche.

Preuve de la désinstanciation :

La situation la plus simple est celle où la désinstanciation intervient juste après l'instanciation, sans qu'il y ait eu d'autre instanciation entre temps.

Appelons O le volume porteur, G la génératrice de la caractéristique et VU son volume utile.

1er cas : la caractéristique est négative ; son volume utile est donc obtenu par intersection entre le porteur et sa génératrice ($VU = O \cap G$). L'instanciation est la différence entre le porteur et le volume utile ($O - VU$) et la désinstanciation, l'opération booléenne inverse, c'est-à-dire l'union avec le volume utile. L'historique de l'objet après instanciation de la caractéristique devient donc $O - VU$, puis avec la désinstanciation on obtient $(O - VU) \cup VU$. Il faudrait donc que le résultat de la désinstanciation soit égal à O.

Hypothèse : $VU = O \cap G$ (hyp 1)

Montrons que : $(O - VU) \cup VU = O$

$(O - VU) \cup VU = [O - (O \cap G)] \cup (O \cap G)$ d'après hypothèse (hyp 1)

$= (O - G) \cup (O \cap G)$ d'après propriété (10)

$= O$ d'après propriété (13)

2nd cas : la caractéristique est positive ; son volume utile est donc obtenu par différence entre son volume infini et le porteur ($VU = G - O$). L'instanciation représente l'union entre le porteur et le volume utile ($O \cup VU$) et la désinstanciation, l'opération booléenne inverse, c'est-à-dire le retrait du volume utile.

Hypothèse : $VU = G - O$ (hyp 2)

Montrons que : $(O \cup VU) - VU = O$

$(O \cup VU) - VU = [O \cup (G - O)] - (G - O)$ d'après hypothèse (hyp 2)

$= (O \cup G) - (G - O)$ d'après propriété (12)

$= O$ d'après propriété (14)

Une situation un peu moins simple est celle où il y a eu une opération de modélisation (instanciation d'une autre caractéristique AC) entre l'instanciation et la désinstanciation de la caractéristique à modifier CM. Nous distinguons quatre cas, selon le type des caractéristiques CM et AC (qui peuvent être positives ou négatives). Montrons qu'en réalisant la désinstanciation, on obtient un porteur à partir duquel on peut entamer la réinstanciation rendue nécessaire par la modification.

Appelons VU le volume utile de la caractéristique à modifier CM, C le volume utile de la caractéristique AC (dont l'instanciation suit l'instanciation de CM et précède sa désinstanciation) et O l'objet porteur.

Nous rappelons que le volume utile VU de la caractéristique modifiée CM n'interfère pas avec le volume utile de la caractéristique AC instanciée après. L'intersection entre ces deux volumes utiles est donc vide. En effet, si C et VU interféraient, la caractéristique AC aurait été désinstanciée avant CM (dans l'ordre inverse de l'historique). Nous avons donc l'hypothèse :

$C \cap VU = \varnothing$ (H)

1. La caractéristique modifiée CM est négative

Nous rappelons que le volume utile VU de la caractéristique modifiée est calculé par intersection entre la génératrice et le porteur ($VU = O \cap G$). Le volume utile est donc inclus dans le volume porteur, d'où l'hypothèse :

Hypothèse : $VU \subset O$ (h1)

L'historique de l'objet après instanciation de la caractéristique CM est donc $O - VU$.

1.1. La caractéristique suivante AC est négative

(h1) et (H) $\Rightarrow$ $VU \subset (O - C)$ (h1.1)

L'historique de l'objet après instanciation de la caractéristique AC devient donc $(O - VU) - C$. Le résultat de la désinstanciation est $(O - VU - C) \cup VU$.

$(O - VU - C) \cup VU = (O - C - VU) \cup VU$ d'après (9)

$= (O - C) \cup VU$ d'après (12)

$= O - C$ d'après (h1.1) et (8)

1.2. La caractéristique suivante AC est positive

L'historique de l'objet après instanciation de AC devient donc $(O - VU) \cup C$.

$[(O - VU) \cup C] \cup VU = (O - VU) \cup VU \cup C$ d'après (3) et (1)

$= (O \cup VU) \cup C$ d'après (12)

$= O \cup C$ d'après (h1) et (8)

2. La caractéristique modifiée est positive

Hypothèse : $VU \cap O = \varnothing$ (h2)

2.1. Les instanciations suivantes sont négatives

$[(O \cup VU) - C] - VU = [(O \cup VU) - VU] - C$ d'après (9)

$= (O - VU) - C$ d'après (11)

$= O - C$ d'après (h2) et (7)

2.2. Les instanciations suivantes sont positives

(h2) et (H) $\Rightarrow$ $VU \cap (O \cup C) = \varnothing$ (h2.2)

$(O \cup VU \cup C) - VU = (O \cup C \cup VU) - VU$ d'après (1)

$= (O \cup C) - VU$ d'après (11)

$= O \cup C$ d'après (h2.2) et (7)

Lorsqu'il y a eu plusieurs créations de caractéristiques entre l'instanciation et la désinstanciation de CM, il suffit de généraliser les règles démontrées ci-dessus, en les appliquant plusieurs fois successivement, autant de fois qu'il y a de caractéristiques intercalées.

4.3. Problèmes liés à la réinstanciation

Certaines géométries (surfaces, lignes...) d'une caractéristique peuvent être contraintes. Si, lors d'une réévaluation de la caractéristique, ces géométries viennent à disparaître, il n'est plus possible de gérer correctement ces contraintes. Dans l'état actuel de nos réflexions, nous estimons que cette situation ne devrait pas survenir car elle traduirait une incohérence : une réinstanciation doit produire autant de géométries que l'instanciation initiale.

Ceci n'implique pas que le nombre de faces générées, par exemple, soit constant. Par contre, chaque géométrie de la génératrice doit produire une géométrie dans l'objet lors de la réinstanciation. Ceci assure la validité de la caractéristique réinstanciée et permet de faire une correspondance bijective entre les anciennes géométries et les nouvelles, ce qui garantit en même temps le maintien des contraintes géométriques.

La réinstanciation soulève une autre difficulté, sans rapport avec la précédente. La suite de cette partie y est consacrée.

Le mécanisme qui gère une modification a été décrit dans le paragraphe 3 comme un processus très systématique en trois étapes : propagation et satisfaction des contraintes, désinstanciation des caractéristiques à corriger, réinstanciation de ces mêmes caractéristiques.

Pourtant, une situation particulière peut remettre en cause cet aspect très systématique. En effet, c'est pendant la première phase que la liste des caractéristiques à désinstancier puis à réinstancier est établie. Ces caractéristiques sont celles dont un paramètre a été modifié suite à la propagation de contraintes ou celles qui interfèrent avec une caractéristique de la première catégorie. Or, la détection des interférences s'appuie sur les volumes utiles des caractéristiques, qui ne sont réévalués que lors de la troisième phase de la gestion des modifications. Pendant la réinstanciation, il peut donc arriver qu'une caractéristique, parce qu'elle a été agrandie ou déplacée, interfère avec une autre caractéristique à laquelle elle ne se superposait pas et qu'il n'était donc pas prévu de réévaluer. En conséquence, le volume utile de cette autre caractéristique n'est plus à jour, ce qui fausserait les désinstanciations à venir ou la génération d'une gamme d'usinage.

Une première solution pour pallier ce problème est de recommencer toute l'étape de mise à jour du B-Rep. Lorsqu'une interférence est détectée au cours d'une réinstanciation, la nouvelle caractéristique qui interfère est ajoutée à la liste des caractéristiques à désinstancier. À partir de l'objet tel qu'il était avant la modification, qui aura donc été sauvegardé, on recommence les désinstanciations et les réinstanciations, avec la liste de caractéristiques complétée. Le risque de cette solution est de multiplier les tentatives de mise à jour du B-Rep, puisque tout l'algorithme est déroulé entièrement à chaque détection d'une nouvelle interférence. Le seul avantage de cette solution est son caractère systématique.

La deuxième solution consiste à prévoir les futures interférences, avant d'entamer les désinstanciations. Comme la détection des interférences s'appuie sur les volumes utiles, il serait souhaitable de pouvoir calculer à l'avance les nouveaux volumes utiles des caractéristiques modifiées. Or, ceci n'est pas possible, sans les désinstancier au préalable. Par contre, les génératrices peuvent être obtenues par exécution de la procédure de création de la forme dès que la première étape est achevée, c'est-à-dire quand les paramètres sont mis à jour. L'idée est donc de prévoir les interférences en exploitant les nouvelles génératrices, plutôt que les volumes utiles. Nous proposons donc de réaliser la première opération booléenne de l'instanciation entre ces génératrices et le volume porteur, c'est-à-dire celle qui permet a priori de calculer le volume utile. Nous rappelons que cette opération est une intersection dans le cas d'une caractéristique négative et une différence dans le cas d'une caractéristique positive. À la différence de l'instanciation, le résultat de l'opération booléenne ne sera pas égal au nouveau volume utile, puisque la caractéristique n'aura pas été supprimée du volume porteur, mais simplement à la différence entre le nouveau volume utile et l'ancien. Ce résultat peut être nul, si le nouveau volume utile est plus petit que l'ancien par exemple, auquel cas il n'y a pas d'interférence supplémentaire. Par contre, si le volume obtenu est non nul, il faut rechercher les interférences entre ce volume et son environnement pour prévoir les interférences qui apparaîtront lors de la réinstanciation (voir figure 3.10). Si des interférences sont détectées, les caractéristiques concernées sont simplement ajoutées à la liste des caractéristiques à désinstancier. La mise à jour du B-Rep peut alors être réalisée de la même manière que celle présentée dans le paragraphe 3.2.

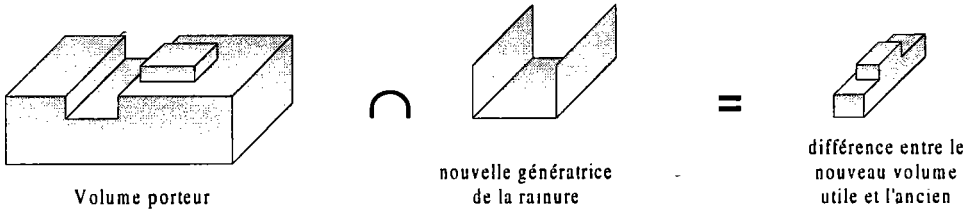


Figure 3.10 : La différence, entre le nouveau volume utile de la rainure et l'ancien, interfère avec le bossage.

Le coût de cette solution peut être exprimé en nombre d'opérations booléennes à réaliser. La première, entre la nouvelle génératrice et le porteur, permet de calculer la différence entre le nouveau volume utile et l'ancien. La désinstanciation et la réinstanciation exigent respectivement une et deux opérations booléennes, soit un total de quatre pour la caractéristique modifiée. Supposons alors que n interférences soient détectées entre le résultat de la première opération et son environnement, les caractéristiques qui interfèrent à présent (avec la différence entre le nouveau volume utile et l'ancien) doivent être désinstanciées et réinstanciées, ce qui implique trois opérations supplémentaires par caractéristique, soit au total trois fois n. Le coût total de cette solution en nombre d'opérations booléennes, en cas de détection de n nouvelles interférences, s'élève donc à $4 + 3n$.

Nous proposons une dernière solution, qui consiste à agir "intelligemment" en cours de réinstanciation. Comme nous l'avons déjà souligné, l'ordre des réinstanciations doit être inverse à l'historique de création. Nous ne pouvons donc pas désinstancier la caractéristique avec laquelle on détecte une interférence au moment où cette interférence est détectée. Il faut donc trouver une solution pour mettre à jour les deux caractéristiques qui interfèrent, sans les désinstancier ni les réinstancier. Nous constatons que la difficulté est de gérer la partie commune aux deux caractéristiques, en particulier de décider auquel des volumes utiles il convient de l'affecter.

Or, si ces caractéristiques sont de même nature (toutes les deux positives ou toutes les deux négatives), peu importe le volume utile auquel appartient la partie commune. En effet, puisqu'elles interfèrent désormais, si l'une doit être désinstanciée, l'autre le sera forcément aussi. Leur partie commune sera donc bien ajoutée ou retirée, selon le cas, car elle fait partie de l'un ou l'autre des volumes utiles. En effet , elle est contenue dans le volume utile de la caractéristique avec laquelle on détecte une interférence. Ceci simplifie les calculs puisqu'il suffit de réinstancier la caractéristique modifiée de manière classique, sans se préoccuper de l'autre (voir figure 3.11).

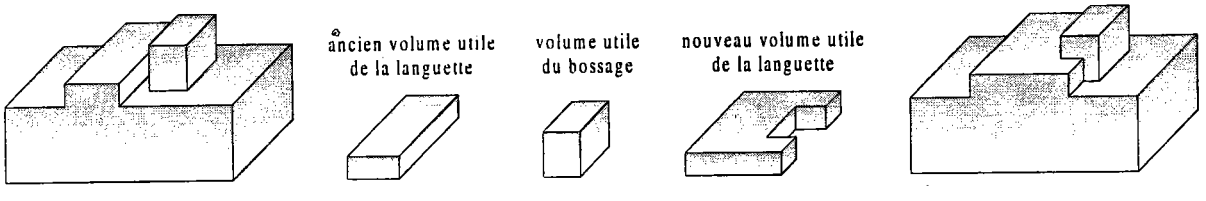


Figure 3.11 : Caractéristiques de même nature (languette et bossage) qui interfèrent suite à un élargissement de la languette.

Par contre, si les deux caractéristiques qui interfèrent sont de natures opposées, leur partie commune peut appartenir au mauvais volume utile et donc être ajoutée alors qu'elle devrait être

retirée ou inversement. Nous proposons donc d'établir une priorité entre les caractéristiques positives et les négatives. C'est à l'opérateur de déterminer la priorité qu'il désire. Lorsque le cas se présente, il sera donc consulté, à moins qu'il n'ait configuré son système une fois pour toutes, en attribuant la plus forte priorité aux caractéristiques positives, aux négatives ou aux plus anciennes (prédominance de l'historique).

Appelons CM la caractéristique modifiée qui vient d'être réinstanciée, VU_{CM} son nouveau volume utile, CI la caractéristique interférant et VU_{CI} son volume utile.

Si CM est plus prioritaire que CI, la pièce obtenue après la réinstanciation de CM est correcte.

Par contre, si CI est plus prioritaire que CM, la pièce finale doit être mise à jour. Pour évaluer le volume de matière correspondant à l'interférence entre les deux caractéristiques, nous calculons l'intersection entre le volume utile de CM et la génératrice de CI (une intersection entre les volumes utiles est insuffisante). Appelons VI le volume résultant de cette intersection. Si CM est positive et CI est négative, VI est retiré à la pièce ; si CM est négative et CI positive, VI est ajouté à la pièce.

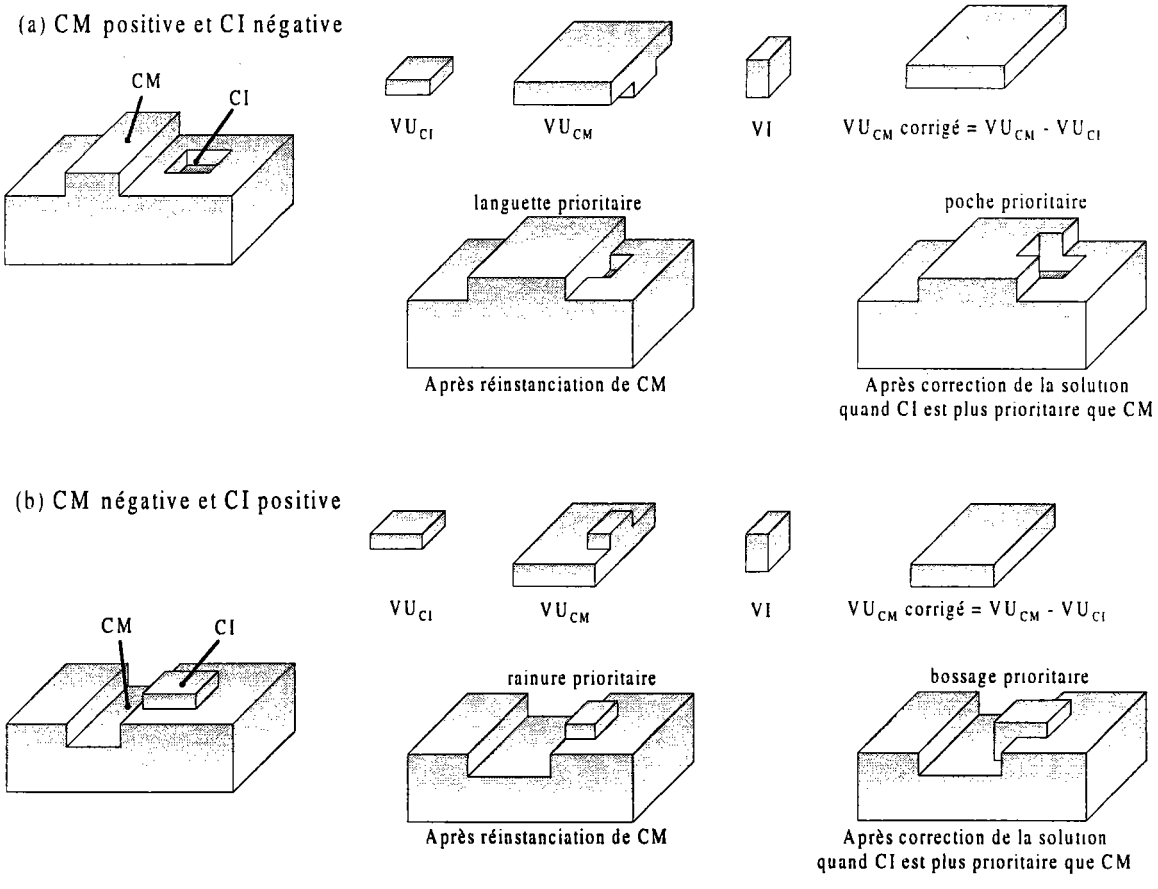


Figure 3.12 : Détection, en cours de réinstanciation, d'une interférence entre une caractéristique réinstanciée, CM, et une caractéristique non réinstanciée, CI. Dans le premier cas (a), CM ajoute de la matière, CI en retire. Dans le second cas (b), c'est l'inverse.

Dans les deux cas, nous constatons, de plus, que pour éviter des problèmes lors de futures désinstanciations, les volumes utiles devraient être disjoints. Nous avons donc vérifié, pour tous les cas de figure, qu'il suffit pour corriger VU_{CM} de lui retirer VU_{CI}, quelles que soient les priorités. La figure 3.12 illustre ces résultats.

Nous pouvons également évaluer le coût de cette solution, comme nous l'avons fait pour la précédente. Il faut, en effet, trois opérations booléennes pour la désinstanciation et la réinstanciation de la caractéristique modifiée, jusqu'à n opérations supplémentaires pour la mise à jour du volume utile de CM par soustraction de chacun des volumes utiles des n caractéristiques CI (lorsque les CI sont de nature opposée à CM), et si les CI sont plus prioritaires que CM, au plus $2n$ opérations pour le calcul des VI (intersection entre le volume utile de CM et la génératrice de CI) et la correction du résultat de la réinstanciation. Le coût de cette solution est donc au plus égal à $3 + 3n$ opérations booléennes. Nous avons cependant remarqué qu'il est inutile d'effectuer un traitement au niveau de la caractéristique avec laquelle on détecte une interférence lorsque cette dernière a même nature que la caractéristique qui empiète sur elle. Dans le meilleur des cas, quand CM a même nature que toutes les caractéristiques qu'elle recouvre, le nombre d'opérations booléennes et donc réduit à $3 + n$. Nous jugeons donc cette solution préférable à la précédente.

Remarquons qu'il n'est utile de chercher de nouvelles interférences qu'avec les caractéristiques qui sont redimensionnées ou déplacées. En effet, si elles ne sont réévalués que parce qu'elles interfèrent, mais qu'aucun de leurs paramètres (ni de dimensions, ni de position) n'est modifié, leur volume utile ne peut pas interférer avec une caractéristique qu'il ne recouvrait pas avant sa réévaluation.

5. CONCLUSION

Ce chapitre a montré que la représentation des caractéristiques de forme proposée au chapitre 2 est adaptée à la gestion des modifications, qui sont réalisées en deux étapes :

- mise à jour de la couche sémantique c'est-à-dire de toutes les valeurs des paramètres qui sont modifiés (après propagation des contraintes) et constitution de la liste de toutes les caractéristiques liées aux caractéristiques modifiées par des contraintes ou interférant géométriquement avec elles ;

- mise à jour de la couche B-Rep par désinstanciation des caractéristiques de la liste évoquée ci-dessus, puis réinstanciation de toutes les caractéristiques désinstanciées, dans l'ordre de l'historique.

Cette gestion des modifications implique, pour chaque caractéristique concernée, jusqu'à quatre opérations booléennes : une pour la détection des interférences, une pour la désinstanciation, une pour le calcul du nouveau volume utile et une pour la réinstanciation. Les opérations booléennes étant relativement coûteuses, nous pouvons nous demander si une réexécution de l'historique de tous les volumes concernés par une modification ne serait pas plus rentable, même si elle impliquerait sans doute la réévaluation d'éléments inchangés.

Appelons NCT le nombre de caractéristiques totales dans les volumes ayant subi une modification et NCM le nombre de caractéristiques effectivement modifiées ($NCM \leq NCT$). La réexécution systématique de l'historique implique un nombre d'opérations booléennes égal à deux fois NCT alors que la méthode de modification par désinstanciation décrite ci-dessus en implique quatre fois NCM. Elle n'est donc intéressante, d'un point de vue performance, qui si $4*NCM < 2*NCT$, c'est-à-dire si le nombre de caractéristiques modifiées est inférieur à 50 % du nombre total de caractéristiques, si l'on considère que toutes les opérations booléennes ont une complexité équivalente. D'un strict point de vue performances, la méthode est donc souvent plus rentable. De plus, la réexécution systématique n'a qu'un intérêt réel moindre du point de vue recherche par rapport à la méthode "intelligente".

Chapitre 4 - DÉFINITION INTERACTIVE DE NOUVELLES CARACTÉRISTIQUES GÉNÉRIQUES

1. INTRODUCTION

Nous ne pouvons pas envisager que le système de CFAO, dans sa version de base, fournisse au concepteur toutes les caractéristiques de sa spécialité. Il faut donc lui permettre d'en intégrer de nouvelles dont il pourra se servir de manière répétitive.

Pour que les nouvelles caractéristiques génériques soient exploitables de la même manière que les caractéristiques prédéfinies, elles doivent impérativement avoir la même structure (des données et un comportement). Cette structure a été décrite en détail dans la première partie du chapitre 2. Un opérateur non informaticien doit donc être capable de mener à bien une réalisation informatique sans connaissance informatique. Il s'agit bien d'une réalisation informatique car il doit entre autres décrire le comportement de la caractéristique, autrement dit des algorithmes (obtention de la forme, vérification de la validité...).

Les premières propositions pour enrichir la bibliothèque de caractéristiques ont été faites dans [GMP 96a]. Elles ont été développées dans [GMP 96b].

Nous allons voir dans la première partie de ce chapitre comment définir une nouvelle caractéristique générique et l'intégrer à la bibliothèque des caractéristiques prédéfinies. Une deuxième partie sera dédiée plus particulièrement à la description interactive de la forme de cette nouvelle caractéristique. Enfin, la dernière partie traitera de la définition du reste du

comportement, c'est-à-dire de l'extraction et de la transformation en caractéristiques d'application.

Nous précisons que ce chapitre est une spécification du module de description interactive de caractéristiques de forme tel que nous le prévoyons. Le problème est traité dans son ensemble, mais il reste à approfondir quelques points de détails qui seront mis en évidence dans le texte. Il n'a pas fait l'objet d'une mise en œuvre à la date de la rédaction de ce manuscrit mais figure dans nos priorités pour la période à venir.

2. EXTENSION DE LA BIBLIOTHÈQUE

L'extension de la bibliothèque soulève deux types de problèmes : d'une part la description de la caractéristique (nature des données et algorithmes des méthodes), d'autre part son intégration dans le logiciel. Les deux paragraphes à venir abordent ces deux problèmes.

2.1. Description d'une nouvelle caractéristique générique

Nous avons souligné dans l'introduction que la caractéristique créée devait avoir la même structure que les caractéristiques prédéfinies. Une normalisation du format des caractéristiques de forme apparaît donc indispensable. Nous pouvons ainsi guider précisément l'utilisateur du logiciel dans l'étape de description d'une nouvelle caractéristique, par exemple en créant le squelette de la description et en le faisant compléter par l'opérateur.

Le système devrait même offrir le moyen de réutiliser tout ou partie d'une autre caractéristique et une organisation hiérarchique de la bibliothèque, combinée à une approche objets, prend alors toute sa signification. Lorsque la caractéristique créée est une particularisation d'une autre, comme l'est la rainure à queue d'aronde pour la rainure classique à trois faces, il est clair que les mécanismes d'héritage et de surcharge sont utiles : l'héritage, car la queue d'aronde réutilise une partie du comportement de la rainure ; la surcharge, car une autre partie du comportement, bien qu'ayant le même objectif, doit être adaptée. Ceci confirme l'intérêt d'une approche objets dans la structuration de la bibliothèque. De plus, l'organisation hiérarchique de la bibliothèque permet de fournir à l'utilisateur un squelette qu'il n'aura qu'à compléter. En effet, si l'opérateur précise la position d'une nouvelle caractéristique dans la hiérarchie, les fonctions à

définir seront celles de la caractéristique dont elle est dérivée ; il faudra soit les surcharger si elles sont différentes, soit simplement en hériter.

Nous avons abordé, au chapitre 2, la structuration de la bibliothèque en proposant une hiérarchie dans laquelle nous distinguons les caractéristiques primaires des secondaires, puis dans les secondaires, les positives des négatives. La première distinction se justifie par le fait que seules les caractéristiques secondaires sont des altérations de leur volume porteur ; elles nécessitent donc d'être combinées avec le porteur, contrairement aux primaires. La distinction entre caractéristiques positives et négatives est due au fait que l'opérateur de combinaison avec le porteur est différent. En fait, que ce soit pour le calcul du volume utile, pour l'instanciation proprement dite ou pour la désinstanciation, l'opération booléenne effectuée est commune à toutes les caractéristiques positives ainsi qu'à toutes les négatives, mais différente entre les positives et les négatives. Cette classification permet de spécifier les méthodes de calcul du volume utile, d'évaluation et d'annulation au niveau des caractéristiques secondaires positives et négatives, et d'en faire hériter les classes dérivées.

Nous aurions également pu regrouper les caractéristiques par forme semblable : de type *boîte* comme la marche, la rainure, la poche... ou de type *cylindre* comme le trou... L'intérêt d'un tel regroupement est la similarité du procédé de fabrication, qui est vraisemblablement commun pour une même forme. Cependant, même si les formes de certaines caractéristiques sont semblables, leurs génératrices ne sont pas identiques. En effet, si la forme de base est une boîte, pour les exemples cités, nous constatons que seules quatre faces de la boîte sont conservées pour la génératrice de la poche, trois pour la rainure et deux pour la marche. La procédure de création de la forme n'étant pas la même, il paraît inutile de regrouper ces caractéristiques. De plus, certaines caractéristiques qui appartiennent à une même famille peuvent avoir des formes différentes. Pour la famille des rainures, par exemple, nous pouvons citer des formes en U, en V, en T, en queue d'aronde...

Une autre possibilité de classement est un regroupement par fonction, telle qu'un guidage, un blocage... Un problème se pose alors lorsqu'une même caractéristique a des fonctions différentes. Une rainure, par exemple, peut réaliser un guidage en translation ou un blocage en rotation. Dans ce cas, il est délicat de classer la caractéristique en question. Nous abandonnerons donc également cette possibilité et conservons la hiérarchie de la figure 2.2 pour la structure de la bibliothèque des caractéristiques génériques.

Le fait de préciser la place de la nouvelle caractéristique dans la hiérarchie implique que sa catégorie (primaire ou secondaire) et son type (positif ou négatif) sont déterminés implicitement.

Ce sont les premières données nécessaires à la définition d'une caractéristique. Nous rappelons que les autres données sont une liste de paramètres et un ensemble de contraintes génériques. Nous proposons de préciser ces données au moment de la description de la forme, puisqu'elles en dépendent l'une comme l'autre. Nous y reviendrons aux paragraphes 3.4.1 et 3.4.2.

En complément des données, nous avons vu, au chapitre 2, que les caractéristiques génériques étaient également modélisées par des méthodes, permettant de décrire une partie du comportement. Aussi, le langage utilisé pour décrire les traitements doit être accessible à un non-informaticien. Nous le considérons comme un sur-ensemble du langage de développement du système de CAO car il fournit les mêmes outils mais en les intégrant dans un environnement plus souple, en offrant par exemple des outils algorithmiques pour passer automatiquement en revue toutes les faces d'un objet ou toutes les arêtes d'une face. Ceci étant, il paraît malgré tout très ambitieux d'espérer offrir les moyens, à un non-informaticien, de décrire n'importe quel traitement, aussi compliqué soit-il. Aussi l'intervention d'un informaticien reste-t-elle certainement nécessaire, avec une exception pour la description de la forme d'une caractéristique de forme. Dans ce cas, nous proposons d'exploiter l'interactivité de l'environnement de CAO qui se substitue alors au langage de développement. Nous précisons au paragraphe 3 la manière dont le concepteur peut décrire la forme d'une nouvelle caractéristique de forme.

2.2. Intégration dans la bibliothèque

Nous avons jusqu'alors supposé l'existence d'une bibliothèque de caractéristiques génériques qui pouvaient être instanciées à la demande du concepteur. Il ne semble pas nécessaire de justifier de son existence : il faut bien stocker ces caractéristiques d'une manière ou d'une autre. La discussion porte surtout sur la manière de procéder. Pour cela, deux approches sont possibles :

- la bibliothèque est une suite de fonctions compilées et liées au code. C'est donc une partie de l'exécutable ;
- la bibliothèque est une base de données externe au programme, avec des méthodes associées qui seront interprétées au moment de leur utilisation.

Pour appréhender la différence, nous pouvons momentanément revenir à l'instanciation des caractéristiques. Dans le premier cas, l'instanciation provoque l'appel de fonctions. Les fonctions en question décrivent par exemple le moyen d'acquérir la position, l'orientation et les dimensions, de calculer la forme exacte de la caractéristique... Dans ce contexte, une

programmation objets offre un confort supplémentaire en permettant de mettre en commun des traitements (héritage) ou d'en particulariser (surcharge).

Si la bibliothèque est une base de données externe au programme, l'instanciation exige l'exécution de fonctions non compilées d'où la nécessité d'un interpréteur. D'un point de vue performances, la première approche est donc préférable.

Par contre, l'intégration de la caractéristique générique nouvellement décrite ne soulève aucun problème particulier si la bibliothèque est une base de données : il suffit d'y ajouter un élément. Si la bibliothèque est intégrée au logiciel sous la forme d'une liste d'objets compilés, l'intégration passe par les étapes suivantes :

- traduction du comportement en un langage de programmation (par exemple C++) ;
- compilation (négligeable car compilation séparée) ;
- édition des liens (négligeable car éditeur incrémental).

L'opérateur devra donc patienter, ce qui semble un argument en faveur d'une organisation en base de données. En réalité, le délai dû à la traduction ne doit pas être pris en compte, car il intervient aussi dans l'organisation en base de données, au moment de l'interprétation. Il intervient même autant de fois que la caractéristique est instanciée, ce qui n'est pas le cas avec la version compilée et compense donc le temps perdu à la compilation et à l'édition des liens.

Nous avons choisi une solution intermédiaire, c'est-à-dire que dans un premier temps, l'organisation en base de données et méthodes interprétées semble préférable, puis une version stable compilée sera créée. En effet, lors de la phase de mise au point, de nombreuses modifications sont nécessaires. Devoir recompiler à chaque changement paraît un réel obstacle. La solution interprétée sera donc temporairement adoptée. Par contre, quand la définition de la nouvelle caractéristique est donnée dans sa version définitive, elle sera transformée en une nouvelle classe munie de ses méthodes, compilée et liée au reste du code.

3. PROCÉDURE DE CRÉATION DE LA FORME

Parmi les méthodes qui déterminent le comportement de la caractéristique, nous nous sommes intéressés plus particulièrement à la création de la forme. Contrairement aux autres méthodes, la description de la forme d'une nouvelle caractéristique peut être réalisée de manière interactive, ce qui permet d'éviter au maximum à l'opérateur d'avoir à écrire des parties du programme. Si

d'autres méthodes s'avèrent nécessaires, nous pensons que l'intervention d'un programmeur est indispensable, sauf pour les deux méthodes abordées en fin de chapitre (extraction et *mapping*), qui sont générée automatiquement.

La création de la forme d'une nouvelle caractéristique générique peut être vue sous différents aspects, qui feront l'objet des trois paragraphes suivants. Dans un premier temps, il faut déterminer clairement ce que l'on veut représenter, pour choisir un type abstrait pour la forme. Dans un second temps, il faut une structure de données permettant d'exprimer le type abstrait. Enfin, il faut permettre la description interactive de la forme, par l'opérateur.

3.1. Modélisation de la forme

La forme d'une caractéristique générique est paramétrée par des dimensions, mais aussi par des quantités permettant de gérer la position et l'orientation lors de l'instanciation et également des entités de référence comme des faces de référence ou de base, des axes et des plans de symétrie, des repères locaux...

Cette vision des caractéristiques les fait apparaître comme des pièces paramétrées ; nous verrons par exemple que des techniques classiques de paramétrage, comme l'historique de construction ou la géométrie variationnelle, sont utilisées. Il y a cependant des particularités qui motivent une étude des caractéristiques de forme en tant que telles. La différence vient, entre autres, du fait que la topologie de la caractéristique varie à l'instanciation.

Nous avons vu, au chapitre 2, que la forme d'une caractéristique est modélisée par une surface génératrice. L'idée de mémoriser ainsi la forme d'une caractéristique est née des réflexions que nous avons menées au sujet de l'enrichissement de la bibliothèque. Nous retraçons donc ces réflexions, en résumant les trois modes de représentation possibles, à savoir :

- un volume correspondant exactement à celui de la caractéristique instanciée, au paramétrage près, c'est-à-dire dimensions, position et orientation ;
- un volume suffisamment surdimensionné pour garantir qu'il débouche, dans le cas d'un enlèvement de matière, ou atteigne la frontière de la pièce qui doit porter l'instance, dans le cas d'un ajout de matière ;
- une surface génératrice qui porte la peau de la caractéristique, sans limiter ses dimensions au niveau générique ; les frontières ne seront évaluées qu'à l'instanciation.

3.1.1. Volume correspondant à l'instance

La première solution pour représenter la forme d'une caractéristique générique est d'utiliser un volume correspondant à celui de l'instance. L'intérêt de cette solution est de faciliter l'instanciation, qui se résume alors à un simple dimensionnement suivi d'une opération booléenne. L'opération est une différence pour une caractéristique négative, c'est-à-dire un retrait de matière, et une union pour une positive, c'est-à-dire un ajout de matière. En particulier, il n'est pas nécessaire de vérifier l'absence d'interférence entre l'instance et une partie du volume qu'elle n'est pas supposée atteindre.

Les avantages de la modélisation par un volume correspondant exactement à celui de la caractéristique instanciée sont les suivants :

- après valuation des paramètres, il suffit de réaliser une unique opération booléenne pour instancier la caractéristique. Ceci ne constitue cependant un avantage que pour le développeur. Il n'y a là aucun apport pour l'opérateur ;
- les outils de modélisation géométrique dont on dispose permettent a priori une création simple du volume. Il s'agit cette fois d'un réel apport pour l'opérateur qui peut utiliser toute la puissance de l'environnement de CAO pour décrire la forme de la caractéristique.

Différents problèmes se posent toutefois. Une fois le paramétrage établi, la forme de la caractéristique ne peut que très peu évoluer au moment de l'instanciation. En particulier, les paramètres ayant une valeur constante, fixée a priori au moment de l'instanciation, ils ne peuvent pas varier dans une instance. Si l'on a, par exemple, paramétré une rainure par sa hauteur, sa largeur et sa longueur, on interdit une rainure inclinée, dont la hauteur varie, comme celle de la figure 4.1.

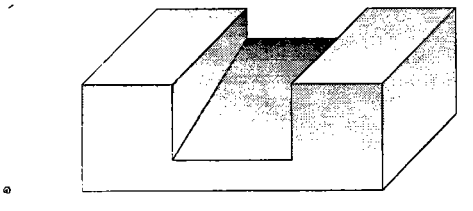


Figure 4.1 : Rainure dont la hauteur n'est pas constante.

Plus généralement, il paraît difficile d'avoir une forme de géométrie et topologie suffisamment générales pour qu'elle puisse être déformée en n'importe quelle rainure par modification des seuls paramètres. Par exemple, si on considère une rainure à trois faces planes, la forme ne peut pas être décrite de façon systématique. En effet, la base notamment peut varier en fonction de l'instanciation, comme l'illustre la figure 4.2 ci-dessous.

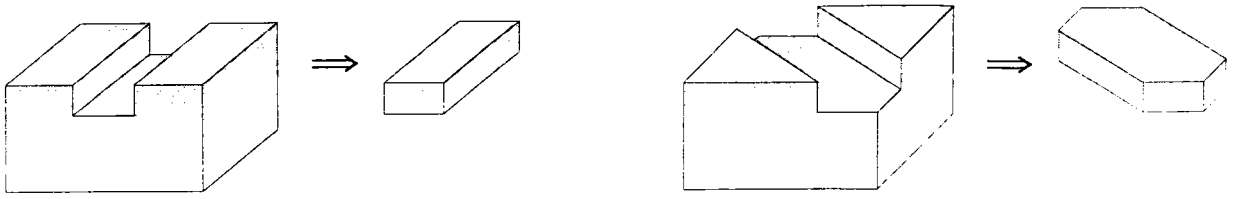


Figure 4.2 : Forme de la rainure variable selon l'instanciation.

Ces deux remarques soulignent le fait que le volume correspondant exactement à l'instance impose une orientation limitée. La topologie ne peut donc pas toujours être établie au niveau générique car la forme des faces dans l'instance peut varier. Pour pallier ce problème, il faudrait définir autant de caractéristiques génériques qu'il y a de formes possibles pour une instance, mais ceci n'est pas réaliste : nous ne pouvons pas prévoir toutes les possibilités.

De plus, lors de l'instanciation, il faut dimensionner le volume exactement pour que l'opération booléenne n'ajoute, ni ne retire trop de matière. Ceci impose à l'opérateur d'indiquer des dimensions très précises, alors qu'il semblerait plus naturel et plus facile d'utilisation, que la caractéristique se prolonge automatiquement jusqu'à atteindre la frontière de l'objet. Dans le cas d'un perçage non traversant, ceci revient à n'indiquer que la position du fond et le diamètre, le perçage "remontant" automatiquement jusqu'à déboucher.

En outre, il se peut, parce qu'elles ont des fonctions semblables, qu'on regroupe sous un même nom des caractéristiques ayant des formes différentes. Il n'est alors plus possible de représenter la caractéristique générique avec un même volume ; il faut donc soit prévoir autant de caractéristiques génériques qu'il y a de possibilités, ce qui retire tout intérêt à la généricité, soit autoriser une unique caractéristique générique regroupant toutes les variantes envisageables. C'est une solution très intéressante, mais qu'il n'est pas possible de réaliser en représentant la caractéristique par un volume. Pour reprendre l'exemple de la rainure, deux possibilités se présentent : soit prévoir autant de rainures génériques qu'il y a de rainures possibles (en U, en T, en V, à fond cylindrique...), soit donner une définition plus générale qui autoriserait n'importe quel profil pour la rainure instanciée.

Le fait que la caractéristique générique soit représentée par un volume limite enfin considérablement la possibilité qu'une face de l'instance adopte la géométrie d'une face du porteur. C'est par exemple le cas du fond d'une poche si cette dernière est placée sur une surface gauche et en épouse la forme. En effet, au niveau générique, on ne peut anticiper la forme du porteur et il n'est donc pas possible de spécifier complètement le volume de la future instance.

Pour résoudre ce problème, nous pourrions considérer la nature des surfaces comme un paramètre. Nous développerons cette possibilité dans le paragraphe 3.4.4.

Le nombre d'inconvénients cités est important par rapport aux avantages. Nous concluons donc que la méthode ne semble pas appropriée et ne sera pas retenue. Plutôt que d'utiliser un volume exact pour représenter la forme de la caractéristique, il est donc préférable d'utiliser une surface.

3.1.2. Volume surdimensionné

Avant de décider de modéliser la forme d'une caractéristique par une génératrice, c'est-à-dire une surface qui n'est pas fermée pour la majorité des caractéristiques secondaires, la deuxième solution possible est un recours à un volume surdimensionné, c'est-à-dire suffisamment grand pour dépasser (ou atteindre), à l'instanciation, les limites du porteur là où c'est nécessaire. La description interactive d'un volume est certainement plus aisée que celle d'une surface. De plus, le fait de surdimensionner le volume résout les deux premiers problèmes posés par le volume exact, c'est-à-dire ceux qui limitent l'orientation de la caractéristique. En effet, les rainures des figures 4.1 et 4.2 peuvent être engendrées par un même volume parallélépipédique surdimensionné. Pour deux raisons, nous préférons pourtant la modélisation par une surface génératrice.

D'abord, il n'est pas complètement immédiat, pour l'opérateur, d'initialiser à une quantité volontairement excessive certaines dimensions du volume. Dans le cas de la rainure traversante, les paramètres à surdimensionner sont la hauteur et la largeur. Or pour indiquer des valeurs trop grandes, il faut au moins une idée de la dimension de la pièce à rainurer, ce qui implique une démarche supplémentaire de l'opérateur. Évidemment, nous pourrions envisager un calcul automatique de ces mêmes quantités, mais l'argument suivant évite de se poser le problème.

En effet, la description d'un volume nécessite souvent davantage de paramètres. Par exemple, dans le cas d'une rainure, le volume correspondant est a priori un parallélépipède rectangle, dont les paramètres sont la largeur, la longueur et la hauteur. Il faut attribuer des valeurs à ces trois quantités à l'instanciation, manuellement ou de manière automatique. La surface génératrice correspondant à cette même rainure n'a qu'un paramètre, la largeur. Le problème de l'initialisation de la longueur et de la hauteur ne se pose donc plus.

3.1.3. *Surface génératrice*

Après réflexion, nous avons donc choisi de représenter les caractéristiques par leur surface génératrice. La définition en a été donnée au chapitre 2, mais nous rappelons tout de même qu'il s'agit d'une surface qui porte la peau de la caractéristique et qui n'est pas obligatoirement fermée. Elle ne détermine donc pas toujours un volume ; en particulier pour les caractéristiques secondaires, elle délimite souvent un demi-espace.

En fait, l'approche surfacique diminue le nombre de paramètres, ce qui paraît logique puisque pour décrire complètement un volume, il faut spécifier certaines parties qui n'auront pourtant aucun rôle actif à l'instanciation. Notons également que les trois premiers des cinq reproches faits à la première méthode disparaissent. La valeur d'un paramètre n'est plus figée au sein d'une instance donnée car le paramètre en question disparaît : c'est le cas de la hauteur d'une rainure, d'un trou, d'une poche, de la longueur d'une rainure, d'un congé ou d'un arrondi... Les variations de la topologie, qui empêchaient de prévoir la forme de l'instance sont gérées par les opérations booléennes. L'ajustage automatique de l'instance au porteur est réalisé, grâce à la non-limitation de la génératrice dans certaines directions. Par contre, les deux dernières limitations subsistent et seront abordées en fin de chapitre.

3.2. Structure de données

Pour mémoriser la forme de la surface génératrice d'une caractéristique au niveau générique, il existe deux possibilités :

- stocker la génératrice de manière évaluée dans un modèle géométrique ;
- conserver la procédure de création, c'est-à-dire la séquence de création des entités de base (sommets, arêtes, faces, volumes primitifs) et leur combinaison. La forme n'est pas évaluée ; pour l'obtenir, il faut exécuter la procédure.

3.2.1. *Modèle géométrique*

La première solution pour mémoriser la forme est de la stocker, au niveau générique, de manière évaluée de la génératrice et ses paramètres, dans un B-Rep. Cette solution présente les avantages bien connus de la représentation B-Rep. En particulier, le fait que la forme soit évaluée améliore les performances de visualisation par exemple. En outre, la connaissance explicite des

entités topologiques (sommets, arêtes, faces) permet de mémoriser facilement les contraintes géométriques, par des relations entre ces entités.

Cependant, au niveau générique, les paramètres ne sont pas connus. Au moment de l'instanciation, il reste alors à adapter la représentation générique au support considéré par ajustement de la forme au moyen de la géométrie variationnelle, en agissant uniquement sur les paramètres. Cependant, il n'est pas toujours aisé d'obtenir le volume souhaité, à partir d'un volume prédéfini : si la géométrie peut être adaptée, à condition qu'elle soit exprimée en fonction des paramètres, la topologie quant à elle ne peut pas être modifiée.

3.2.2. Procédure de création

Les difficultés à ajuster un B-Rep prédéfini nous ont amenés à imaginer une autre solution. Nous désirions éliminer les inconvénients de la solution précédente, mais pas ses avantages. Aussi avons nous choisi de mémoriser la génératrice, au niveau générique, par une procédure de création, c'est-à-dire un sous-programme indiquant la suite des instructions à effectuer pour obtenir la forme de la caractéristique générique. Au moment de l'instanciation, les paramètres sont connus et la procédure de création peut donc être exécutée et générer un B-Rep correctement dimensionné, placé et orienté.

Nous rappelons que nous faisons une distinction entre les caractéristiques génériques et les caractéristiques d'instanciation. Aussi, la procédure de création est stockée au niveau des caractéristiques génériques et elle génère une représentation évaluée, sous forme d'un B-Rep et d'un graphe conceptuel (décrit en détail au chapitre 2), qui est mémorisée au niveau des caractéristiques d'instanciation. Ceci permet donc de bénéficier des avantages du B-Rep exposés précédemment, au niveau des caractéristiques d'instanciation.

Une procédure de création très sommaire a été présentée au paragraphe 2.1 du chapitre 2, une version plus précise mais encore informelle est donnée ci-dessous, des exemples réels sont fournis en annexe E. Le chapitre 5 couvre les aspects techniques de la procédure de création de la forme.

```
CréationForme (Param1, Param2, ..., Pos, PremInst)
/*  Entrée :    Param1, Param2... = liste des paramètres de la forme                    */
/*                Pos             = repère local de la caractéristique                */
/*                PremInst        = booléen indiquant s'il s'agit d'une première instanciation */
/*  Sortie :    numéro de l'objet créé                                                  */
Début
   Consultation des valeurs des paramètres numériques.
   /*  à partir des objets de type "Paramètres", on obtient des valeurs numériques     */

   Création des sommets en fonction des valeurs des paramètres de dimensions
   et de position.
   /*  en appliquant le changement de repère donné par le positionnement de la CPAE   */
   /*  aux points exprimés dans le repère local de la caractéristique générique       */

   Création des arêtes.
   Création des faces.
   Création du volume.

   Si PremInst Alors
   /*  s'il s'agit de la première instanciation de la caractéristique                  */
      Création des géométries et des nœuds associés aux faces.
      Ajout des contraintes génériques.
   Sinon
      Mise à jour des géométries avec les équations des plans des nouvelles faces.
   FinSi
Fin
```

3.3. Description interactive

Pour générer la procédure de création de la forme de la génératrice d'une nouvelle caractéristique générique, nous avons considéré deux options : l'écriture de la procédure par l'opérateur ou l'espionnage de ses actions interactives.

L'écriture par l'opérateur implique une connaissance du langage de programmation. Toutefois, ce langage est assez sommaire, donc relativement facile à utiliser pour l'opérateur. Des outils de création d'entités de base (sommets, arêtes, faces, volumes) sont nécessaires pour lui éviter d'avoir une connaissance de la structure de données utilisée pour modéliser les caractéristiques. En effet, il ne doit pas se soucier de la manière dont sont chaînées les arêtes dans une représentation B-Rep par exemple. Il peut ensuite combiner des volumes élémentaires ou des demi-espaces avec des opérations booléennes. Là encore, la structure utilisée est transparente. En effet, la procédure écrite pourrait être la même, qu'il s'agisse d'un B-Rep ou d'un CSG. Les paramètres de la caractéristique ainsi décrite correspondent aux paramètres de la procédure de création, par exemple la longueur des arêtes, les dimensions des volumes de base...

Si la procédure de création est générée par l'espionnage des actions interactives de l'opérateur lors de la description d'une nouvelle forme, la liste chronologique des actions exécutées est mémorisée : à chaque menu utilisé correspond une instruction ou une suite d'instructions. Toutes

les valeurs qu'il donne deviennent des paramètres de la procédure auxquels il attribuera de nouvelles valeurs au moment des instanciations suivantes. De nombreux travaux ont été réalisés pour générer automatiquement un sous-programme à partir de l'espionnage d'actions interactives. Les résultats obtenus sont probants et réutilisables, nous n'avons donc pas approfondi davantage le sujet. Nous nous appuierons en particulier sur les travaux de [Pot 95], lorsque nous réaliserons la mise en œuvre.

Nous énonçons à présent différentes approches pour la description interactive de la génératrice. Leurs avantages et inconvénients respectifs sont exposés. Dans cette partie, nous ne nous intéresserons qu'à la partie description et non plus à la partie modélisation de cette surface.

3.3.1. Description par extrusion

La première méthode pour décrire interactivement une nouvelle caractéristique de forme, s'applique aux caractéristiques pouvant être représentées par une extrusion. Dans un premier temps, l'opérateur crée interactivement un profil, composé de lignes (segments, courbes...), fermé ou non. S'il est ouvert, le profil peut donc être prolongé par continuité, avec toutefois des nuances pour les lignes courbes. En effet, il est souvent délicat de prolonger des arcs de cercles par exemple [ChH 95]. Dans un second temps, l'opérateur décrit une trajectoire, finie ou non. La génératrice est créée en appliquant une extrusion du profil selon la trajectoire. Un profil ouvert et une trajectoire infinie permettent d'engendrer des faces qui se prolongent tant que nécessaire. Cette méthode est semblable à celle décrite dans [DZL 93], dans lequel le profil est toutefois fermé et la trajectoire finie. La figure 4.3 illustre le cas d'une marche, dont le profil est ouvert et la trajectoire infinie.

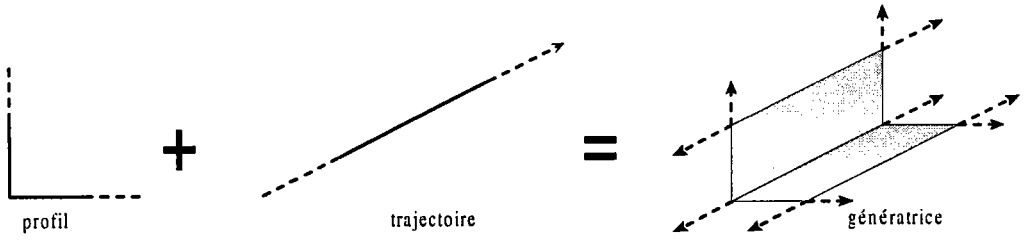


Figure 4.3 : Profil d'une marche.

La méthode peut être généralisée à un profil qui n'est pas forcement continu, comme celui de la figure 4.4. Dans cet exemple, la trajectoire est une demi-droite ; elle est donc limitée d'un côté. Si elle ne l'était pas, l'instanciation de la caractéristique provoquerait une division du volume porteur en deux composantes connexes. La validité d'une telle caractéristique, dont le profil est

discontinu et la trajectoire infinie, peut donc être discutée. Si une caractéristique ne doit pas diviser son porteur en plusieurs composantes connexes, l'insertion dans la bibliothèque de cette nouvelle caractéristique ne sera pas autorisée.

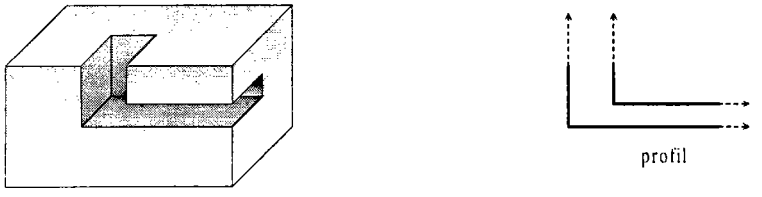


Figure 4.4 : caractéristique dont le profil est composé de parties disjointes.

Cette méthode a pour principal intérêt de couvrir une large gamme de caractéristiques usuelles : rainure en U, en T, en V, queue d'aronde, trou à section constante, marche, poche, congé, arrondi, chanfrein... En outre, il est relativement facile pour l'utilisateur de décrire un profil et une trajectoire. Nous retenons donc cette méthode de description interactive de la forme d'une nouvelle caractéristique.

3.3.2. Description par une liste de surfaces

Pour décrire la génératrice d'une nouvelle caractéristique générique, l'opérateur peut également identifier un certain nombre de faces, sur une pièce déjà créée. Cette méthode se décompose donc en deux étapes : tout d'abord la création d'une pièce complète, avec des outils de modélisation quelconque, puis la désignation, par l'opérateur, des faces constituant la caractéristique. Pour obtenir une forme générique, il faut alors prolonger ces faces, ce qui ne semble pas trivial au premier abord. Le problème principal de cette méthode de description de la forme est la vérification de la cohérence de ce qui est montré. En effet, si l'opérateur indique des faces quelconques de la pièce, elles peuvent ne pas être adjacentes, ce qui déterminerait une caractéristique non valide.

Une autre solution consiste à énumérer les surfaces formant la génératrice et à préciser des contours sur ces surfaces pour représenter les faces de la caractéristique. Cette solution n'est cependant pas très adaptée à la notion de prolongement "à l'infini" de certaines faces car des faces limitées par un contour sont forcément finies. Nous pourrions autoriser des contours ouverts, mais la description de contours sur des surfaces semble tout de même délicate ; cette solution n'est citée que dans un souci d'exhaustivité et ne sera pas détaillée davantage. Cette méthode est orientée représentation B-Rep ; elle est très générale mais pénible à utiliser car elle manipule des notions de bas niveau.

Pour rendre cette méthode plus conviviale, nous conservons le principe, mais en y associant une couche de dialogue pour faciliter le travail de l'opérateur. Ce dernier donne tout d'abord une liste de surfaces sécantes, réelles ou virtuelles. Les surfaces constituant la génératrice sont réelles et les plans de symétrie sont des surfaces virtuelles. Ensuite, l'opérateur précise la position de la matière par rapport à ces surfaces. La génératrice est déduite de calculs d'intersections et de règles de cohérence, éventuellement avec l'assistance de l'opérateur.

Nous concluons finalement qu'une description de la génératrice de la caractéristique par un ensemble de surfaces est délicate, dans la mesure où les outils de manipulation des surfaces sont difficiles d'accès. La puissance de ces outils ne compensant pas leur lourdeur, nous n'avons retenu aucune des méthodes proposées.

Les outils pour la manipulation de volumes sont beaucoup plus courants. C'est pourquoi nous décidons d'utiliser des volumes plutôt que des surfaces pour décrire la génératrice, ce qui est plus facile d'un point de vue interactif.

3.3.3. Description par un volume ouvert

La dernière solution permet de décrire la génératrice à l'aide d'un volume. Le principe est de construire un volume correspondant à la forme de la caractéristique et d'y supprimer certaines faces pour qu'il s'étende autant que possible. Les faces adjacentes à celles qui sont supprimées sont prolongées de manière naturelle, c'est-à-dire suivant leur surface porteuse. Par exemple, supposons que l'on veuille définir une poche. Le volume correspondant à cette caractéristique générique est un parallélépipède. Pour étendre la poche vers le haut, c'est-à-dire du côté où elle débouche hors du support, on supprime la face du dessus, comme le montre la figure 4.5 (la face à supprimer est hachurée). Un exemple de procédure de création obtenue par cette méthode de description est donné en annexe E.

Figure 4.5 : Suppression d'une face d'une boîte pour obtenir la génératrice de la poche.

Cependant, l'extension d'un volume n'est pas toujours sans poser problème. En effet, la suppression d'une face, ou le déplacement de celle-ci vers les limites de l'espace, peut engendrer

des défauts de cohérence dans le volume. C'est le cas notamment quand deux faces adjacentes à la face supprimée se coupent avant la frontière de l'espace. Par exemple, une queue d'aronde ne peut pas s'étendre au-delà d'un certain seuil, comme l'indique la figure 4.6. L'extension d'une surface cylindrique, et plus généralement d'une surface non plane, pose également un certain nombre de problèmes. L'extension du volume sera donc admise tant que la cohérence est conservée. Remarquons toutefois que la résolution de ce problème est complexe et qu'elle exige une étude approfondie, dont on trouvera des éléments dans [ChH 95].

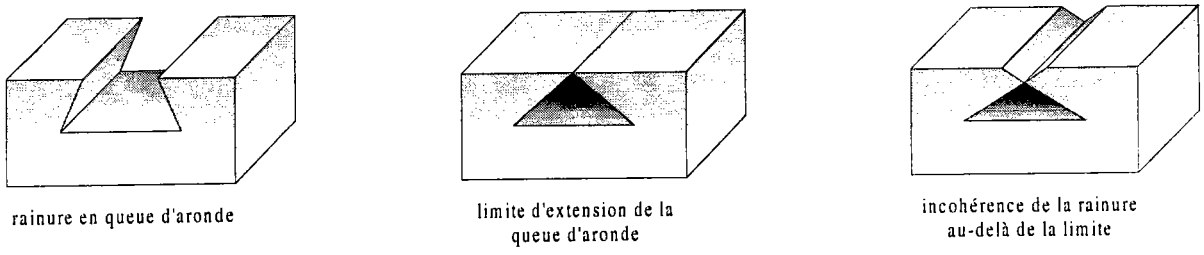


Figure 4.6 : Extension d'une queue d'aronde.

Il faut noter que cette approche, basée sur la manipulation des volumes, n'est pas un retour à une modélisation de la caractéristique par un volume surdimensionné. Cette solution permet bien de décrire la surface génératrice d'une nouvelle caractéristique ; elle ne constitue qu'un artifice interactif pour y parvenir.

L'avantage de cette solution est la facilité de description de la forme pour l'opérateur, puisqu'il peut utiliser tous les outils de modélisation géométrique. Pour supprimer les faces du volume, il suffit de les désigner. La procédure de création de la forme sera donc composée de deux parties : la création du volume puis la suppression éventuelle de certaines faces. Nous retenons donc également cette solution pour décrire interactivement la génératrice, principalement à cause de la variété des formes qu'elle permet de décrire.

3.4. Compléments de définition

Nous rappelons que la forme d'une caractéristique générique est conservée par une surface génératrice, qui est modélisée par une procédure de création. Les techniques pour la description interactive d'une nouvelle caractéristique de forme, donc d'une génératrice, sont l'extrusion d'un profil et la suppression de faces dans un volume.

L'analyse du modeleur à base de caractéristiques menée dans le chapitre 2 a montré que la procédure de création est supposée générer à la fois un B-Rep de la génératrice et son graphe

conceptuel. Les paramètres de la procédure correspondent aux paramètres génériques de la nouvelle caractéristique. Au cours de la description interactive de la forme, il faut donc établir cette liste de paramètres. Nous traiterons de la spécification des paramètres dans le paragraphe 3.4.1.

Le graphe conceptuel mémorise la liste des contraintes génériques de la génératrice. Ces contraintes sont obtenues pendant la description interactive de la forme soit par détection automatique, soit pas spécification explicite de la part de l'opérateur. Le paragraphe 3.4.2 est donc consacré à l'obtention du graphe conceptuel.

Enfin, nous ferons deux remarques complémentaires sur la rigidité de la description de la forme, qui peut être assouplie, si besoin est, quant au nombre (paragraphe 3.4.3) et à la nature (paragraphe 3.4.4) des faces de la génératrice.

3.4.1. Spécification des paramètres

Dans la phase de description de la forme d'une nouvelle caractéristique, nous supposons travailler dans le repère local de la caractéristique générique. L'opérateur peut donc difficilement spécifier d'autres paramètres que ceux de dimensions, en particulier ceux d'orientation et de position. Ce n'est pas très gênant car nous jugeons généralement inutile d'avoir ces spécifications au niveau de la caractéristique générique, les méthodes de positionnement et d'orientation étant communes à toutes les caractéristiques. Les paramètres de position et d'orientation apparaîtront donc de la même manière pour toutes les caractéristiques, en paramètre de la procédure de création, sous forme d'un repère local. Il faut donc prévoir un outil au niveau des mécanismes d'instanciation, qui permette d'exprimer les contraintes de positionnement et d'orientation de l'utilisateur en mouvements du repère local. Par exemple, si l'opérateur désire que le fond d'une rainure soit à une certaine distance d'une autre face de la scène, l'outil en question doit être capable d'en déduire une nouvelle position du repère local. Quelques questions se posent alors :

- il est probable que des contraintes porteront sur les faces de l'instance. Quelle influence cela a-t-il sur le modèle utilisé pour la génératrice ? Cela exclut-il en particulier une modélisation par arbre de construction, dans lequel la notion de face n'est pas immédiatement présente ?
- est-il possible de transformer toutes les contraintes en mouvements du repère local ?
- toutes les caractéristiques génériques peuvent-elles être exprimées dans un repère local, en particulier les raccordements ?

Nous pouvons apporter quelques éléments de réponse à ces interrogations. En ce qui concerne le premier point, la notion de face n'est pas indispensable, puisque les contraintes s'appliquent sur des géométries. Or, même dans un arbre de construction, les géométries sont connues car les équations des surfaces sont forcément disponibles. Par contre, s'il existe des contraintes portant sur des arêtes, elles sont plus difficiles à exprimer dans une représentation de type CSG. En effet, toutes les arêtes ne sont pas présentes dans les volumes primitifs, notamment celles qui apparaissent à la suite d'une combinaison booléenne entre des volumes. Une représentation CSG de la génératrice nuit donc à la prise en compte des contraintes et donc également à la qualité du dialogue. C'est la raison pour laquelle nous n'avions pas envisagé cette hypothèse dans la partie 3.2, consacrée à la modélisation de la génératrice.

Le deuxième problème posé est la transformation d'un ensemble de contraintes en mouvements du repère local. La principale difficulté est de déterminer si le système n'est ni sous-contraint, ni sur-contraint. Par exemple, il apparaît rapidement que deux contraintes d'angles sont insuffisantes pour orienter un objet ; il semblerait donc qu'il en faille trois. Cependant, une étude montre que trois contraintes d'angles risquent souvent d'être incompatibles et provoquent une sur-contrainte [Kla 96].

Enfin, le troisième point évoqué concerne les raccordements. Il paraît en effet relativement simple de définir un raccordement entre des faces d'un même objet (congé, chanfrein...). Par contre, entre deux objets différents, il est plus délicat de spécifier un repère local dans lequel le raccordement pourrait être exprimé.

Revenons aux paramètres de dimensions, qui seront exprimés par l'opérateur au moment de la description interactive de la forme. Dans le cas d'une description par extrusion, les paramètres de dimensions de la caractéristique correspondront vraisemblablement aux dimensions du profil et à certains paramètres de la trajectoire. L'opérateur doit donc donner un nom à chaque paramètre utile.

Dans le cas d'une description de la forme par un volume, ouvert par suppression de faces, les paramètres de la caractéristique nouvellement définie sont les mêmes que ceux du volume de départ. Les paramètres correspondant aux faces détruites sont cependant supprimés de la liste des paramètres. Cette suppression de paramètres peut être effectuée explicitement par l'opérateur qui indique alors, dans la liste des paramètres, ceux qu'il ne souhaite pas conserver. Une seconde solution consiste à retirer automatiquement les paramètres liés à une face supprimée. Par exemple, la définition d'une poche est basée sur une boîte, dont on supprime la face supérieure. La boîte est définie par trois paramètres : largeur, longueur et hauteur. La suppression de la face supérieure implique la suppression du paramètre "hauteur" dans la liste des paramètres de la

poche. Par analogie, si le volume de départ est constitué d'une combinaison de volumes primitifs, la liste des paramètres de la caractéristique est l'ensemble des paramètres des volumes primitifs, de laquelle sont supprimés ceux correspondant aux faces détruites.

3.4.2. Contraintes génériques

Quelle que soit la méthode choisie pour décrire la forme, il n'y a pas de problème pour générer le B-Rep correspondant. Cependant, comme nous l'avons indiqué au chapitre 2, la procédure de création de la forme permet de générer, en plus du B-Rep, le graphe conceptuel de la génératrice ou graphe de contraintes. Or ce graphe mémorise les contraintes génériques de la caractéristique. Il faut donc spécifier ces contraintes au moment de la description de la forme, pour pouvoir créer le graphe en même temps que le B-Rep.

Les contraintes génériques que l'on peut imposer aux caractéristiques permettent de préciser une partie de leur comportement. On peut par exemple spécifier une contrainte du type : la largeur de la nouvelle caractéristique doit, dans toutes les instances, être supérieure à une valeur seuil minimale. Cette valeur est peut-être liée à des contraintes de fabrication dans la mesure où une caractéristique de largeur inférieure ne serait peut-être pas usinable.

Ainsi, on peut imposer des contraintes numériques sur tous les paramètres de dimensions, sur des distances ou sur des angles. Les contraintes géométriques s'expriment sur des entités géométriques telles que les surfaces, voire des entités plus complexes (composition d'entités de base). Elles traduisent des relations de parallélisme, perpendicularité, coaxialité, concentricité, adjacence entre surfaces convexes, concaves ou tangentes. Certaines contraintes enfin sont internes à une seule entité comme l'horizontalité ou la verticalité d'un plan.

Si la forme de la nouvelle caractéristique est créée par extrusion, les contraintes géométriques 2D peuvent être détectées automatiquement, pendant la description interactive du profil. Elles sont donc insérées spontanément dans le graphe conceptuel, reliant alors les nœuds correspondants aux lignes du profil. Cependant, il se peut que certaines contraintes soient détectées, mais qu'elles ne soient pas nécessaires pour déterminer la caractéristique. L'opérateur doit pouvoir supprimer, dans le graphe, les contraintes qu'il ne souhaite pas conserver. La première solution est de demander confirmation à l'opérateur, lors de la détection d'une contrainte, avant de l'insérer dans le graphe. Une autre solution est de créer la contrainte automatiquement (si l'on juge par exemple qu'une confirmation systématique est trop pénible pour l'opérateur) ; dans ce cas, il faut donner les moyens à l'opérateur d'agir sur le graphe a

posteriori, pour éliminer les contraintes qui ne l'intéressent pas. Nous proposons donc aussi d'éditer le graphe conceptuel pendant la description de la forme, comme suggéré par Salomons *et al.* [SSJ 94].

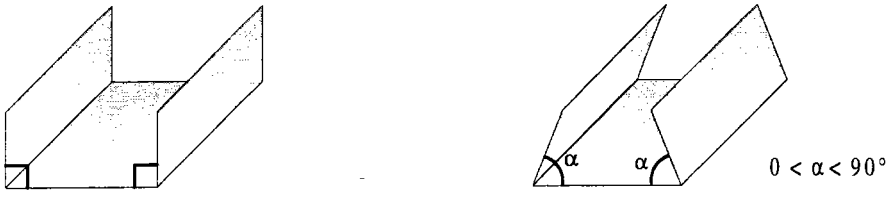
Si la forme de la génératrice est créée à partir d'un volume duquel des faces sont supprimées, les contraintes génériques du volume de départ serviront de base pour créer le graphe conceptuel de la nouvelle caractéristique. Lorsque les faces sont supprimées, les contraintes correspondantes, c'est-à-dire celles qui portent sur les géométries de ces faces (ou des arêtes éventuellement supprimées), sont éliminées du graphe, le cas échéant. De même que dans le cas de l'extrusion, il faut autoriser l'opérateur à revenir sur certaines contraintes, qu'il ne souhaite pas conserver.

Dans les deux cas, c'est-à-dire que la description interactive corresponde à une extrusion ou à un volume ouvert, l'opérateur peut ajouter, en plus des contraintes géométriques créées automatiquement, des contraintes génériques équationnelles, du type : "la hauteur du trou est supérieure à un pourcentage du rayon". Nous rappelons que toutes les contraintes sont stockées dans une même liste de contraintes génériques, mais que seules les contraintes géométriques apparaissent dans le graphe conceptuel.

Nous remarquons que certaines contraintes génériques peuvent être implicites, c'est-à-dire qu'elles sont automatiquement vérifiées si l'on exécute la procédure de création de la forme de la caractéristique. Par exemple, si l'on crée une boîte, les faces sont toujours deux à deux perpendiculaires ou parallèles. Le fait de mémoriser de telles contraintes génériques semble donc inutile, puisque pour instancier une boîte, la même procédure de création sera toujours exécutée. En fait, ces contraintes implicites sont conservées explicitement car elles sont nécessaires pour la gestion des contraintes. En effet, il ne faudrait pas autoriser une modification, lors de la propagation de contraintes, qui serait remise en cause par l'exécution de la procédure de création. De plus, le fait de conserver explicitement ces contraintes favorise la possibilité d'évolution du logiciel vers une gestion variationnelle des contraintes. En effet, la géométrie variationnelle autorise la déformation d'un volume directement, sans passer par l'exécution de sa procédure de création. Comme les contraintes génériques doivent toujours être respectées, il faut les conserver.

Les contraintes permettent en outre de définir des familles de caractéristiques. En effet, l'idée est de concevoir une nouvelle caractéristique à partir d'une caractéristique présente dans la bibliothèque, en indiquant simplement des contraintes différentes. Par exemple, la génératrice d'une rainure est généralement composée de trois faces planes, dont deux sont orthogonales à la

troisième, comme le montre la figure 4.7(a). Ceci est exprimé par des contraintes qui imposent l'existence de deux angles droits. On pourrait définir une nouvelle caractéristique, composée de trois faces planes également, mais en fixant simplement que les angles entre deux faces symétriques et la troisième doivent être identiques, compris entre deux valeurs (respectivement minimum et maximum), comme sur la figure 4.7(b). La queue d'aronde serait ainsi ajoutée à la bibliothèque, simplement en modifiant les contraintes génériques de la rainure.



(a) rainure orhogonale (b) queue d'aronde

Figure 4.7 : Contraintes différentes pour la rainure ou la queue d'aronde.

3.4.3. Nombre de faces variable

Pour que la description générique de la forme soit la plus générale possible, le nombre de faces de l'instance pourrait être variable. En effet, pour une poche par exemple, le nombre de faces latérales n'est pas fixé a priori et est déterminé par la forme de la base de la poche instanciée. Si la base de la poche est triangulaire, le nombre de faces latérales est de trois ; si la base est quadrangulaire, il est de quatre... Ceci n'empêche pas cependant qu'il y ait des faces stables, qui existent toujours, comme le fond de la poche, en plus de celles dont le nombre varie. Dans la caractéristique générique, il faut donc prévoir un nombre constant de faces *N*, c'est-à-dire que toutes ses instances auront au moins *N* faces, et spécifier que l'opérateur pourra ou devra ajouter des faces lors de l'instanciation.

Ceci implique qu'à l'instanciation, l'opérateur doit définir complètement les faces supplémentaires, ce qui lui impose une charge de travail importante. C'est donc peu pratique, surtout si l'opérateur souhaite instancier plus d'une fois cette même caractéristique. Il n'y aurait alors plus guère d'intérêt à la généricité. Cette option est d'autant moins valable que si la bibliothèque est bien conçue, il ne serait pas plus compliqué pour l'opérateur de construire une caractéristique générique "poche" très générale et de la particulariser en une autre poche correspondant précisément à celle dont il a besoin. Un nombre variable de faces nous paraît donc une option à retenir mais seulement pour des caractéristiques servant à en définir d'autres, c'est-à-dire qui seront précisées ou dérivées, mais qui ne peuvent pas être instanciées directement.

Pour revenir à la programmation objet, ces caractéristiques correspondent donc à des classes abstraites.

3.4.4. Nature des faces variable

De même que le nombre de faces peut varier, la nature des faces pourrait ne pas toujours être précisée au niveau générique. Ceci autoriserait par exemple une rainure générique à trois faces, deux faces planes et une troisième de nature variable. Nous regrouperions ainsi, dans une même catégorie, une rainure à trois faces planes et une rainure à deux faces planes et une face cylindrique. Cependant les conclusions concernant un nombre variable de faces s'appliquent ici aussi, c'est-à-dire que ce qui n'a pas été spécifié au niveau générique doit forcément l'être à l'instanciation, ce qui la rend d'autant plus lourde.

Il y a un cas dans lequel il est toutefois utile de ne pas préciser la nature des faces au niveau générique : c'est quand la nature dépend du porteur, par exemple quand le fond de la rainure doit suivre la face qui porte la rainure. Dans ce cas, il y a moins de problèmes de dialogue, car diverses informations peuvent être déduites de la face qui porte la caractéristique (nature, certains paramètres...). Il reste cependant à gérer un certain nombre de problèmes calculatoires pour construire une génératrice cohérente à partir des faces décrites au niveau générique et de la face extraite du porteur.

4. DÉFINITION DU RESTE DU COMPORTEMENT

La description de la forme d'une caractéristique contribue à déterminer une partie de son comportement. Dans le chapitre 2, nous proposions de compléter le comportement par des méthodes d'extraction ou de génération de modèles applicatifs. Il faut également prévoir ces méthodes lors de la définition de nouvelles caractéristiques génériques.

4.1. Règles d'extraction

Quand l'opérateur définit une nouvelle caractéristique, il décrit sa génératrice. Elle pourra donc être appariée avec les faces du modèle géométrique, comme nous le suggérions au chapitre 2. Pour l'extraction telle que nous l'envisageons, il n'y a donc rien de particulier à

spécifier au moment de la définition d'une nouvelle caractéristique générique. La procédure d'extraction est commune à toutes les caractéristiques.

4.2. Règles de transformation (mapping)

Lorsqu'un modèle de conception est transformé en modèle d'application, il s'agit d'une transformation globale du modèle et non d'une transformation individuelle de toutes les caractéristiques qui le composent. Il est donc délicat, au niveau d'une caractéristique générique, de donner des règles de transformation.

Nous pouvons cependant citer deux cas particuliers, dans lesquels la génératrice, qui permet de décrire la forme d'une caractéristique générique, peut être exploitée par les applications : la génération de gammes d'usinage et la visualisation.

En effet, dans le chapitre 2, nous avons montré que la génératrice permet de choisir l'outil et sa trajectoire, pour l'usinage de la caractéristique. Or cette génératrice est définie lors de la création d'une nouvelle caractéristique générique dans la bibliothèque. Elle est donc exploitable de la même manière que celle des caractéristiques prédéfinies.

La visualisation peut exploiter la connaissance des caractéristiques de forme pour obtenir son propre modèle de visualisation, en tenant compte de trois aspects différents [LaV 97]. Le premier objectif, pour optimiser la visualisation, est d'avoir un maximum de faces planes dans le modèle de visualisation. Pour cela, les caractéristiques sont divisées en deux catégories : celles dont la génératrice est composée de faces planes uniquement et celles qui contiennent des surfaces gauches. L'optimisation suggérée pour la première catégorie est de réaliser un découpage *(clipping)* des faces visibles par les faces non visibles. Les caractéristiques de la seconde catégorie sont décomposées en un ensemble de faces planes et un ensemble de faces gauches. Deux algorithmes différents sont appliqués pour visualiser ces deux ensembles de faces.

Le deuxième aspect pris en compte par la visualisation est le principe de construction qui lie les caractéristiques. Le but est de supprimer du modèle de visualisation un certain nombre de composants non visibles. Ainsi, le trou lamé, par exemple, est composé de deux cylindres. Si l'un n'est pas visible l'autre ne le sera pas non plus.

Enfin, le dernier aspect exploité par la visualisation est la connaissance d'un principe d'assemblage. Si on sait, par exemple, qu'une face est percée de n trous, on peut déduire que si la

face n'est pas visible, aucun trou ne l'est. On peut donc les omettre dans le modèle de visualisation.

5. CONCLUSION

Nous avons vu dans ce chapitre, qu'il était nécessaire de pouvoir enrichir la bibliothèque. En effet, comme les caractéristiques prédéfinies ne peuvent pas couvrir toutes les spécialités des différents utilisateurs du système de conception, il est indispensable de permettre aux divers opérateurs de définir leurs propres caractéristiques. Pour cela, il faut donc normaliser la structure qui permet de les stocker dans la bibliothèque.

Nous avons choisi de représenter la forme des caractéristiques par une surface génératrice. La structure de données qui permet de modéliser cette génératrice est une procédure de création. Elle peut être obtenue de deux manières. Pour la première solution, l'opérateur écrit lui-même la procédure, ce qui implique qu'il connaisse le langage de programmation. La seconde solution consiste à générer automatiquement la procédure de création, en espionnant les actions interactives de l'opérateur pendant la description de la forme.

La description interactive de la surface génératrice de la caractéristique sera faite soit par l'extrusion d'un profil 2D, soit par un volume, qui sera étendu selon certaines directions par suppression de faces.

Nous constatons toutefois que si la procédure de création est obtenue par espionnage des actions interactives de l'opérateur, il se peut qu'elle comporte des opérations inutiles, témoignant par exemple des hésitations de l'opérateur lors de la construction de la nouvelle caractéristique. Il faudrait donc prévoir de simplifier la procédure, en supprimant les instructions qui ne sont pas constructives. Aussi, la simplification de la procédure de création de la forme constitue une perspective de recherche prioritaire.

La méthode décrite pour créer la forme d'une nouvelle caractéristique générique secondaire peut s'appliquer également pour des caractéristiques primaires, en ce qui concerne la description de la surface génératrice, qui dans ce cas sera finie. Dans le premier cas, le profil 2D sera fermé et la trajectoire d'extrusion finie. Dans le second cas, aucune face ne sera enlevée au volume pour le prolonger.

D'autres méthodes de définition de nouvelles caractéristiques génériques pourraient autoriser que la forme ne soit pas complètement déterminée. Dans ce cas, il faudrait compléter la définition, au niveau générique ou pour chaque instance. Nous envisageons de prendre en compte les aspects fonctionnels. Par exemple, si l'opérateur souhaite instancier une rainure dont les trois faces sont de nature inconnue, il précise que sa fonction est un guidage en translation d'une autre pièce, déjà construite, dont le profil est donc connu (composé de surfaces fonctionnelles). On peut alors en déduire la forme et les dimensions de l'instance. On peut imaginer la même façon de procéder avec d'autres caractéristiques comme la languette, le trou fileté (avec une vis de même diamètre et de même pas de vis), la marche, la poche...

En ce qui concerne les caractéristiques dites de raccordement comme le congé ou le chanfrein, on peut certainement prendre en compte l'aspect fabrication. On peut ainsi prévoir qu'on ne pourra pas usiner d'arêtes vives par exemple et que la présence d'un congé est donc nécessaire.

Chapitre 5 - MISE EN ŒUVRE

1. INTRODUCTION

Pour la mise en œuvre des propositions faites dans les chapitres précédents, nous avons choisi d'utiliser une approche orientée objets. À plusieurs reprises, nous avons montré l'intérêt d'une telle approche pour réaliser les concepts présentés.

En effet, nous rappelons la volonté de structurer la bibliothèque de caractéristiques de manière hiérarchique, en décomposant les caractéristiques en primaires et secondaires et les caractéristiques secondaires en positives et négatives. Cette hiérarchie peut être représentée par l'héritage entre les classes représentant tous ces types de caractéristiques.

En outre, la volonté de réunir dans une caractéristique un ensemble de paramètres et un comportement correspond à la notion d'encapsulation des langages orientés objets.

De plus, nous désirons garantir l'extensibilité du logiciel, pour permettre la définition de nouvelles caractéristiques par l'utilisateur. Aussi, lors de l'enrichissement de la bibliothèque, il est intéressant de pouvoir réutiliser tout ou partie d'une caractéristique prédéfinie, ce qui est rendu possible par la programmation objets.

Les méthodes orientées objets sont basées sur un processus créatif décomposé en trois activités : la compréhension du problème, l'invention de la solution et enfin sa réalisation. La création d'un logiciel est donc accomplie en trois phases : l'analyse, la conception et la programmation.

L'analyse permet, à partir d'une spécification, d'aboutir à un modèle globale pour la réalisation du système informatique et à des propriétés. Elle se fait en collaboration étroite avec

les experts du domaine. Le produit de l'analyse doit être soumis aux experts. Les objets du modèle doivent correspondre à des objets du domaine. L'analyse est donc le but à atteindre.

La conception est la façon d'atteindre le but. À partir du modèle d'analyse, le modèle de conception est créé. Les décisions d'ordre informatique sont de deux types : l'étude algorithmique et la représentation physique.

Une première partie de ce chapitre est consacrée à l'analyse orientée objets pour la réalisation de notre modeleur à base de caractéristiques. Une seconde partie donnera quelques précisions sur certains choix qui ont été faits pour la mise en œuvre.

2. ANALYSE ORIENTÉE OBJETS

La méthode orientée objets utilisée dans ce chapitre est la méthode OOAD *(Object-Oriented Analysis and Design)* [Boo 91]. Son formalisme est décrit dans l'annexe D. Nous donnons d'abord dans ce paragraphe un schéma général des classes constituant le système, non détaillé, qui permet d'avoir une vue du modèle objet à un niveau d'abstraction élevé. Nous verrons que chaque classe abstraite peut être détaillée, pour former une catégorie. Nous donnerons donc le diagramme des catégories, puis les diagrammes des classes par catégories, qui représentent un niveau d'abstraction plus bas. Un dernier schéma permet d'expliciter les liens existant entre les catégories.

Avant d'entamer réellement ce chapitre, il nous semble important de donner quelques précisions sur le vocabulaire employé. Dans le chapitre 2, nous avons défini les termes de caractéristiques génériques et caractéristiques d'instanciation. Le terme d'instanciation y a également été défini et il sera utilisé dans ce chapitre de la même façon, c'est-à-dire comme étant la création et l'insertion d'une caractéristique dans le modèle de conception, et non au sens du mécanisme d'instanciation orienté objets. Notons également que les caractéristiques d'instanciation seront identifiées par le terme caractéristiques *prêtes à l'emploi*, pour éviter toute confusion.

2.1. Diagramme général des classes

Le modèle créé par notre système à base de caractéristiques est composé d'un ensemble de volumes. La première classe identifiée est donc la classe *Modèles*, dont la responsabilité principale est de permettre l'ajout d'un nouveau volume au modèle courant. Il s'agit d'une simple insertion d'un élément à un ensemble, car les volumes sont supposés être indépendants et non interférents. On pourrait toutefois vérifier que certaines conditions sont respectées, lors de l'ajout d'un nouveau volume à la scène, mais nous n'avons précisé aucune règle pour cela. Le modèle doit en outre assurer sa propre sauvegarde, le chargement d'un autre modèle, ou encore la remise à zéro du modèle courant pour en créer un nouveau.

Un volume est constitué d'un ensemble de composants, ajoutés l'un après l'autre. L'objet de notre système de conception est de représenter les volumes à l'aide de caractéristiques de forme. Cependant, dans un souci de généralité, nous avons prévu d'ajouter à l'historique de construction du volume les opérations booléennes et les transformations géométriques. C'est pourquoi nous étendrons la notion de caractéristiques à la notion de composants, permettant de représenter ces trois types d'informations. Dans ce paragraphe, nous ne détaillons toutefois pas davantage les composants, pour mettre l'accent sur le fait que notre système gère avant tout les caractéristiques. En particulier, dans la figure 5.3, nous verrons apparaître la classe *Caractéristiques* plutôt que la classe *Composants*.

Un volume doit permettre d'accéder à sa représentation B-Rep, combinaison de tous ses composants. Il doit en outre pouvoir renseigner son état, qui peut être réel ou virtuel. En effet, un volume devient virtuel dès l'instant où il a été combiné à un ou plusieurs autres volumes. Nous remarquons que si le premier composant d'un volume est une caractéristique, c'est forcément une caractéristique primaire. Notons également que l'ajout d'un composant ne se limite pas à sa seule insertion dans un ensemble, mais qu'il déclenche en outre les méthodes permettant au composant de s'évaluer au sein de son volume porteur, notamment en calculant sa forme.

Les caractéristiques qui constituent le volume sont soumises à des contraintes. La classe *Caractéristiques* a donc la responsabilité d'ajouter des contraintes. Une fois la contrainte insérée, il faut vérifier qu'elle est satisfaite ; cette vérification est du ressort du gestionnaire de contraintes. Remarquons que l'ajout de contraintes génériques n'est pas réalisée par l'opérateur, mais directement lors de la création de la forme de la caractéristique. Cette méthode sera donc utilisée pour créer le graphe conceptuel.

Une caractéristique est en outre déterminée par un certain nombre de paramètres. Une caractéristique a donc la responsabilité de donner et de modifier les valeurs de ses paramètres, qui sont identifiés par un nom.

Nous rappelons que le graphe conceptuel permet de stocker les contraintes géométriques. Il contient initialement les contraintes génériques mais est ensuite étoffé avec les contraintes d'instanciation. Les arcs du graphe correspondent à ces contraintes et les nœuds aux géométries. Nous identifions donc les classes *Nœuds* et *Géométries*. Il nous paraît important, avant de poursuivre, de définir clairement la notion de géométrie. Nous désignerons par géométrie : pour une face, la surface qui la porte (représentée par son équation), pour une arête, la droite et pour un sommet, le point 3D. Nous distinguons donc a priori trois types de géométries. Cependant, dans la maquette réalisée, nous avons choisi de ne traiter que les géométries correspondant à des faces planes : elles sont donc représentées par l'équation d'un plan. Dans une géométrie, deux autres informations sont conservées : une liste des faces qui s'appuient sur la géométrie et une référence vers le nœud du graphe, qui permet de contraindre la géométrie. La figure 5.1 schématise la représentation des géométries.

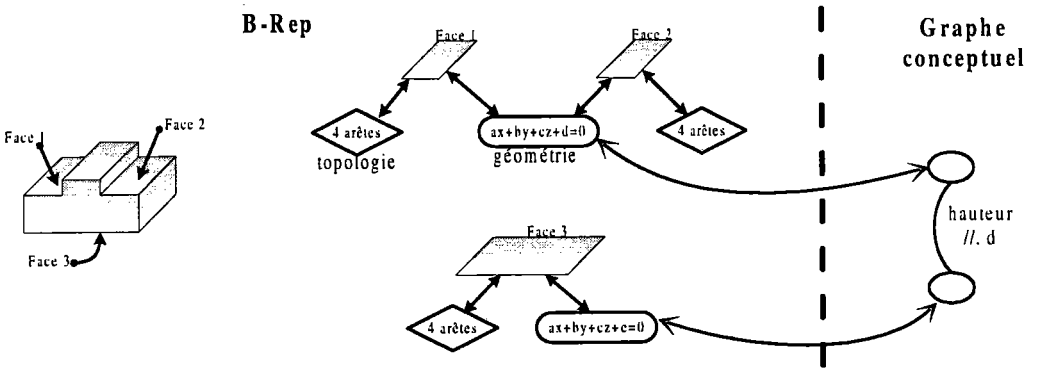


Figure 5.1 : Représentation des géométries.

Nous préférons appliquer les contraintes à des géométries plutôt qu'à des faces car lors des opérations booléennes, les faces des opérandes sont détruites et les nouvelles faces du résultat sont créées, mais elles ont les mêmes géométries. Il est donc important de ne pas perdre, en supprimant les faces, les contraintes qui y sont définies. Or, si elles portent sur les géométries, elles ne seront pas affectées par la suppression de faces. Les géométries seront simplement mises à jour avec les équations des nouvelles faces.

De plus, lors d'une opération booléenne, une face peut être divisée, mais la géométrie des faces obtenues sera la même. C'est le cas par exemple, lors de l'instanciation d'une languette, comme le montre la figure 5.2. La face dans laquelle se trouve la languette est scindée. Si les

contraintes portaient sur la face de départ, elles devraient alors être associées aux deux nouvelles faces. Comme la géométrie des deux nouvelles faces est la même que celle de la première, le report des contraintes se fait spontanément.

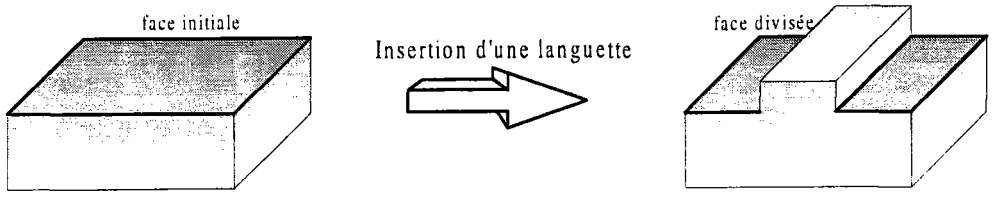


Figure 5.2 : Division d'une face lors de l'instanciation d'une languette.

Nous rappelons enfin que lors de la gestion des modifications, il faut détecter les caractéristiques qui interfèrent géométriquement. Pour cela, nous avons proposé de calculer l'intersection entre leurs englobants. C'est pourquoi nous identifions une dernière classe : *Englobants*.

Le diagramme général des classes, permettant de représenter à un niveau très abstrait l'ensemble du système de modélisation basé sur les caractéristiques que nous avons défini, est donné ci-dessous.

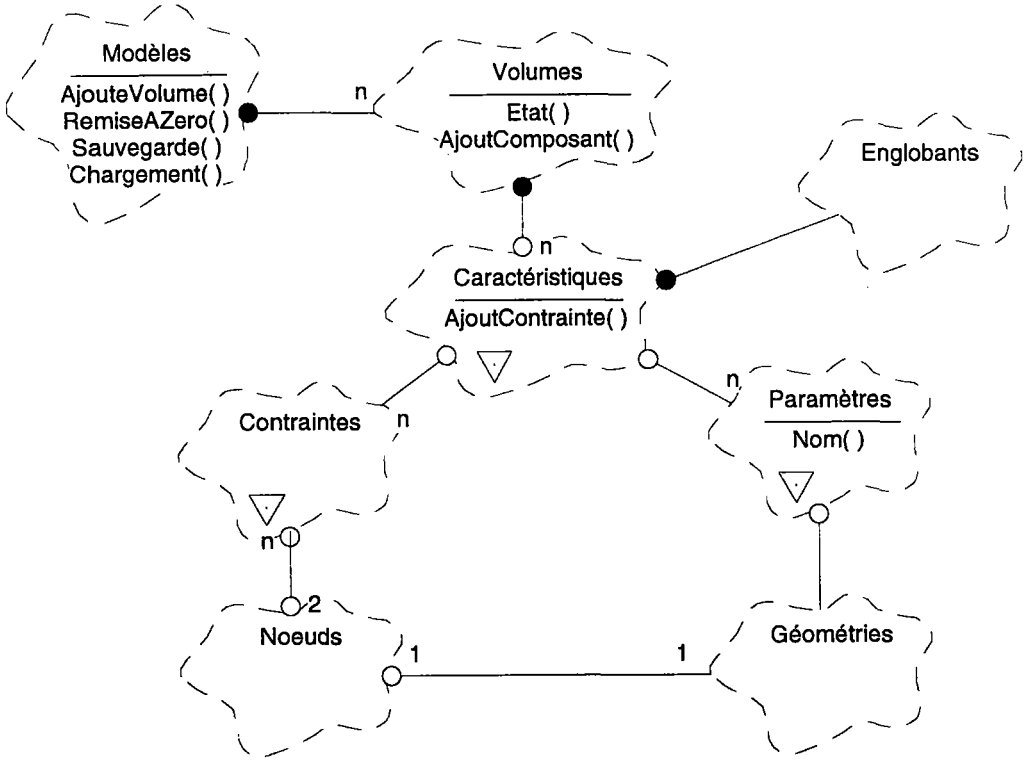


Figure 5.3 : Diagramme général des classes.

L'ensemble des fonctionnalités du système ne peut toutefois pas être décrit à ce niveau très abstrait. En particulier en ce qui concerne les caractéristiques, pour préciser davantage le comportement général du système, il faut passer à un niveau un peu plus détaillé.

Dans le chapitre 2, nous avons distingué les caractéristiques génériques des caractéristiques d'instanciation ou prêtes à l'emploi. Nous identifions donc deux classes : *CGénériques* et *CPAEs*. Ces deux classes héritent de la classe *Caractéristiques*, dont le rôle est de gérer un ensemble de contraintes. Cet ensemble de contraintes est cependant interprété localement de façon différente dans les *CGénériques* et dans les *CPAEs*. En effet, au niveau des *CGénériques*, les contraintes sont génériques, c'est-à-dire qu'elles sont communes à toutes les instances de caractéristiques de même nature. Par contre, dans les *CPAEs*, elles sont considérées comme des contraintes d'instanciation, ce qui permet notamment d'exprimer le dimensionnement et le positionnement de la caractéristique. Notons que l'ajout de contraintes est cependant similaire, qu'il s'agisse de contraintes génériques ou d'instanciation ; cette méthode permet d'insérer un élément dans une liste de contraintes, selon le cas, au niveau de la caractéristique générique ou au niveau de la caractéristique prête à l'emploi. Nous insistons sur le fait qu'il s'agit d'une interprétation contextuelle du contenu de la liste des contraintes, qui reste une structure de données commune à toutes les caractéristiques et donc gérée par la classe *Caractéristiques*.

La gestion des paramètres, déjà évoquée plus haut, est plus particulièrement à la charge des caractéristiques génériques. Elles utilisent donc les ressources de la classe *Paramètres*. En outre, elles ont la responsabilité de générer une *CPAEs* en créant notamment sa forme. Une méthode apparaît alors comme fondamentale pour le système de modélisation par les caractéristiques de forme : la méthode *CréationForme* qui permet de générer à la fois le B-Rep de la génératrice de la CPAE et son graphe conceptuel. Elle est surchargée par toutes les caractéristiques génériques, dérivées de *Cgénériques* et décrite au paragraphe 3.2.2.1.

Les caractéristiques prêtes à l'emploi sont créées à partir d'une caractéristique générique. Une CPAE a connaissance de son volume porteur ainsi que de son repère local exprimé sous la forme d'une matrice de changement de repère, qui permet de déterminer la position et l'orientation de l'instance. Elle a à sa charge de conserver ses volumes utiles sous forme d'un B-Rep. Elle peut donc également fournir son volume englobant.

Les détails donnés ci-dessus sont représentés sur la figure ci-dessous.

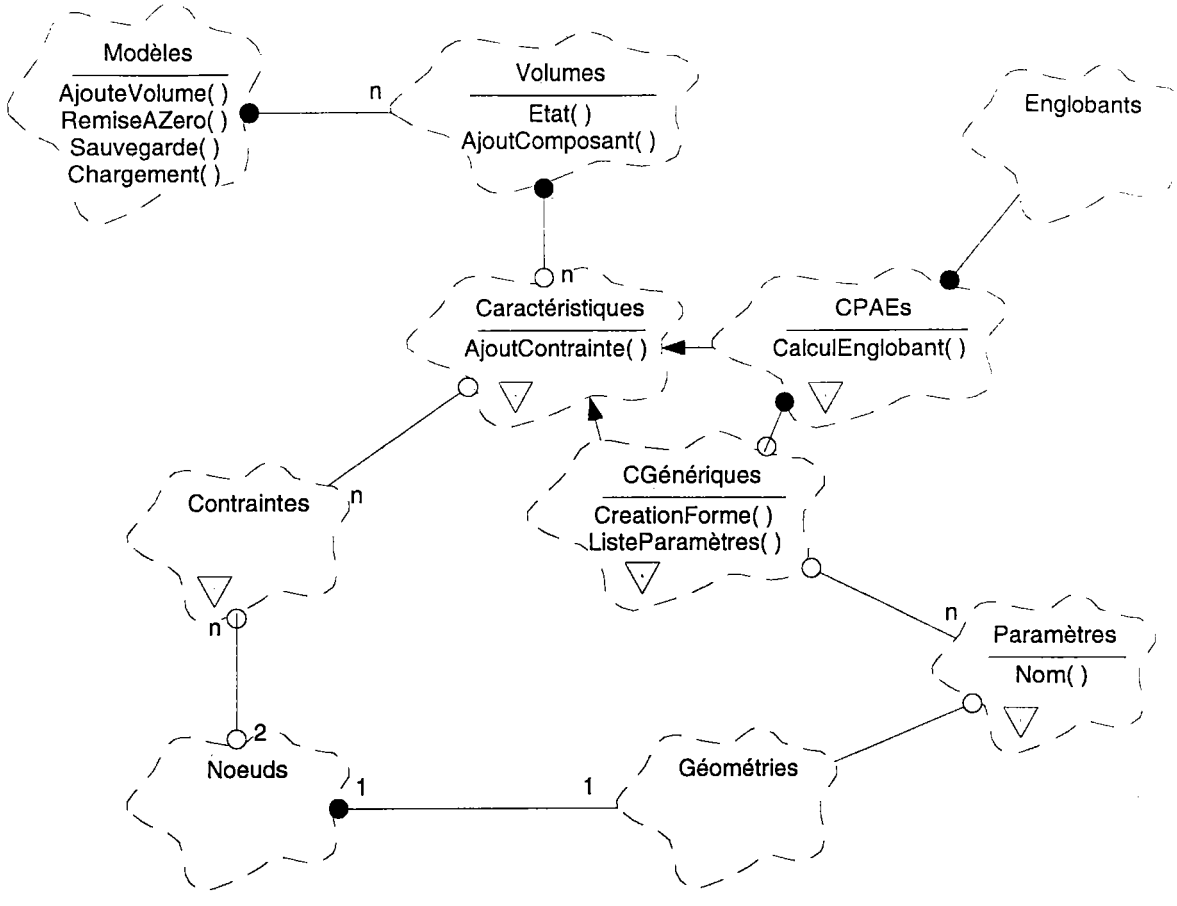


Figure 5.4 : Diagramme général des classes plus détaillé.

2.2. Diagrammes des catégories

À un niveau d'abstraction très élevé, nous pouvons identifier deux catégories : l'une représente notre modeleur à base de caractéristiques et l'autre représente un modeleur géométrique B-Rep. Notre modeleur à base de caractéristiques utilise donc les ressources d'un B-Rep.

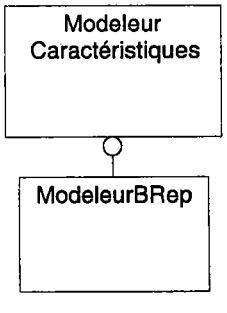


Figure 5.5 : Diagramme des catégories au niveau d'abstraction le plus haut.

Nous pouvons ensuite préciser le modèle sémantique de notre système à base de caractéristiques. Il est formé par un ensemble de volumes (appartenant à la catégorie Modèles),

eux-mêmes détaillés en composants de différentes natures. De manière générale, ces composants utilisent des paramètres et des contraintes, ce qui nous amène donc tout naturellement au diagramme des catégories de la figure 5.6.

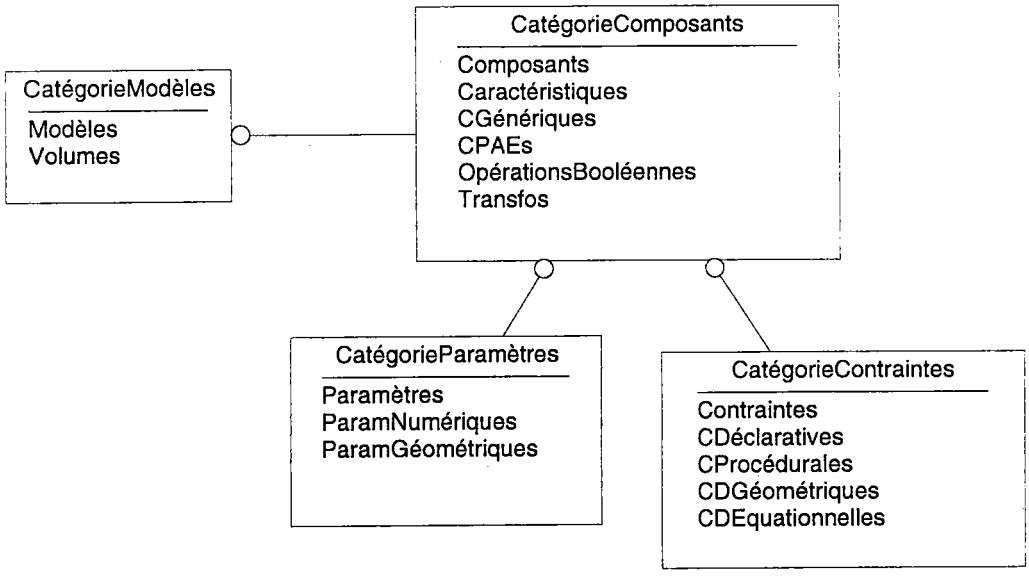


Figure 5.6 : Diagramme des catégories du modeleur à base de caractéristiques.

2.3. Diagrammes des classes par catégories

Nous venons de donner le diagramme général des classes représentant notre système, puis un diagramme des catégories qui le composent. La partie abstraite peut être précisée davantage. C'est pourquoi nous donnons dans la suite un diagramme des classes plus détaillé pour chacune des catégories *Composants, Caractéristiques, Paramètres* et *Contraintes.*

2.3.1. Catégorie Composants

Les composants qui constituent les volumes sont soit des caractéristiques, soit des transformations géométriques, soit des opérations booléennes. On identifie donc naturellement la classe *Composants* qui dérive en *Caractéristiques*, *OpBool* et *Transfos*. Un composant a la responsabilité de s'évaluer et de s'annuler au sein de son volume porteur, qu'il doit donc connaître. La classe *Caractéristiques* sera développée dans le paragraphe suivant, détaillant toute la catégorie *CatégorieCaractéristiques*. La classe *OpBool* dérive elle-même en trois autres classes : *Unions*, *Différences* et *Intersections*. Enfin, la classe *Transfos* dérive en : *Translations*, *Rotations* et *ChangementsEchelles*. Cette description de la catégorie *Composants* aboutit au diagramme des classes ci-dessous.

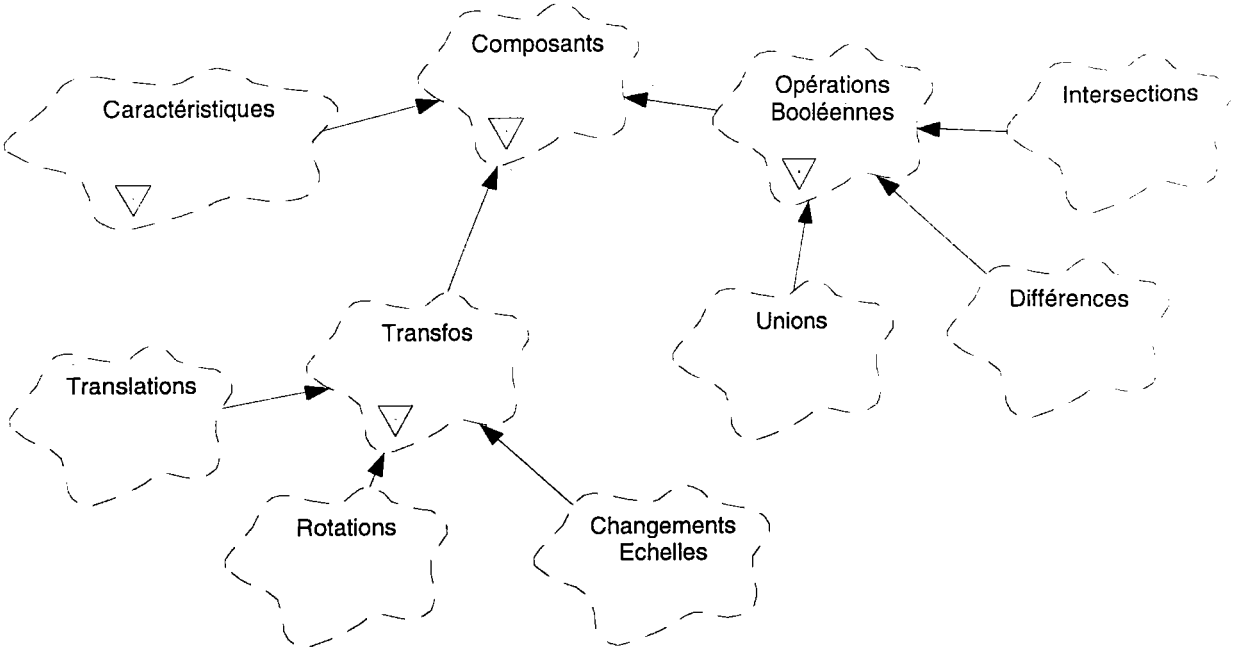


Figure 5.7 : Diagrammes des classes de la catégorie Composants.

2.3.2. *Catégorie Caractéristiques*

Nous avons déjà identifié, au paragraphe 2.1, la classe *Caractéristiques* ainsi que les deux classes dérivées *CGénériques* et *CPAEs*. Nous rappelons que la classe *Caractéristiques* a la responsabilité de gérer des contraintes.

Deux raisons nous ont conduits à ne maintenir qu'une seule liste de contraintes par caractéristique, regroupant à la fois les contraintes géométriques et les contraintes équationnelles. La première est de faciliter un parcours systématique de toutes les contraintes, sans différence de type, puisqu'elles doivent toutes être vérifiées. La seconde raison est la nécessité de conserver l'historique de création des contraintes, pour les contraintes d'instanciation. Ce dernier argument est fondamental si l'on décide de gérer les contraintes de manière paramétrique. Cependant, même si l'on opte pour une gestion variationnelle, l'historique reflète une partie de l'intention de conception, qu'il est nécessaire de conserver.

Notons que le graphe conceptuel de la caractéristique, que nous présentions au chapitre 2 comme un élément essentiel pour la gestion des contraintes ou l'extraction de caractéristiques, ne figure pas explicitement dans la liste des attributs des *CPAEs*. Par exemple, nous aurions pu conserver dans chaque caractéristique la liste des nœuds de son graphe conceptuel. Cela permettrait d'établir une liste des nœuds faiblement contraints par exemple. Mais la liste des nœuds est redondante avec la liste des contraintes car ils sont accessibles soit par l'intermédiaire des contraintes de la caractéristique (une contraintes reliant deux nœuds), soit par la liste de ses entités topologiques (faces...) grâce aux liens entre ces entités topologiques et leurs géométries et

entre les géométries et les nœuds (ces liens sont détaillés en 3.2.2.2). Nous ne conservons donc pas de liste de nœuds.

Dans la hiérarchie de la bibliothèque de caractéristiques génériques, nous les avons classées en caractéristiques primaires et secondaires. Il est donc naturel d'identifier des classes qui représentent les caractéristiques génériques primaires et secondaires (*CGPrimaires* et *CGSecondaires*) et que ces classes dérivent des *CGénériques*. Cette distinction s'explique par le fait qu'une caractéristique primaire peut exister de façon indépendante, ce qui n'est pas le cas d'une caractéristique secondaire. En effet, pour exister, les caractéristiques secondaires nécessitent un support (volume porteur) et doivent être évaluées au moment de l'instanciation. L'évaluation consiste à calculer le volume utile et à réaliser l'opération booléenne adéquate entre ce volume utile et le porteur de la caractéristique.

Au sein même des caractéristiques secondaires, nous avons distingué celles qui retirent de la matière de celles qui en ajoutent. Les classes qui en découlent sont donc *CGSNégatives* et *CGSPositives*, qui dérivent de *CGSecondaires*. Cette distinction vient de la nature différente des opérations booléennes à appliquer aux caractéristiques positives et négatives, que ce soit pour calculer le volume utile ou bien pour évaluer la caractéristique en question sur le volume porteur. En effet, pour le calcul du volume utile, on réalise une intersection entre la génératrice de la caractéristique et le volume porteur dans le cas d'une caractéristique négative, et une différence dans le cas d'une positive. Pour l'instanciation proprement dite, c'est-à-dire l'évaluation du volume comportant la caractéristique, si elle est négative, l'opération est une différence entre le volume porteur et le volume utile, si elle est positive c'est une union. Pour toutes les caractéristiques secondaires, positives ou négatives, il faut écrire des méthodes particulières : calcul du volume utile, évaluation et annulation. Ces méthodes sont déclarées au niveau des caractéristiques secondaires et surchargées dans les positives et dans les négatives. Le volume utile est obtenu en effectuant la première opération booléenne de l'instanciation. L'évaluation de la caractéristique sur son porteur est réalisée par la seconde opération booléenne. Enfin, l'annulation d'une caractéristique, pour la désinstancier lors d'une modification, correspond à l'opération booléenne inverse de la seconde opération de l'instanciation.

Au plus bas de la hiérarchie, là où le niveau de précision est le plus important, nous distinguons un certain nombre de caractéristiques constituant la bibliothèque. Comme caractéristiques primaires, on peut citer les *Blocs* et les *Cylindres*. Les caractéristiques secondaires sont par exemple : les *Rainures*, les *Marches*, les *Poches*, les *TrousBorgnes* ou

TrousDébouchants (négatives) ou encore les *Bossages*, les *RailsEnT* , les *PattesDeFixation* (positives)... Dans toutes ces classes, la méthode de création de la forme est surchargée.

L'ensemble de ces classes forme le diagramme des classes ci-dessous :

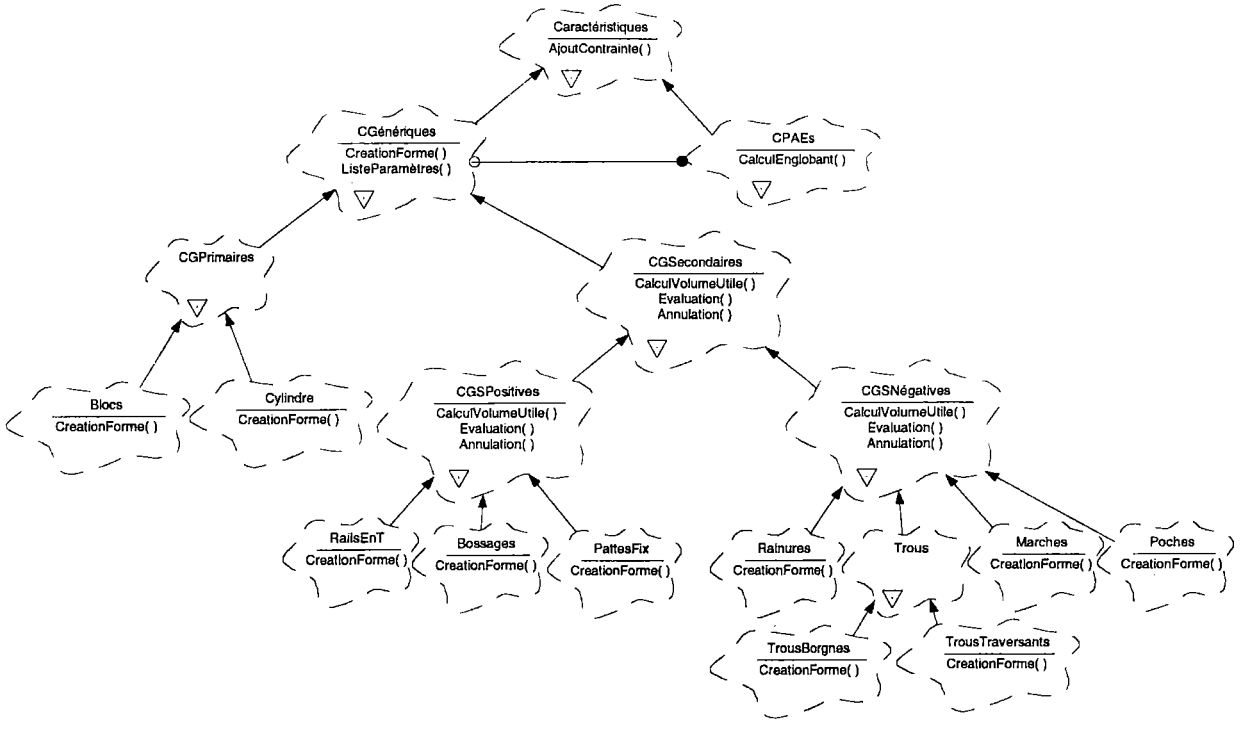


Figure 5.8 : Diagramme des classes pour la catégorie *Caractéristiques*.

2.3.3. Catégorie Paramètres

Comme nous l'avons vu précédemment, une caractéristique est spécifiée par un certain nombre de paramètres. Ces paramètres peuvent être de deux natures : les paramètres numériques et les paramètres géométriques. Les paramètres sont déterminés par un nom et une valeur. Le nom est nécessaire au dialogue pour demander à l'opérateur les valeurs des différents paramètres lors de l'instanciation des caractéristiques. Selon la nature du paramètre, le type de la valeur est différent : numérique ou géométrique. C'est pourquoi la fonction de consultation de la valeur est différente pour chaque type de paramètres et ne peut pas être virtuelle dans la classe de base. Nous identifions donc trois classes, *Paramètres*, *ParamNumériques* et *ParamGéométriques*, qui forment le diagramme des classes suivant :

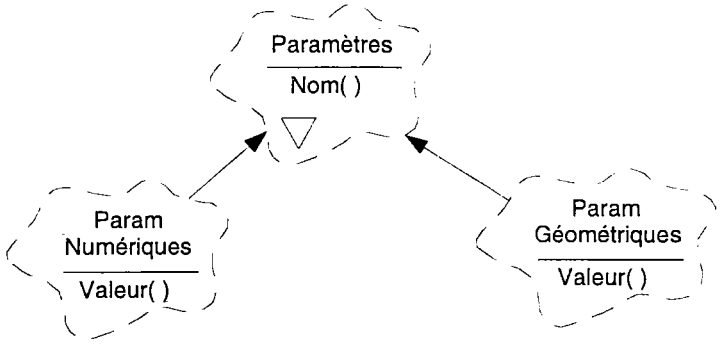


Figure 5.9 : Diagramme des classes de la catégorie *Paramètres*.

2.3.4. Catégorie Contraintes

La dernière catégorie identifiée est celle des contraintes. Au niveau du diagramme des classes, nous ne ferons pas de distinction entre les contraintes génériques et les contraintes d'instanciation car elles sont de même nature. La notion de contraintes représente une abstraction liées à l'interprétation locale (dans les caractéristiques génériques ou d'instanciation). Par contre, nous avons défini dans le chapitre 2, les contraintes déclaratives et les contraintes procédurales. La mise en œuvre n'a été réalisée que pour les contraintes déclaratives (les plus courantes), mais les contraintes procédurales sont tout de même prévues dans le schéma des classes. Nous rappelons également que les contraintes déclaratives peuvent s'exprimer sous deux formes : soit par une équation (contraintes équationnelles), soit par une relation entre deux entités géométriques (contraintes géométriques). Ces remarques amènent donc à identifier les classes *CProcédurales* et *CDéclaratives* qui héritent de la classe *Contraintes*, ainsi que les classes *CDÉquationnelles* et *CDGéométriques* qui héritent de *CDéclaratives*.

Nous rappelons que les contraintes génériques géométriques forment ensemble le graphe conceptuel de la caractéristique. Une contrainte géométrique est un arc entre deux nœuds de ce graphe. Toute contrainte géométrique permet donc d'accéder aux deux nœuds du graphe conceptuel de la caractéristique, représentant les deux géométries contraintes.

Les contraintes géométriques sont représentées par une hiérarchie, car elles peuvent être de différentes natures : coaxialité, concentricité, symétrie, tangence, angle. Parmi ces contraintes d'angle, on peut extraire deux cas particuliers, correspondant à un angle de 0° ou 90°. Ceci détermine deux classes dérivées : *Parallèles* et *Perpendiculaires*. On peut encore préciser davantage la contrainte de parallélisme en fixant une distance entre les deux entités parallèles. Ceci amène donc à identifier une classe supplémentaire, dérivée de *Parallèles*, qui est la classe *Distants*. Pour certaines de ces contraintes dérivées, *Angles* et *Distants*, il y a en plus des deux

nœuds, une valeur numérique exprimée par un paramètre. Pour la symétrie, il faut spécifier une référence, c'est-à-dire le centre ou l'axe de la symétrie. Cette référence est une géométrie.

Une contrainte géométrique s'exprimant entre deux nœuds, il faut également identifier cette classe. Un nœud possède une référence vers une géométrie (la géométrie qu'il permet de contraindre) et une liste de contraintes le liant à d'autres nœuds. Cette classe *Nœuds* ne fait pas réellement partie de la catégorie *Contraintes*, mais elle y est utilisée. Nous la représentons donc également sur le diagramme des classes de la catégorie *Contraintes* ci-dessous.

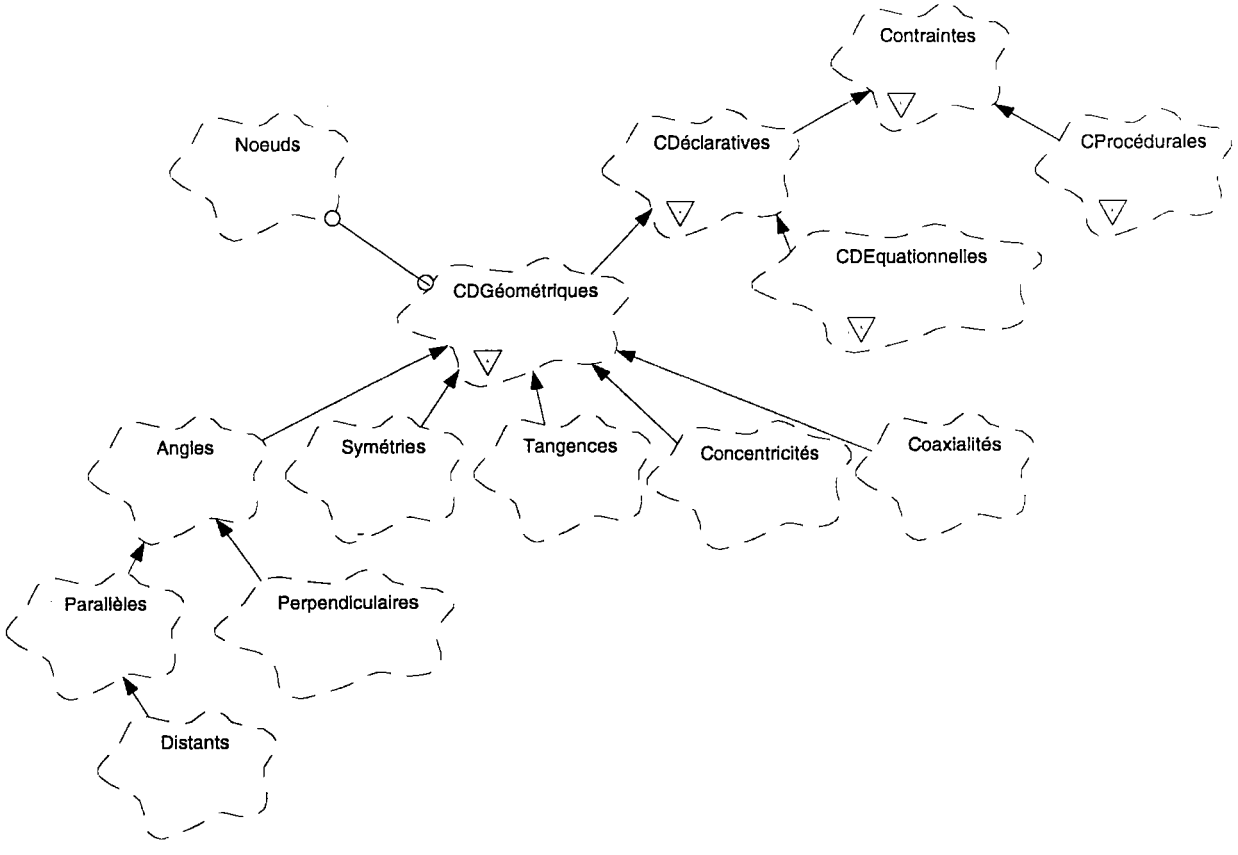


Figure 5.10 : Diagramme des classes de la catégorie *Contraintes*.

2.4. Liens entre les catégories

Si les catégories ont été présentées de façon distincte, elles ne sont toutefois pas indépendantes les unes des autres. En effet, il existe des relations entre des classes de plusieurs catégories, qui sont illustrées par la figure 5.11.

Les caractéristiques gèrent un ensemble de contraintes ; la classe *Caractéristiques* utilise donc la classe *Contraintes*. Les caractéristiques génériques sont représentées en outre par une liste de paramètres ; la classe *CGénériques* est donc composée de la classe *Paramètres*. De plus, la classe

CPAEs est composée des classes *Repères3D* et *Englobants* pour mémoriser respectivement la position et l'englobant de la caractéristique.

Certaines contraintes sont précisées par des valeurs numériques, données sous forme de paramètres. La classe *ParamNumériques* compose donc les classes *Angles* et *Distants*.

La classe *Géométries* est utilisée par trois autres classes : *ParamGéométriques*, *Nœuds* et *Symétries*. En effet, la valeur d'un paramètre géométrique est une géométrie. De même, les nœuds du graphe de contraintes représentent des géométries. Enfin, la contrainte de symétrie a besoin d'une entité de référence qui peut être un point, un axe ou un plan (centre de la symétrie) donc une géométrie.

Le dernier lien mis en évidence est celui entre les classes *Nœuds* et *CDGéométriques* (contraintes) : il s'agit d'un double lien de composition. En effet, les nœuds contiennent une liste d'arcs qui sont des contraintes géométriques et une contrainte géométrique représente un lien entre deux nœuds (ses deux attributs sont donc des nœuds).

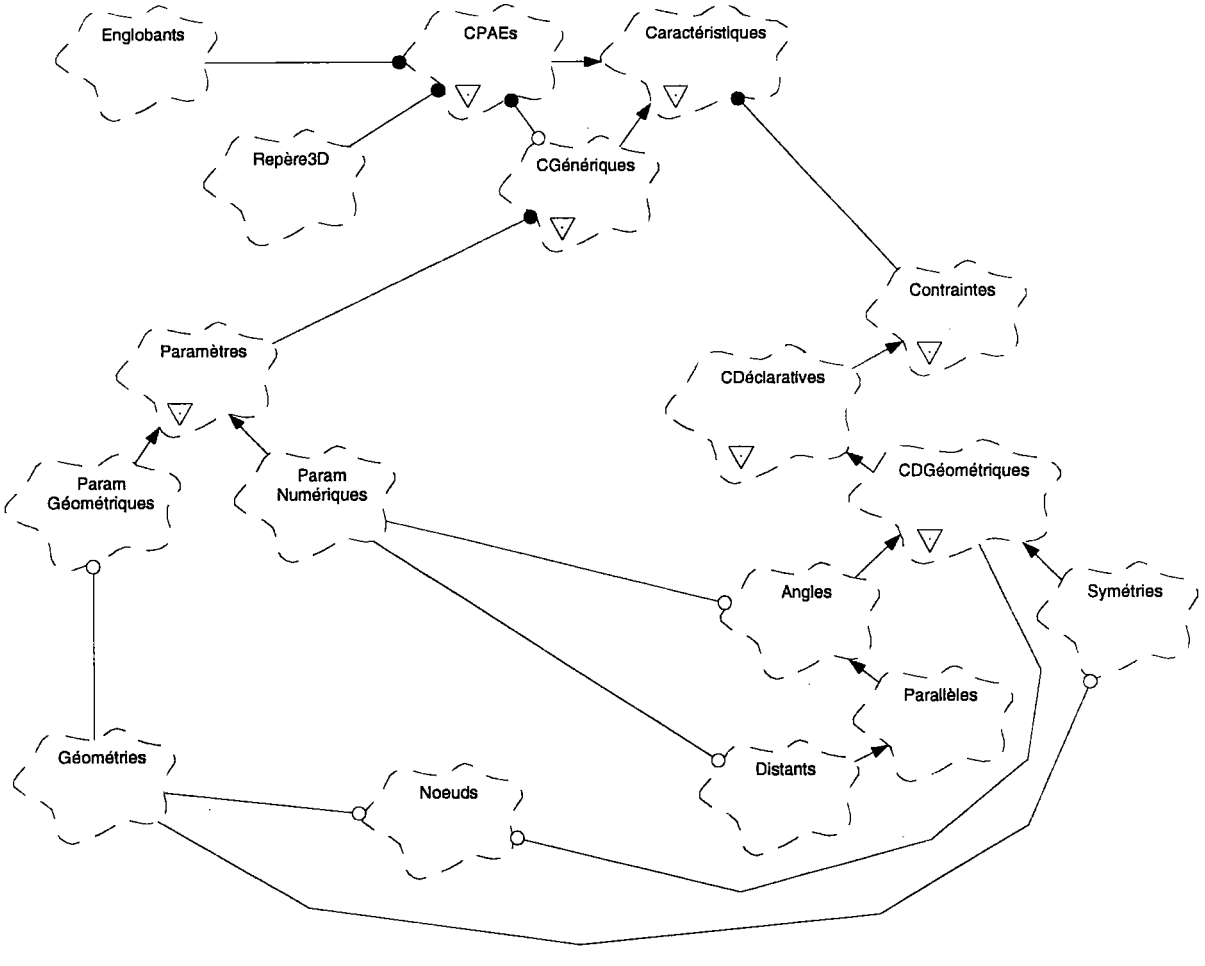


Figure 5.11 : Illustration des liens entre classes de différentes catégories.

3. QUELQUES PRÉCISIONS

Après avoir identifié toutes les classes nécessaires à la réalisation de notre système de CAO, il nous est apparu nécessaire d'introduire la notion d'identifiant, pour représenter tous les objets du système. Cette notion est expliquée dans le paragraphe 3.1.

Nous avons montré, au cours de notre étude, que la fonctionnalité de notre système est de gérer un modèle de conception à l'aide des caractéristiques de forme. Pour cela, un certain nombre d'actions spécifiques doivent être prévues.

Le système tel qu'il est conçu prévoit de gérer des contraintes, qui sont appliquées au caractéristiques. C'est pourquoi, il faut pouvoir attacher des contraintes aux caractéristiques ; nous développons donc l'ajout de contraintes.

Nous rappelons que la génératrice (B-Rep et graphe conceptuel) d'une caractéristique est obtenue par exécution de la procédure de création de la forme. Aussi détaillons nous cette procédure en insistant sur le contexte dans lequel elle peut être utilisée (instanciation ou réinstanciation).

Enfin, la démarche mise en œuvre pour l'instanciation d'une caractéristique mérite quelques commentaires, notamment en ce qui concerne les responsabilités attribuées aux caractéristiques et celles affectées au dialogue homme-machine.

3.1. Notion d'identifiants

Tous les objets sont munis d'un identifiant les qualifiant de façon unique. Un identifiant est composé de deux champs : la clé, qui peut être un entier par exemple, et l'adresse mémoire de l'objet identifié. Cet identifiant est très utile, notamment lors de la sauvegarde du modèle, qui est ainsi uniformisée. En effet, tous les objets du modèle sont sauvés tour à tour et sont donc persistants, grâce à leurs clés qui sont conservées (contrairement à leurs adresses). De plus, quand une référence vers un objet est nécessaire, on peut conserver l'identifiant de cet objet uniquement. Ceci permet de masquer la notion de pointeurs qui est toujours délicate à gérer d'un point de vue programmation. En outre, les performances sont améliorées en terme de stockage et en temps d'exécution. En effet, l'objet identifié n'existe qu'une seule fois en mémoire. De plus, les opérations de copie et d'affectation sont réduites au minimum puisque la représentation de l'objet est unique (seule son adresse est copiée). Les avantages offerts par les identifiants sont

donc nombreux. Un inconvénient apparaît toutefois lors du développement : les méthodes de l'objet référencé ne sont pas immédiatement disponibles au niveau de l'identifiant. En effet, le pointeur masqué par l'identifiant est non typé, de manière à pouvoir pointer sur tout type d 'entité. Il convient donc de typer ce pointeur pour accéder aux services de l'objet pointé. Ce n'est toutefois pas un inconvénient à l'exécution, car les performances ne sont pas altérées.

Les identifiants permettent en outre la généricité . Ainsi, on peut gérer des listes d'identifiants, qui permettent de représenter de la même manière, c'est-à-dire avec la même structure, des listes de paramètres, des listes de contraintes, des listes de caractéristiques, des listes de volumes... L'inconvénient des listes d'identifiants est qu'elles peuvent être hétérogènes ; il ne peut donc pas y avoir de vérification de type entre les divers éléments de la liste. Ainsi, on peut trouver a priori dans une même liste deux objets de types complètement différents, ce qui peut être regrettable. C'est donc au programmeur de garantir une cohérence de type entre éléments d'une même liste d'identifiants. Toutefois, de telles listes représentent l'avantage d'être peu volumineuses, puisqu'elles ne contiennent que des identifiants (clé et adresse) au lieu des objets entiers. Cette solution est donc plus efficace que les patrons, qui offrent a priori les mêmes possibilités.

Les classes qui possèdent un champ de type *Identifiants* sont : *Modèles, Volumes, Composants, Paramètres, Contraintes, Nœuds, Géométries, Englobants*. Nous retrouvons les classes identifiées dans le diagramme général des classes du système.

3.2. Détails de quelques actions réalisées par le système

Nous avons identifié, dans le paragraphe 2, toutes les classes, leurs attributs et leurs méthodes. Certains algorithmes méritent cependant d'être détaillés. C'est pourquoi, nous consacrons le présent paragraphe à l'ajout de contraintes, à la création de la forme et à l'instanciation.

3.2.1. Ajout de contraintes

La méthode *AjoutContrainte* permet, comme son nom l'indique, d'ajouter une contrainte à une caractéristique générique ou à une caractéristique d'instanciation. Dans le premier cas, elle est utilisée pour créer le graphe conceptuel de la caractéristique générique, lors de la création de la forme. Dans le second cas, elle est utilisée quand l'opérateur impose des contraintes lors de l'instanciation de la caractéristique (placement) ou à un moment ultérieur. Dans les deux cas, l'ajout d'une contrainte revient à l'insérer dans la liste des contraintes.

Cependant, l'ajout d'une contrainte générique se fait par la procédure de création de la forme, qui crée successivement, les faces de la caractéristique, les géométries correspondantes, puis les nœuds. Les contraintes sont alors créées à partir de ces nœuds.

Par contre, pour créer une nouvelle contrainte géométrique lors de l'instanciation d'une caractéristique, il existe plusieurs solutions pour l'insérer au graphe conceptuel. La première consiste à identifier les entités topologiques du B-Rep (faces...), sur lesquelles la contrainte porte. À partir de ces entités, on atteint les nœuds, grâce aux liens existants entre le B-Rep et le graphe (ces liens sont détaillés dans le paragraphe 3.2.2.2). Une fois les nœuds connus, la contrainte peut être créée et ajoutée à la liste. La seconde solution est l'édition du graphe conceptuel et l'ajout direct par l'opérateur de la contrainte (c'est-à-dire un arc) entre deux nœuds.

Dans tous les cas, une fois que la contrainte est insérée dans la structure, il faut s'assurer qu'elle est vérifiée. Si tel n'est pas le cas, il faut réaliser les modifications permettant de la satisfaire. Ce travail est à la charge du gestionnaire de contraintes, nous ne le détaillons donc pas.

3.2.2. Création de la forme

L'autre méthode sur laquelle nous nous attardons est la procédure de création de la forme. Cette méthode est surchargée dans toutes les caractéristiques génériques, puisque chacune a une forme particulière.

Créer la forme d'une caractéristique consiste à créer sa représentation dans le B-Rep en termes d'entités de bas niveau (sommets, arêtes, faces) et une représentation de plus haut niveau, c'est-à-dire son graphe conceptuel. On pourrait écrire deux procédures distinctes : l'une pour la création du B-Rep et l'autre pour la création du graphe conceptuel. Cette séparation se justifierait par le fait que lors d'une modification d'un paramètre d'une caractéristique, le B-Rep change mais pas le graphe de contraintes. Il suffirait donc d'exécuter la première des deux procédures. Cependant, nous avons choisi de tout regrouper dans une même procédure, pour deux raisons. D'une part, il est plus simple de créer les nœuds du graphe en même temps que les faces correspondantes, puis immédiatement les contraintes (arcs du graphe) ; dans le cas de deux procédures distinctes, il faudrait parcourir a posteriori les faces de la caractéristique pour créer les nœuds et les contraintes. D'autre part, lors de la description de la forme d'une nouvelle caractéristique générique, la suite d'actions mémorisée pendant l'espionnage correspond à une création complète (B-Rep et graphe). En effet, que le graphe soit créé par détection de contraintes ou à partir d'un graphe existant, il l'est au même moment que le B-Rep.

3.2.2.1. Procédure de création de la forme

La procédure de création de la forme peut avoir deux aspects légèrement différents, selon la provenance de la caractéristique dont elle détermine la forme. En effet, soit la caractéristique est prédéfinie auquel cas la procédure est écrite par un programmeur, soit elle a été décrite interactivement et la procédure est alors obtenue par espionnage. Cependant, dans les deux cas, la procédure a le même rôle.

La procédure de création de la forme peut être écrite par le programmeur du logiciel (celle d'une caractéristique prédéfinie) ou par un opérateur ayant une bonne connaissance de la structure du B-Rep. Elle consiste alors à créer les entités de base du B-Rep (sommets, arêtes, faces), puis les nœuds (géométries) et les arcs (contraintes) du graphe. Les coordonnées de tous les sommets de la caractéristique sont exprimées dans son repère local en fonction des paramètres de dimensions. Leurs coordonnées réelles, dans le repère du monde, sont calculées automatiquement en appliquant un changement de repère, correspondant au positionnement de la caractéristique. Les sommets sont créés d'abord, puis reliés par des arêtes, qui elles-mêmes sont chaînées pour former des faces, qui assemblées constituent le volume 3D de la caractéristique. Lors de la création d'une face dans le B-Rep, la géométrie (c'est-à-dire la surface porteuse de la face) est extraite pour créer le nœud correspondant dans le graphe conceptuel. Lorsque les faces et les nœuds sont créés, toutes les contraintes génériques géométriques de la caractéristique sont ajoutées dans le graphe conceptuel en reliant par des arcs un certain nombre de nœuds. L'aspect d'une telle procédure de création de la forme est donné ci-dessous et le listing de la procédure de création d'une boîte parallélépipédique est fourni dans l'annexe E.

```
CréationForme1   (Param1, Param2, ..., Pos, PremInst)
/*  Entrée :    Param1, Param2... = liste des paramètres de la forme                    */
/*              Pos               = repère local de la caractéristique                  */
/*              PremInst          = booléen indiquant s'il s'agit d'une première instanciation */
/*  Sortie :    numéro de l'objet créé                                                  */
Début
   Consultation des valeurs des paramètres numériques.
   /*  à partir des objets de type "Paramètres", on obtient des valeurs numériques     */

   Création des sommets en fonction des valeurs des paramètres de dimensions
   et de position.
   /*  en appliquant le changement de repère donné par le positionnement de la CPAE   */
   /*  aux points exprimés dans le repère local de la caractéristique générique       */

   Création des arêtes.
   Création des faces.
   Création du volume.

   Si PremInst Alors
   /*  s'il s'agit de la première instanciation de la caractéristique                  */
      Création des géométries et des nœuds associés aux faces.
      Ajout des contraintes génériques.
   Sinon
      Mise à jour des géométries avec les équations des plans des nouvelles faces.
   FinSi
Fin
```

Nous avons annoncé dans le chapitre 4 que la procédure de création de la forme pouvait être obtenue par espionnage des actions interactives de l'opérateur, lors de la définition interactive de nouvelles caractéristiques génériques. Dans ce cas, il y a deux possibilités : d'une part la description de la forme par l'extrusion d'un profil, d'autre part la suppression de faces dans un volume ou une combinaison de volumes prédéfinis. Pour illustrer cette seconde possibilité, une marche peut être obtenue en créant une boîte et en supprimant quatre faces. Lors de la destruction d'une face dans le B-Rep, il faut également supprimer sa géométrie si elle est la seule à s'appuyer dessus, ainsi que le nœud correspondant dans le graphe conceptuel, avec tous les arcs qui s'y rapportent et qui correspondent aux contraintes qui portaient sur la face. Un exemple de création de la forme par suppression de faces dans un volume est donné ci-dessous et le listing de la création d'une marche est disponible dans l'annexe E.

```
CréationForme2 (Param1, Param2, ..., Pos, PremInst)
/*  Entrée :    Param1, Param2... = liste des paramètres de la forme                    */
/*              Pos               = repère local de la caractéristique                  */
/*              PremInst          = booléen indiquant s'il s'agit d'une première instanciation */
/*  Sortie :    numéro de l'objet créé                                                  */
Début
   /*  Création d'un volume avec la première version de la procédure                   */
   CréationForme1 (Param1, Param2, ..., Pos, PremInst)

   /*  Suppression de la face, éventuellement de sa géométrie, des contraintes associées et du nœud */
   SuppressionFace (i)        // i est le rang de la face à supprimer
   SuppressionFace (j)        // j est le rang d'une autre face à supprimer
   ...
   Retour du volume créé.
Fin
```

3.2.2.2. Première instanciation

Lors de l'instanciation, la procédure de création est exécutée et les géométries sont créées en même temps que les faces de la génératrice. Des liens entre ces géométries et les faces correspondantes sont établis, puis les contraintes génériques sont créées dans le graphe, entre les nœuds correspondant aux géométries (voir figure 5.1, page 145).

La première opération booléenne de l'instanciation permet de calculer le volume utile. Lorsqu'il est obtenu, il faut établir les liens entre ses faces nouvellement créées et les géométries des faces de la génératrice détruites lors de l'opération booléenne. Pour cela, les opérations booléennes ont été adaptées pour donner en plus du volume résultat, pour chacune de ses faces, la liste des faces des opérandes desquelles elle provient. En effet, toute face générée lors d'une opération booléenne a la même géométrie qu'au moins une face des opérandes. Pour retrouver une géométrie à partir d'une face, un tableau de références est maintenu entre les numéros des faces et les géométries associées. Le numéro de face représente l'indice d'entrée dans le tableau. L'élément d'indice *i* est la liste des identifiants des géométries correspondant à la face numéro *i* dans le B-Rep. Quand, après une opération booléenne, une nouvelle face *nf* provient d'une ancienne face *af*, il suffit d'ajouter la nouvelle face dans la géométrie de l'ancienne et une référence dans le tableau des références entre cette nouvelle face et sa géométrie. On supprime ensuite, dans la liste des faces s'appuyant sur la géométrie, la face de l'opérande qui a été détruite dans l'opération booléenne. La figure ci-dessous schématise la mise à jour de ces liens.

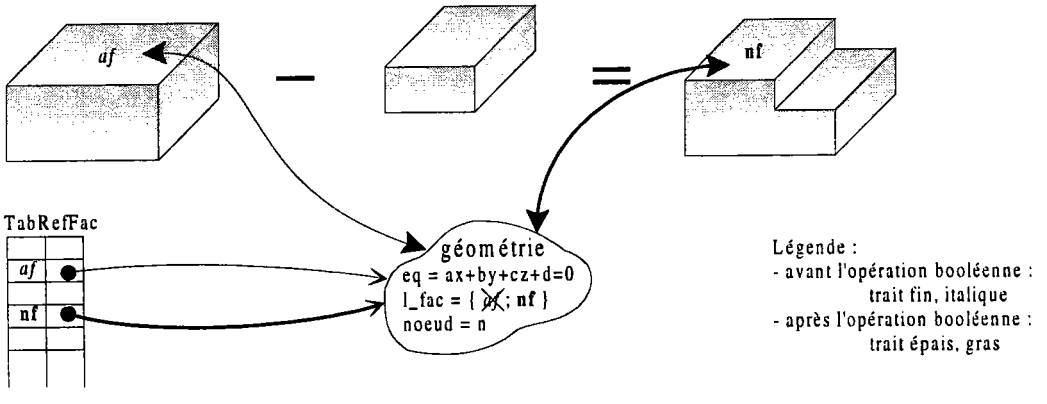


Figure 5.12 : Mise à jour des liens entre une nouvelle face créée lors d'une opération booléenne et la géométrie de la face dont elle provient.

Une nouvelle face *nf* peut également provenir de plusieurs faces distinctes *afs*, lors par exemple de l'annulation d'une rainure. Les géométries des faces devraient être identiques. Si elles ne le sont pas, pour ne perdre aucune des contraintes portant sur ces différentes géométries, au lieu de rajouter une référence entre *nf* et la géométrie, on conserve la liste des géométries des faces dont *nf* provient.

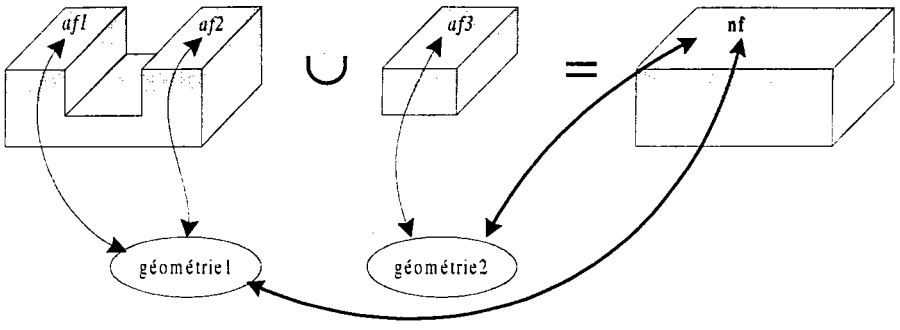


Figure 5.13 : Mise à jour des liens lorsqu'une nouvelle face provient de plusieurs faces de géométries différentes, lors de l'annulation d'une rainure.

La seconde opération booléenne réalisée lors de l'instanciation, entre le volume porteur et le volume utile de la caractéristique, permet d'instancier véritablement la caractéristique sur son porteur. Là encore, de nouvelles faces sont créées par l'opération booléenne, provenant des faces des opérandes (volume porteur et volume utile de la caractéristique). Les liens avec les géométries sont établis de la même façon que lors de la première opération booléenne.

3.2.2.3. Réinstanciation lors de modifications

La procédure de création de la forme est également exécutée lors de modifications du modèle de conception. Comme nous l'avons détaillé dans le chapitre 3, une modification est réalisée en plusieurs étapes : elle consiste d'abord à désinstancier tout ce qui est affecté par la modification, puis à tout réinstancier.

Nous rappelons que la désinstanciation consiste à réaliser l'opération booléenne inverse de l'instanciation. Qui dit opération booléenne, dit problèmes de mise à jour des liens entre les faces et les géométries. D'autant plus que la désinstanciation consiste justement à supprimer un certain nombre de faces (celles correspondant à la caractéristique à désinstancier). Or, nous rappelons que les contraintes géométriques relient deux nœuds du graphe conceptuel, qui correspondent aux géométries des faces contraintes. La désinstanciation ne doit donc pas détruire le graphe conceptuel, pour conserver les contraintes qui devront à nouveau être vérifiées lors de la réinstanciation. Les faces sont donc supprimées par l'opération booléenne, mais les géométries correspondantes sont conservées, donc les contraintes géométriques également.

Lors de la réinstanciation, la procédure de création est de nouveau exécutée, mais avec une option particulière (booléen) qui indique qu'il ne s'agit pas d'une première instanciation. C'est le rôle du paramètre *PremInst* dans les deux exemples de la procédure de création de la forme fournis quelques pages plus haut. En effet, qu'il s'agisse d'une première instanciation ou d'une réinstanciation, la création du B-Rep est similaire, dans la mesure où les entités créées (sommets,

arêtes, faces, volume) sont les mêmes ; seuls les paramètres changent. Par contre, dans le graphe conceptuel, ni les nœuds correspondant aux nouvelles faces, ni les arcs représentant les contraintes géométriques ne sont recréés ; seules les géométries sont mises à jour par les équations des nouvelles faces.

Il faut donc conserver un lien permettant de retrouver à quelle géométrie la i^{ème} face de l'objet correspond. Pour réaliser cela, on maintient un tableau, au niveau de chaque caractéristique, qui permet de faire la correspondance entre l'ordre des faces de la caractéristique (de 1 à n si la caractéristique possède n faces) et les numéros des faces dans le B-Rep. Ce tableau est créé lors de la première instanciation de la caractéristique pour être exploité lors de réinstanciations. En effet, à ce moment, on retrouve facilement le numéro de la i^{ème} face de la caractéristique, par lequel on atteint sa géométrie. Lorsque la nouvelle i^{ème} face est créée, la géométrie correspondante est remise à jour et le numéro de la i^{ème} face également. La figure 5.14 représente ce tableau de correspondance pour une poche.

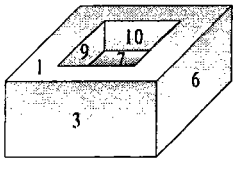


Tableau des faces de la poche

1	7
2	8
3	9
4	10
5	11

Figure 5.14 : Tableau de correspondance entre le rang des faces d'une caractéristique et leurs numéro dans le B-Rep.

3.2.3. Démarche d'instanciation

Nous avons vu au chapitre 2 que pour réaliser l'instanciation d'une caractéristique, c'est-à-dire sa création et son insertion dans le modèle de conception, il fallait renseigner certaines données. En effet, il faut d'abord dimensionner et placer la caractéristique. Pour réaliser ces opérations de façon interactive, nous avons d'abord envisagé la procédure d'instanciation de la façon suivante :

```
void CPAEs :: Traitement_Instanciation ()
Début
   caract.CréationForme (Param..., Pos, Vrai)
      // l'opérateur a fourni les valeurs de certains paramètres, d'autres ont des valeurs par défaut

   TantQue non ( Placée () et Dimensionnée () )   Faire
      cont ← Acquisition_Contrainte ()
         // les contraintes sont acquises par le module de dialogue...

      AjoutContrainte (cont)
         // ... puis insérées dans la caractéristique et les paramètres sont recalculés

      caract.CréationForme (Param..., Pos, Vrai)
         // la forme de la génératrice est réévaluée avec les nouvelles valeurs des paramètres

      Vue.Affichage ()
   FinTantQue
Fin
```

Pour l'instanciation des caractéristiques secondaires, il faut, en plus du traitement décrit ci-dessus, renseigner le volume porteur de la caractéristique et après avoir calculé la génératrice, réaliser les deux opérations booléennes de l'instanciation (la première pour le calcul du volume utile et la seconde pour l'instanciation proprement dite). L'instanciation des caractéristiques secondaires est donc réalisée par l'algorithme ci-dessous :

```
void CPAESecondaires :: Instanciation ()
Début
   Acquisition_Porteur ()
   Traitement_Instanciation ()
   Calcul_Volume_Utile ()          // première opération booléenne de l'instanciation
   Evaluation ()                   // seconde opération booléenne de l'instanciation
Fin
```

Nous avons toutefois conclu qu'il n'était pas de la responsabilité de la caractéristique de faire l'acquisition des paramètres de dimensions, de position et d'orientation. Le dialogue dispose des méthodes qui lui permettent d'ajouter une contrainte, de saisir les paramètres de dimensions et de savoir si une caractéristique est bien déterminée, c'est-à-dire si elle est correctement dimensionnée, placée et orientée. C'est donc du ressort du dialogue de réaliser l'acquisition des paramètres et des contraintes. L'instanciation dans le modèle est donc réalisée avec les bonnes valeurs des paramètres.

4. COMMENTAIRES SUR L'APPROCHE ORIENTÉE OBJETS

Les méthodes orientées objets permettent la manipulation de classes, qui sont des structures de haut niveau regroupant des données et un comportement. Ainsi, on peut attribuer à chaque classe manipulée les responsabilités qui lui sont propres. Cependant, cette répartition des

responsabilités est délicate et nécessite sans aucun doute une grande expérience de l'approche orientée objets. En particulier, les méthodes relativement générales ne correspondent pas toujours de manière évidente à des classes spécifiques. Il est donc relativement fréquent de les placer dans des classes inappropriées, ce qui nuit à une bonne représentation de l'architecture du système.

L'intérêt principal de l'approche objets est l'héritage. Il permet en effet de représenter un niveau d'abstraction très élevé et d'avoir ainsi une vue globale du système à mettre en œuvre. On peut obtenir un schéma général des classes comme celui du paragraphe 2.1, dans lequel seules les classes abstraites sont représentées. Le comportement général du système peut donc être exprimé en ne considérant que les responsabilités de ces classes abstraites, de haut niveau. Les détails plus spécifiques sont précisés dans les classes dérivées. Par exemple, toutes les caractéristiques ont la responsabilité de créer leur forme, mais comme elle est spécifique à chacune, la méthode CréationForme est surchargée.

L'héritage permet en outre de factoriser les points communs à plusieurs classes, c'est-à-dire de ne définir certaines méthodes que pour les classes de haut niveau. Par exemple, ceci est particulièrement utile pour décrire les méthodes communes à plusieurs caractéristiques dérivées d'un même parent. En effet, le calcul du volume utile se fait de manière identique pour toutes les caractéristiques secondaires positives, et d'une autre manière pour toutes les caractéristiques négatives. Cette méthode n'est donc pas surchargée pour toutes les caractéristiques. Il en est de même pour l'évaluation d'une caractéristique sur son volume porteur qui est réalisée par une opération booléenne qui dépend du type (positif ou négatif) de la caractéristique.

Enfin, l'héritage induit une structure hiérarchique naturelle, ce qui semble bien adapté, dans notre cas, à la structure hiérarchique de la bibliothèque de caractéristiques. Les caractéristiques spécialisées (rainure, marche...) représentent les feuilles de cette hiérarchie. L'héritage est particulièrement propice à l'extension de la bibliothèque. En effet, lorsqu'il définit une nouvelle caractéristique générique, l'opérateur connaît le comportement de cette caractéristique ; il connaît en particulier sa nature (primaire ou secondaire) et son type (positif ou négatif). Il n'a donc qu'à préciser le comportement particulier, comme le création de la forme. Ceci se traduit par la surcharge des méthodes virtuelles.

Le langage de programmation que nous avons utilisé pour la mise en œuvre est le langage C++. Ce langage couvre une large variété d'applications car il est interfacé avec divers logiciels (interfaces graphiques, réseau...). En particulier, il convient bien aux applications scientifiques (calculs, graphisme...). En outre, le C++ est un langage portable et homogène. La manipulation des objets est aisée en ce sens que la déclaration de l'héritage se fait simplement. Cependant, la

mise en œuvre du polymorphisme est délicate. Si la conversion de type est naturelle dans le sens ascendant (d'une classe dérivée vers une classe de base), elle n'est ni automatique ni simple dans le sens descendant. En effet, si l'on veut manipuler un objet très précis à partir d'une instance plus générale, le typage dynamique n'est pas automatique et il faut le forcer (par un opérateur de transtypage ou *cast*). En outre, il oblige à la manipulation de pointeurs ou de références, c'est-à-dire des notions de programmation de très bas niveau. Ceci implique une gestion dynamique de la mémoire (allocation, désallocation), qui est souvent la cause d'erreurs difficiles à détecter. Ces notions de bas niveau contrastent avec les concepts de haut niveau tels que les classes et l'héritage. Le C++ est donc à la fois simple, car il s'appuie sur un langage bien maîtrisé, le langage C, et à la fois délicat, car il véhicule tous les aspects de bas niveau du C.

5. CONCLUSION

Ce chapitre a présenté la mise en œuvre de notre modeleur à base de caractéristiques, décrit dans les chapitres précédents. Nous avons exposé, dans une première partie, les résultats de l'analyse et de la conception qui ont été réalisées, dans le formalisme OOAD. Ces résultats ont permis d'effectuer la programmation du modeleur en langage C++ [Cop 92], sur stations SUN, sous le système unix (SunOS 5.5. Solaris). Quelques copies d'écran de pièces modélisées avec ce modeleur sont disponibles dans l'annexe F. Dans la seconde partie du chapitre, nous avons justifié certains choix faits pour la mise en œuvre. L'outil autorisant la définition interactive de nouvelles caractéristiques de forme génériques a été analysé (chapitre 4) mais n'est pas encore implanté au moment de la rédaction de ce manuscrit.

Nous signalons que la mise en œuvre s'est appuyée sur des modules existants. Il s'agit du modeleur B-Rep, des opérations booléennes et d'un logiciel de visualisation développés par d'autres membres du laboratoire. Enfin, pour rendre notre logiciel convivial et pleinement opérationnel, une coopération plus approfondie avec les équipes de recherche travaillant sur le dialogue homme-machine et la gestion des contraintes demeure nécessaire.

CONCLUSION

L'objectif de ce manuscrit a été de présenter notre modeleur à base de caractéristiques au niveau conceptuel comme au niveau réalisation. Nous pensons avoir fait un certain nombre de propositions originales. De nombreux systèmes existent, mais ne prennent souvent pas tous les problèmes en compte et présentent donc quelques déficiences. L'étude bibliographique a montré les solutions adoptées de façon quasiment unanime, les originalités de quelques systèmes et les lacunes subsistant encore. Cependant, nous sommes conscients que les propositions faites ne permettent pas de résoudre tous les problèmes et qu'il reste de nombreuses perspectives à développer. Cette conclusion a pour objet de réaliser un bilan du travail effectué et des résultats obtenus, ainsi que de présenter la manière dont nous envisageons la suite de notre travail de recherche.

Le système de Conception Assistée par Ordinateur que nous avons développé est constitué d'une bibliothèque de caractéristiques génériques et d'un modèle de conception. La structure de la bibliothèque est hiérarchique, dans la mesure où les caractéristiques sont classées en primaires et secondaires, et les caractéristiques secondaires elles-mêmes sont soit positives, soit négatives. Le modèle de conception est structuré en deux couches. Une couche sémantique permet de mémoriser l'historique de construction de la pièce modélisée, en maintenant une liste de volumes composés d'instances de caractéristiques. Une couche évaluée représente la forme finie par un B-Rep.

Nous avons proposé une solution pour modéliser les caractéristiques de forme dans le système de CAO, en intégrant leur forme et leur fonction. En ce qui concerne la forme, la principale originalité est sa représentation par une surface génératrice, correspondant souvent, pour les caractéristiques secondaires, à un volume ouvert, qui s'étend dans certaines directions. L'intérêt de cette solution est de garantir que les caractéristiques négatives débouchent de leur volume

porteur et que les positives en atteignent les limites, ce qui leur confère un comportement très utile de prolongement automatique. Nous avons ensuite choisi de conserver, pour chaque instance de caractéristique, ce que nous avons appelé le volume utile, dans le principal but de faciliter la gestion des modifications du modèle de conception. À ce sujet, nous avons évoqué un problème qui n'est pas encore complètement résolu à l'heure actuelle. Il s'agit de l'assistance au choix du volume utile quand plusieurs volumes résultent de l'opération booléenne entre la surface génératrice et le volume porteur. En particulier, nous nous proposons d'étudier davantage les possibilités de conservation de ce choix, afin de mieux maintenir l'intention de conception.

Nous pensons avoir correctement appréhendé la phase de conception détaillée. L'intention de conception a été prise en compte sous forme de contraintes géométriques ou équationnelles. Nous avons choisi de représenter les contraintes géométriques par un graphe conceptuel, qui mémorise les contraintes génériques de chaque caractéristique. Les contraintes d'instanciation, qui permettent par exemple d'exprimer un positionnement relatif entre deux caractéristiques, sont alors ajoutées entre deux nœuds des graphes conceptuels respectifs des deux caractéristiques. Nous rappelons que les avantages offerts par graphe conceptuel sont multiples. En effet, il peut être exploité lors de l'extraction de caractéristiques, il contribue au maintient de l'intention de conception et il représente le moyen de fixer un mode d'ancrage pour les caractéristiques secondaires.

Nous envisageons à présent d'approfondir notre travail en réalisant une véritable aide à la conception dès les premiers instants, c'est-à-dire lors de la conception fonctionnelle. En particulier, nous considérons qu'il faut assister le concepteur lors de la décomposition fonctionnelle du cahier des charges du produit. En outre, une association entre une fonction à réaliser et une caractéristique solution est nécessaire et elle doit être la plus automatique possible. Notre système de modélisation basé sur les caractéristiques constitue donc la base d'un futur système de conception fonctionnelle.

Pour améliorer la procédure d'instanciation, nous pourrions essayer de rendre le dialogue de saisie des paramètres (de dimensions, position et orientation) plus convivial pour l'utilisateur. Ceci implique une collaboration étroite avec l'équipe de recherche travaillant sur le dialogue homme-machine. En particulier, nous avons évoqué, dans le paragraphe traitant de l'instanciation des caractéristiques, la transformation d'un ensemble de contraintes d'instanciation en position du repère local. Rendre cette transformation automatique représente un des buts à atteindre. Cette automatisation nécessite la collaboration avec l'équipe de recherche s'intéressant à la gestion des contraintes.

Nous avons également proposé un algorithme de gestion des modifications du modèle de conception. Il se décompose en deux phases : mise à jour de la couche sémantique avec propagation des contraintes, puis mise à jour de la couche B-Rep. La première étape permet de calculer les nouvelles valeurs des paramètres modifiés après propagation des contraintes. Une gestion des contraintes est donc indispensable. La part de travail qui revient au modeleur est d'établir la liste de toutes les caractéristiques concernées par la modification, c'est-à-dire celles qui interfèrent géométriquement avec la caractéristique modifiée et celles qui y sont liées par des contraintes. La gestion des contraintes proprement dite est à la charge d'un solveur, ou plus généralement d'un gestionnaire de contraintes, ce qui implique une fois de plus une collaboration étroite avec "l'équipe contraintes".

La mise à jour de la couche B-Rep est réalisée par la désinstanciation de toutes les caractéristiques de la liste évoquée ci-dessus, puis par leur réinstanciation avec les nouvelles valeurs des paramètres. La désinstanciation et la réinstanciation sont effectuées par des opérations booléennes entre le volume porteur et le volume utile de la caractéristique modifiée. Ce volume utile est donc conservé pour la désinstanciation et réévalué lors de la réinstanciation. L'amélioration des performances, par rapport à une réévaluation systématique de l'historique de conception, est notable si le nombre de caractéristiques modifiées est inférieur à 50 % du nombre total de caractéristiques.

Nous avons en outre étudié l'extension de la bibliothèque des caractéristiques prédéfinies. Nous proposons d'espionner la description interactive par l'opérateur de la forme des nouvelles caractéristiques génériques, pour obtenir la procédure de création de la forme. Cette procédure permet la génération automatique du B-Rep et du graphe conceptuel, c'est-à-dire de l'ensemble des contraintes génériques qui peuvent être détectées ou imposées par l'opérateur, de la génératrice de la nouvelle caractéristique. Cependant, il se peut que l'utilisateur hésite dans la description interactive, ce qui génère des instructions inutiles dans la procédure. Aussi, nous envisageons, en perspective, de la simplifier de manière automatique. Un autre concept intéressant serait de permettre une définition de nouvelle caractéristique, sans détailler la forme a priori, mais en décrivant la caractéristique uniquement en termes de fonctions. La forme pourrait alors être déduite automatiquement.

Une mise en œuvre a été réalisée pour valider les concepts proposés. Nous avons montré l'intérêt d'une programmation objets, particulièrement pour la représentation de la structure hiérarchique de la bibliothèque. Une méthodologie d'analyse et de conception orientée objets a été suivie et a abouti à l'établissement de plusieurs diagrammes de classes, représentant tous les

concepts utilisés. Nous remarquons que des modules déjà opérationnels au sein du laboratoire ont été réutilisés et n'ont donc pas fait l'objet d'un développement spécifique, mais tout au plus d'une légère adaptation : il s'agit des opérations booléennes et du module de visualisation. La collaboration avec d'autres équipes a été nécessaire, une fois de plus.

Le système a été conçu de telle sorte que diverses extensions soient possibles. Elles constituent également des perspectives intéressantes pour enrichir les qualités du système. La première concerne la nature des caractéristiques prises en compte. D'autres types de caractéristiques peuvent être considérés comme des attributs des caractéristiques de forme, comme les tolérances par exemple, et pourront donc y être associés. Pour compléter le système, il faudrait également développer un module d'extraction de caractéristiques de forme, qui rendrait ainsi possible le passage du modèle évalué (B-Rep) au modèle à base de caractéristiques et faciliter la génération de modèles applicatifs.

Nous avons souligné, à plusieurs reprises tout au long de cette conclusion, l'importance de la collaboration avec d'autres équipes de recherche. Nous rappelons la coopération des personnes ayant mis en œuvre respectivement les opérations booléennes et le visualiseur, pour la réalisation de la maquette. Cette collaboration entre plusieurs équipes est concrète, puisqu'elle a abouti à la rédaction de deux articles en commun. Le premier présente l'architecture générale proposée pour le système REGAIN [GJL 97a], projet fédérateur de notre laboratoire. Le second article [GJL 97b] est plus axé sur la prise en compte des contraintes par le dialogue et par le modeleur à base de caractéristiques, et sur leur gestion. Nous espérons pour la suite de notre recherche une association avec les équipes spécialisées dans le dialogue homme-machine et la gestion des contraintes, pour pouvoir mener à bien toutes les perspectives évoquées dans les paragraphes précédents.

Annexe A - TABLEAU COMPARATIF DE QUELQUES SYSTÈMES DE MODÉLISATION À BASE DE CARACTÉRISTIQUES

Le tableau qui suit a été la première conclusion de l'étude bibliographique que nous avons menée. Il avait pour vocation d'établir un comparatif des systèmes étudiés. Il a donc été le point de départ de la spécification de notre modèle à base de caractéristiques, en mettant en évidence les aspects qu'il nous paraissait intéressant de conserver, d'améliorer ou de rejeter.

Les critères de comparaison ont diverses sources. Certains ont été proposés par d'autres auteurs dans leurs propres comparatifs, d'autres sont là parce qu'il constituaient les points forts (ou faibles) de modeleurs ayant fait l'objet de publications, certains enfin sont le fruit de nos réflexions.

	Caractéristiques	modèle	liaisons	interférences	bibliothèque
[RoL 88] Purdue University (Indiana, USA)	forme (face ou ensemble de faces) tolérances technologiques	structure CGS + arbre de caract. (graphe de relation) + graphe de graphe d'adjacence de faces	relations spatiales (position, orientation) contraintes liant des paramètres possible d'après e-mail		base de données dans laquelle on peut attacher des informations aux caractéristiques (cf. e-mail)
[ShR 88] Arizona Sate University (Arizona, USA)	forme (volumes) tolérance matériau	graphe de caractéristiques + instances + arbre CSG + B-Rep si nécessaire	héritage de paramètres (fils fonction du père)	indiquées au moment de l'instanciation (positionnement relatif)	bibliothèque de caractéristiques génériques pour chacun des trois modeleurs
[CMV 91] Ohio Sate University (Ohio, USA)	forme (volumes) tolérance fonctionnelle	trois niveaux : graphe de relation + B-Rep pour chaque caract. + B-Rep final	relations spatiales (est-sur, est-dans, adjacent) et relations entre dimensions		organisée de façon hiérarchique (caract. définies de façon générique dans des classes)
[ShL 93] National Tsing Hua University (Taiwan)	forme (volumes) tolérance	graphe de dépendances + B-Rep paramétré	distance dans une relation père - fils dans une même direction	indiquées au moment de l'instanciation (positionnement relatif)	caract. de forme représentée par un B-Rep, des paramètres de taille et de position et des contraintes
[DZL 93] Huazhong University of Science and Technology (China)	forme (volumes) tolérance rugosité	"CSG" + B-Rep	relations spatiales entre caractéristiques	résolues par le processeur d'opérations booléennes	chaque caract. est représentée par un nom, une trajectoire et une génératrice (extrusion)
[LaM 93] Helsinki University of Technology (Finland)	forme (volumes)	B-Rep sous forme de graphe SAAG + arbre de caractéristiques	relation parent - enfant (position relative)	indiquées au moment de l'instanciation (positionnement relatif)	composée de classes de caractéristiques
[MFG 96] Genova (Italy) Darmstadt (Germany)	forme (volumes)	graphe de relation (caract. explicites) + modèle intermédiaire (explicites et détectées) + B-Rep + modèles d'application	relations spatiales et topologiques dans le graphe de caractéristiques	intersection entre caractéristiques stockées dans la représentation intermédiaire (adjacence)	composée de caractéristiques prédéfinies et de caractéristiques définies par l'utilisateur
[GoT 91] Universidade de Coimbra (Portugal)	forme (volumes)	B-Rep + CSG + graphe			non (procédure de création)
[ReC 86] University of Rochester (NewYork, USA)	forme (surfaces) tolérance	CSG + VGraph			

ajout	taxonomie	mapping	détection	repère	objets fictifs
possible d'après e-mail	celle de CAM-I			local	référentiel
par spécif. de : - paramètres utilisateur - param. hérités - rep. solide - ens. de règles	oui mais non utilisée explicitement	langage interprété		local	oui mais la définition n'est pas précisée
précision de : forme, dimensions, type de dim., repère local ; création d'une classe dans la hiérarchie	hiérarchie : primaires / secondaires positives / négatives			primaire : global secondaire : local	référentiels
création de la forme puis spécification des dimensions et des contraintes pour préciser le comportement	plusieurs définitions pour une même caractéristique de forme avec des comportements différents			local	axes
définition d'une forme paramétrée (forme 2D + trajectoire) relations spatiales et règles de mapping	structurée de façon hiérarchique (cylindre → vis)	maillage préparation de la fabrication	rendue inutile par le mapping	local	référentiel
outil : langage de définition de caractéristiques	permet l'héritage de l'information commune		reconnaissance incrémentale	local (certainement)	
création avec le modeleur solide ou le modeleur à base de caractéristiques définition de contraintes (comportement)	spécifiques aux applications	transformation de la représentation intermédiaire en modèles orientés applications et vice-versa	le module de reconnaissance permet l'extraction et la transformation en caractéristiques d'application	?	
				global (certainement)	
				local	axes, plans

Annexe B - STRUCTURE DU MODÈLE DE CONCEPTION

Cette annexe a pour objet de détailler la structure du modèle de conception. Nous détaillons donc d'abord cette structure, puis nous l'illustrons par un exemple. Enfin, la structure de données informatique sera expliquée.

1. Description du modèle

Le modèle de conception est une structure qui permet de mémoriser toute la scène, c'est-à-dire tous les volumes qui la composent. Il permet également de mémoriser un historique de construction, utile pour le maintien de l'intention de conception. L'ordre de création des volumes composant la scène est donc conservé dans une liste chaînée.

Chaque volume est représenté par une structure en deux couches. La première est une représentation évaluée de la forme, de type B-Rep ; c'est donc un modèle purement géométrique. Nous ne détaillons pas cette structure car il s'agit d'un B-Rep classique, structuré en faces, arêtes et sommets. Nous avons utilisé le B-Rep développé pour les opérations booléennes.

La seconde couche de la structure représentant un volume permet de mémoriser l'historique de construction par une liste des composants du volume (cf. figure B.1). Nous entendons par composant soit une caractéristique, soit une opération booléenne entre deux volumes (union, intersection, différence), soit une transformation géométrique (translation, rotation, changement d'échelle). Nous pouvons faire une analogie entre ces composants et les nœuds d'un arbre CSG, où les volumes élémentaires sont toutefois remplacés par des caractéristiques. Notons cependant qu'un volume élémentaire d'un CSG est une représentation purement géométrique, alors qu'une

caractéristique est une représentation d'un niveau sémantique plus élevé. En effet, en plus de la forme, les caractéristiques permettent de mémoriser l'intention de conception, en particulier par des contraintes. Nous rappelons que les contraintes géométriques sont mémorisées dans un graphe conceptuel dont les nœuds représentent des géométries du B-Rep et les arcs sont valués par des contraintes et éventuellement des paramètres.

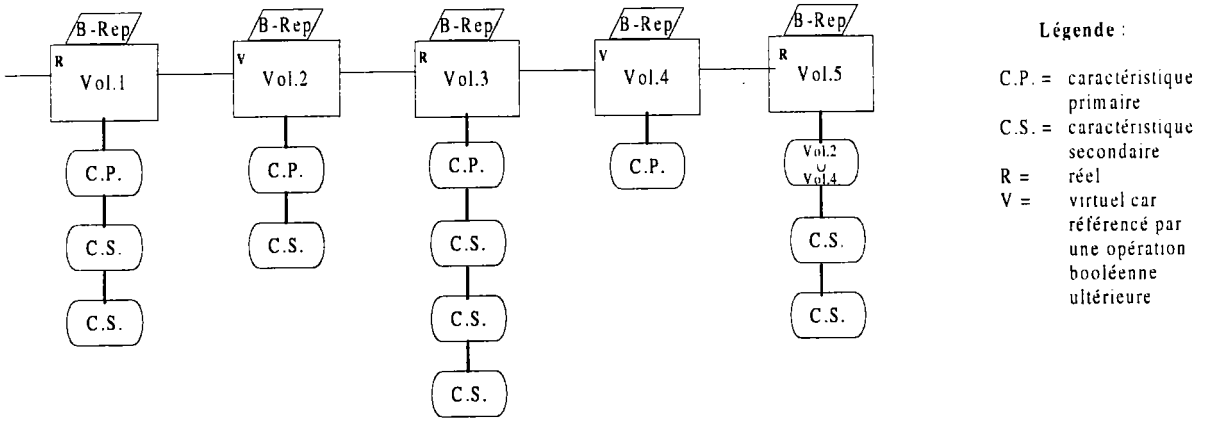


Figure B.1 : Représentation d'une scène par une liste de volumes.

2. Exemple de modèle

Pour clarifier cette structure, nous proposons en guise d'exemple, de détailler les étapes de la création d'un volume composé d'une boîte parallélépipédique et d'une rainure. La première étape est la création dans le modèle de la boîte. L'opérateur doit d'abord dimensionner et positionner l'instance, puis elle est insérée dans le modèle. Le volume créé, à partir de la caractéristique générique primaire *boîte*, est représenté par le B-Rep de la boîte et son graphe conceptuel. La figure B.2 représente le modèle de conception simplifié du volume, avec les liens entre les deux couches. Notons toutefois que, comme nous l'avons déjà précisé, les liens entre B-Rep et graphe passent par l'intermédiaire de géométries. Cependant, pour simplifier le schéma, nous avons représenté ces liens directement entre les nœuds et les faces du B-Rep.

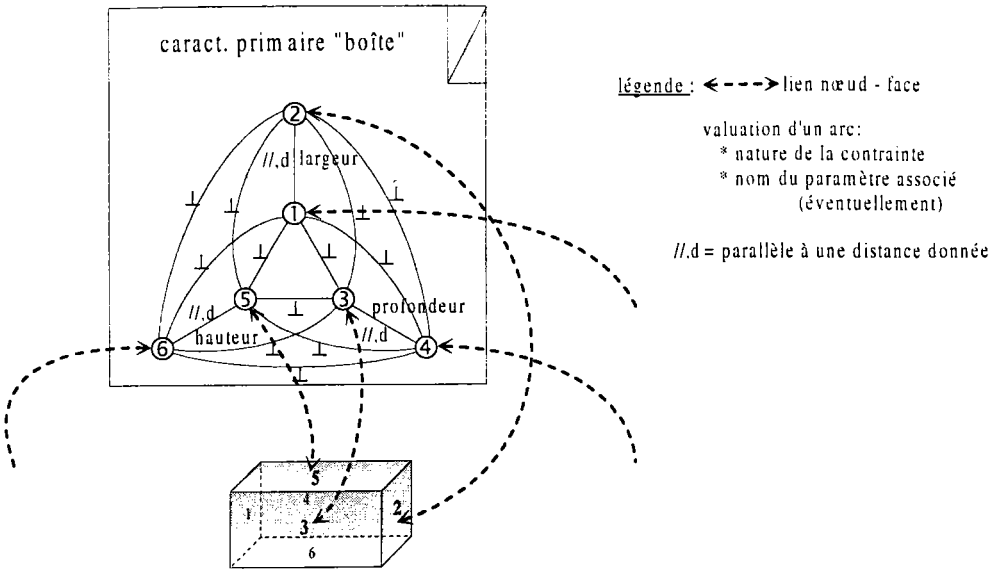


Figure B.2 : Modèle de conception après la première étape.

La seconde étape de la création de la pièce est l'instanciation de la rainure. Son dimensionnement et son positionnement sont établis de façon interactive par l'opérateur. Dans l'exemple, nous supposerons que le positionnement a été réalisé à l'aide de deux contraintes de distance entre les faces de la rainure et celles de la boîte. Son orientation est également précisée (vers le haut) ; la rainure est alors bien définie et peut être instanciée. Comme nous l'avons décrit dans le chapitre 2, l'instanciation est réalisée par deux opérations booléennes : la première détermine le volume utile de la caractéristique (qui sera conservé) et la seconde réalise l'évaluation de la caractéristique dans son porteur. Le résultat de cette seconde opération booléenne représente le nouveau B-Rep du volume. Le nouveau modèle de conception est représenté par la figure B.3.

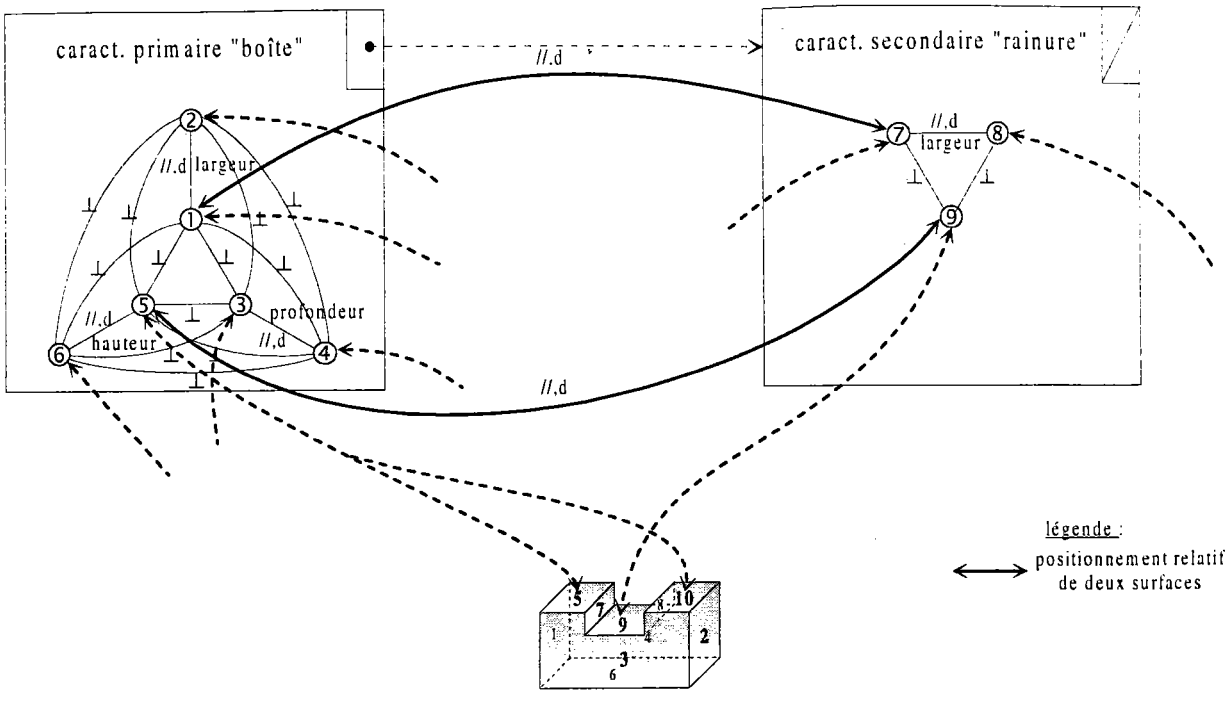


Figure B.3 : Modèle de conception après la deuxième étape.

3. Réalisation informatique

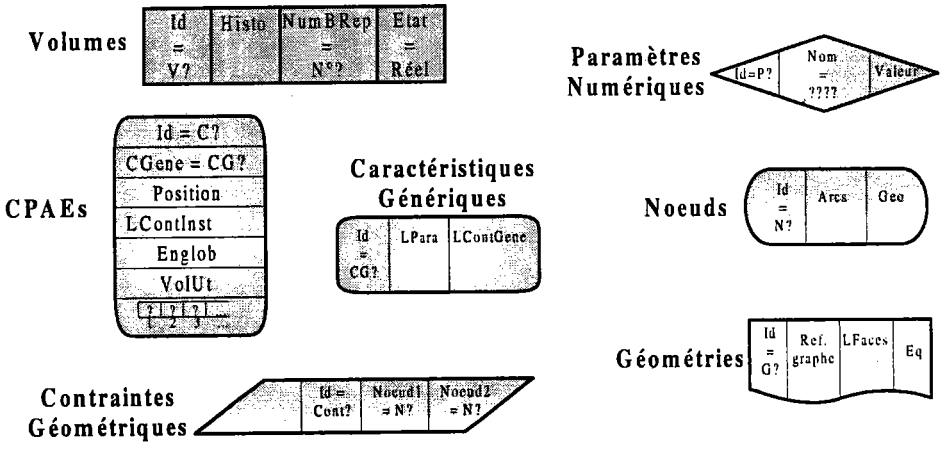
Les figures B.2 et B.3 schématisent le graphe et le B-Rep, mais ne donnent pas une idée précise de la réalisation informatique de la structure. La description des classes mises en œuvre pour les volumes, les caractéristiques, les paramètres et les contraintes a été donnée au chapitre 5.

Nous rappelons tout de même qu'un volume est défini par trois données : l'historique de construction (liste de composants), le numéro de son B-Rep évalué et un état (réel ou virtuel). L'état est une donnée permettant de faire la distinction entre un volume qui existe réellement dans la scène en tant que tel et un volume ayant existé pendant la création du modèle, mais ayant été supprimé ou combiné. En effet, lorsque deux volumes existant dans la scène sont combinés, un nouveau volume est créé, résultant de l'opération booléenne. Les deux opérandes deviennent alors virtuels. Ils ne sont pas supprimés dans le modèle pour conserver l'historique et permettre par exemple une marche arrière.

Nous rappelons également qu'une caractéristique instanciée est créée à partir d'une caractéristique générique. Cette caractéristique générique est définie par une liste de paramètres et une liste de contraintes génériques. La caractéristique instanciée possède, entre autres données, une liste de contraintes d'instanciation.

Une contrainte géométrique est définie par deux nœuds et éventuellement un paramètre numérique. Un nœud est une référence vers une géométrie ; il possède la liste des arcs qui le contraignent. Une géométrie est représentée par une équation (plan) ; elle contient une référence vers un nœud et la liste de toutes les faces qui s'appuient su elle. Enfin, un tableau de correspondance entre le B-Rep et les géométries contient pour chaque face l'identifiant de la caractéristique à laquelle elle appartient (nécessaire pour la désignation).

La structure du modèle, pour l'exemple de la pièce formée d'une boîte rainurée, est donnée dans la figure B.4. Les liens sont schématisés par des flèches, pour qu'ils soient bien visibles, mais sont en réalité des identifiants.



Légende de la figure B.4
(CPAE = Caractéristiques prêtes à l'emploi i.e. caractéristiques d'instanciation)

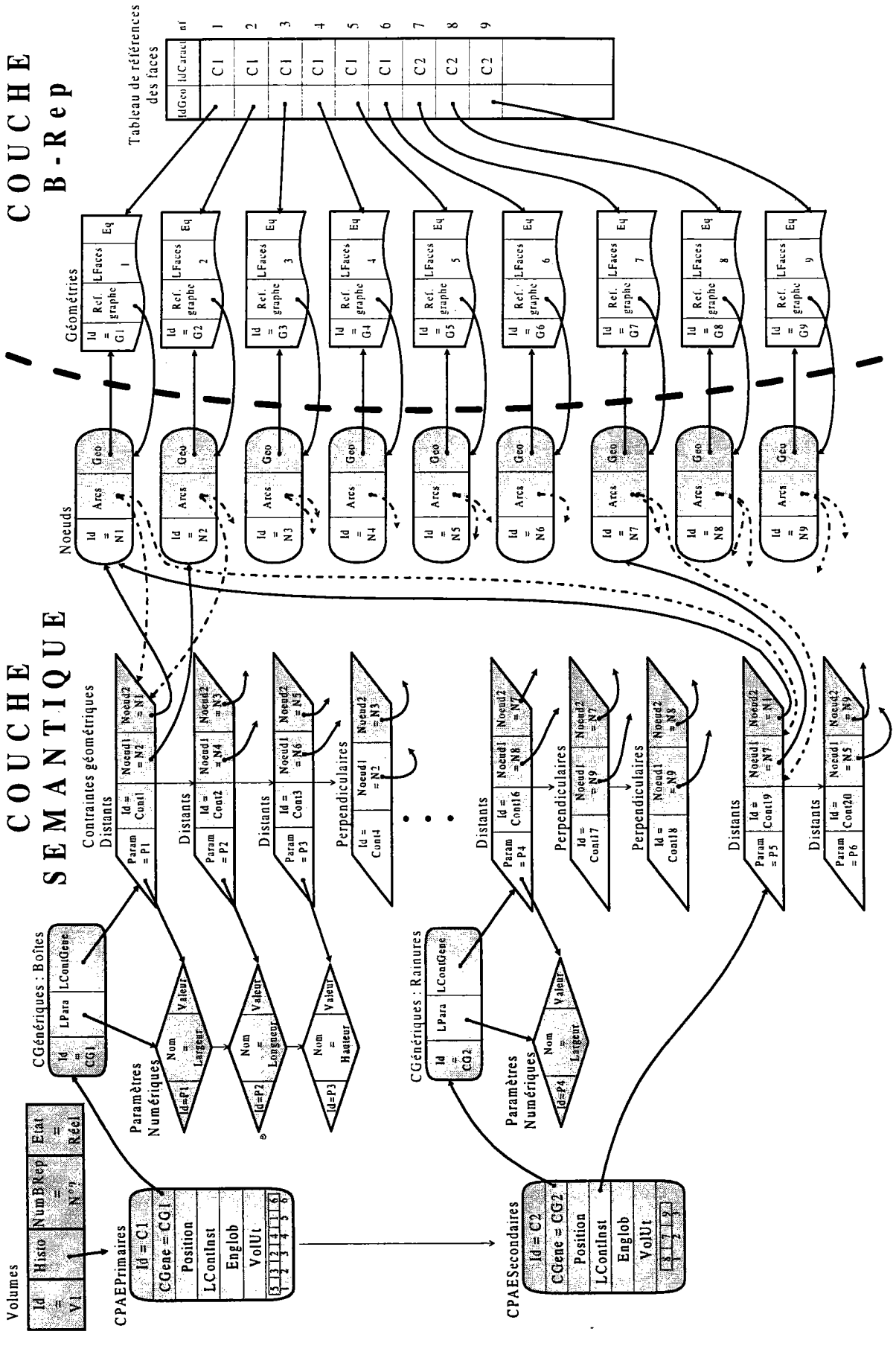


Figure B.4 : Structure du modèle de conception.

Annexe C - PREUVE DES PROPRIÉTÉS DES OPÉRATIONS BOOLÉENNES

Nous rappelons tout d'abord quelques définitions et propriétés des opérations booléennes, ainsi que les lois de De Morgan. Nous donnons donc une liste de règles pour des opérations sur des sous-ensembles A, B et C d'un ensemble X [Hog 92].

1. Définitions

$x \in A$ signifie "x est un élément de A", $A \subseteq B$ signifie "A est un sous-ensemble de B".

$A \cup B$ = $\{x \in X : x \in A$ ***ou*** $x \in B$ (ou les deux)$\}$, l'*union* de A et B.

$A \cap B$ = $\{x \in X : x \in A$ ***et*** $x \in B\}$, l'*intersection* de A et B.

$A - B$ = $\{x \in A : x \notin B\}$, la *différence* de B à A.

A^c = $\{x \in X : x \notin A\}$, le *complément* de A.

2. Règles de base

(a) Commutativité :

$A \cup B = B \cup A$ $\qquad$ $A \cap B = B \cap A$

(b) Associativité :

$(A \cup B) \cup C = A \cup (B \cup C)$ $\qquad$ $(A \cap B) \cap C = A \cap (B \cap C)$

(c) Distributivité :

$A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$ $\qquad$ $A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$

(d) Lois de De Morgan :

$(A \cup B)^c = A^c \cap B^c$ $\qquad$ $(A \cap B)^c = A^c \cup B^c$

(e) Propriétés évidentes :

$A \cup \varnothing = A$ $\qquad$ $A \cap X = A$ $\qquad$ $(A^c)^c = A$

$A \cap \varnothing = \varnothing$ $\qquad$ $A \cup X = X$ $\qquad$ $A \cup A = A = A \cap A$

3. Théorème

Nous constatons que les seules propriétés dont nous disposons impliquent l'union et l'intersection, mais pas la différence. [KiY 92] proposent un théorème permettant d'exprimer la différence en fonction de l'intersection et du complémentaire :

(f) $A - B = A \cap B^c$

4. Propriétés à montrer

si $A \cap B = \varnothing$ alors $A - B = A$ (1)

$A - B - C = A - C - B$ (2)

$A - (A \cap B) = A - B$ (3)

$(A \cup B) - B = A - B$ (4)

$A \cup (B - A) = A \cup B$ (5)

$(A - B) \cup (A \cap B) = A$ (6)

$(A \cup B) - (B - A) = A$ (7)

5. Preuves.

(1)

$A - B = A \cap B^c$	d'après (f)
$= (A \cap B^c) \cup (A \cap A^c)$	car $A \cap A^c = \varnothing$ et (e)
$= A \cap (B^c \cup A^c)$	d'après (c)
$= A \cap (B \cap A)^c$	d'après (d)
$= A \cap \varnothing^c$	par hypothèse $(A \cap B = \varnothing)$
$= A$	car $\varnothing^c = X$ et (e)

(2) $(A - B) - C = (A \cap B^c) \cap C^c$ d'après (f)

$= A \cap (B^c \cap C^c)$ d'après (b)

$= A \cap (C^c \cap B^c)$ d'après (a)

$= (A \cap C^c) \cap B^c$ d'après (b)

$= (A - C) - B$ d'après (f)

(3) $A - (A \cap B) = A \cap (A \cap B)^c$ d'après (f)

$= A \cap (A^c \cup B^c)$ d'après (d)

$= (A \cap A^c) \cup (A \cap B^c)$ d'après (c)

$= A \cap B^c$ car $A \cap A^c = \varnothing$ et (e)

$= A - B$ d'après (f)

(4) $(A \cup B) - B = (A \cup B) \cap B^c$ d'après (f)

$= (A \cap B^c) \cup (B \cap B^c)$ d'après (c)

$= A \cap B^c$ car $B \cap B^c = \varnothing$ et (e)

$= A - B$ d'après (f)

(5) $A \cup (B - A) = A \cup (B \cap A^c)$ d'après (f)

$= (A \cup B) \cap (A \cup A^c)$ d'après (c)

$= A \cup B$ car $A \cup A^c = X$ et (e)

(6) $(A - B) \cup (A \cap B) = (A \cap B^c) \cup (A \cap B)$ d'après (f)

$= A \cap (B^c \cup B)$ d'après (c)

$= A$ car $B^c \cup B = X$ et (e)

(7) $(A \cup B) - (B - A) = (A \cup B) \cap (B \cap A^c)^c$ d'après (f)

$= (A \cup B) \cap (B^c \cup (A^c)^c)$ d'après (d)

$= (A \cup B) \cap (B^c \cup A)$ d'après (e)

$= (A \cup B) \cap (A \cup B^c)$ d'après (a)

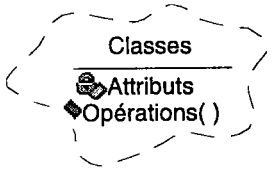
$= A \cup (B \cap B^c)$ d'après (c)

$= A$ car $B \cap B^c = \varnothing$ et (e)

Annexe D - LE FORMALISME OOAD

Cette annexe est un extrait du formalisme OOAD (Object-Oriented Analysis and Design) de [Boo 91], que nous avons utilisé dans le chapitre 4. Nous avons donné dans ce chapitre le diagramme des catégories et les diagrammes des classes. Nous décrivons donc uniquement les éléments composant ces deux types de diagrammes.

1. Classes.



Une classe est représentée par un nuage en pointillés, à l'intérieur duquel on trouve le nom de la classe, ses attributs et ses opérations.

Les attributs et les opérations représentés sont ceux nécessaires à une meilleure compréhension du diagramme. Pour un même système, on peut avoir plusieurs diagrammes de classes.

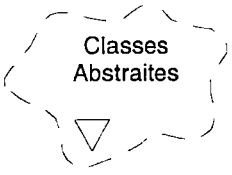
Convention sur les noms d'attributs :

A	→	nom de l'attribut seul
:C	→	classe de l'attribut
A:C	→	nom de l'attribut et sa classe
A:C=E	→	nom de l'attribut, sa classe et sa valeur par défaut

Convention sur les noms d'opérations :

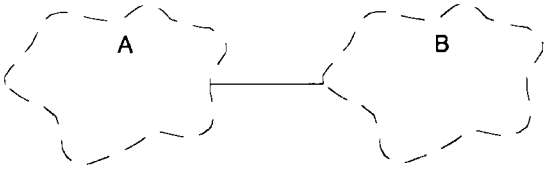
N() → nom de l'opération

R N(arguments) → classe du résultat, nom de l'opération et paramètres



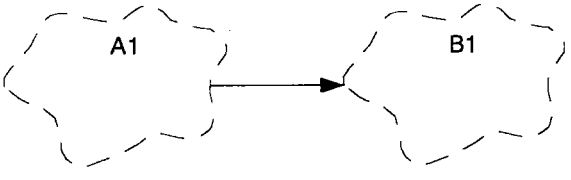
Classes abstraites.

2. Relations entre classes.

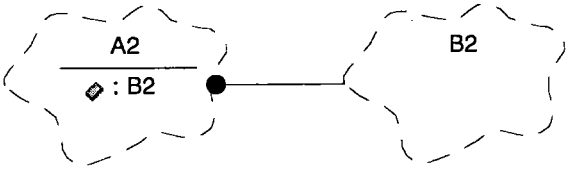


Association.

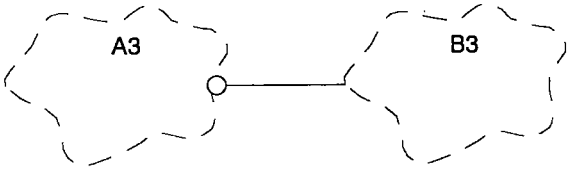
Relation sémantique générale précisée au cours de l'analyse par l'une de trois relations suivante :



Héritage : A1 hérite de B1.

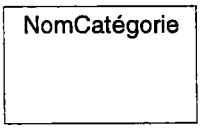


Composition : A2 contient B2.



Utilisation : A3 utilise B3.

3. Catégories de classes.



Lorsque le nombre de classes devient important (≥ 12) on regroupe les classes en catégories. C'est un découpage logique et cohérent avec le problème.

Un diagramme des catégories permet d'avoir une vue logique du système informatique à un niveau d'abstraction plus élevé. Deux catégories peuvent être reliées par des relations d'utilisation. Une catégorie équivaut à un espace de noms en C++.

Annexe E - EXEMPLES DE PROCÉDURES DE CRÉATION DE LA FORME

Cette annexe donne des exemples de procédures de création de la forme de quelques caractéristiques. La première procédure est la création d'une boîte parallélépipédique, à partir des paramètres de dimensions et du repère local. La boîte étant considérée comme un volume de base, c'est une caractéristique prédéfinie. Sa procédure de création permet donc de créer des entités de base du B-Rep (sommets, arêtes, faces puis volume) et le graphe conceptuel associé (dont les nœuds sont des géométries associées aux faces et les arcs sont des contraintes géométriques).

```
int CreerBoite ( const Identifiants& largeur, const Identifiants& longueur,
                 const Identifiants& hauteur, Boolean PremInst, CPAEs& C)
// les données largeur, longueur, hauteur sont des ParamètresNumériques identifiés
// le booléen PremInst indique s'il s'agit d'une première instanciation (vrai) ou d'une réinstanciation (faux)
// C est la caractéristique prête à l'emploi dont on désire créer la forme
{
   ListeEntier  L;
   Points3D     P;
   Identifiants N1, N2, N3, N4, N5, N6;
   double       Larg, Long, Haut;
   Reperes3D    rep = C.Positionnement();    // rep représente le repère local de la caractéristique


// à partir des paramètres numériques, on recherche leurs valeurs
   Larg  = ParamNumeriques::Objet(largeur).Valeur();
   Long  = ParamNumeriques::Objet(longueur).Valeur();
   Haut  = ParamNumeriques::Objet(hauteur).Valeur();
```

```
// connaissant les coordonnées des sommets dans le repère local de la caractéristique,
// ils sont créés correctement placés dans le repère du monde
    P  = rep.RepereVersMonde(Points3D(0,0,Haut));
    int Np1   = MDCreerPoint(P.X(),P.Y(),P.Z());
    P  = rep.RepereVersMonde(Points3D(Larg,0,Haut));
    int Np2   = MDCreerPoint(P.X(),P.Y(),P.Z());
    P  = rep.RepereVersMonde(Points3D(Larg,Long,Haut));
    int Np3   = MDCreerPoint(P.X(),P.Y(),P.Z());
    P  = rep.RepereVersMonde(Points3D(0,Long,Haut));
    int Np4   = MDCreerPoint(P.X(),P.Y(),P.Z());
    P  = rep.RepereVersMonde(Points3D(0,0,0));
    int Np5   = MDCreerPoint(P.X(),P.Y(),P.Z());
    P  = rep.RepereVersMonde(Points3D(Larg,0,0));
    int Np6   = MDCreerPoint(P.X(),P.Y(),P.Z());
    P  = rep.RepereVersMonde(Points3D(Larg,Long,0));
    int Np7   = MDCreerPoint(P.X(),P.Y(),P.Z());
    P  = rep.RepereVersMonde(Points3D(0,Long,0));
    int Np8   = MDCreerPoint(P.X(),P.Y(),P.Z());

// création des arêtes
    int Na1   = MDCreerArete(Np1, Np2);
    int Na2   = MDCreerArete(Np2, Np3);
    int Na3   = MDCreerArete(Np3, Np4);
    int Na4   = MDCreerArete(Np4, Np1);
    int Na5   = MDCreerArete(Np5, Np6);
    int Na6   = MDCreerArete(Np6, Np7);
    int Na7   = MDCreerArete(Np7, Np8);
    int Na8   = MDCreerArete(Np8, Np5);
    int Na9   = MDCreerArete(Np5, Np1);
    int Na10  = MDCreerArete(Np6, Np2);
    int Na11  = MDCreerArete(Np7, Np3);
    int Na12  = MDCreerArete(Np8, Np4);

// creation des faces planes et des nœuds du graphe conceptuel lors de la première instanciation
// lors d'une réinstanciation, les faces sont créées, mais les géométries correspondant aux nœuds
// sont simplement mises à jour
    L = Cons(Na1, Cons(Na2, Cons(Na3, Cons(Na4, ConsVide()))));
    CreationFaceEtNoeud(L, C, 1, PremInst, N1);
    L = Cons(Na5, Cons(Na10, Cons(-Na1, Cons(-Na9, ConsVide()))));
    CreationFaceEtNoeud(L, C, 2, PremInst, N2);
    L = Cons(Na6, Cons(Na11, Cons(-Na2, Cons(-Na10, ConsVide()))));
    CreationFaceEtNoeud(L, C, 3, PremInst, N3);
    L = Cons(Na7, Cons(Na12, Cons(-Na3, Cons(-Na11, ConsVide()))));
    CreationFaceEtNoeud(L, C, 4, PremInst, N4);
    L = Cons(Na8, Cons(Na9, Cons(-Na4, Cons(-Na12, ConsVide()))));
    CreationFaceEtNoeud(L, C, 5, PremInst, N5);
    L = Cons(-Na5, Cons(-Na8, Cons(-Na7, Cons(-Na6, ConsVide()))));
    CreationFaceEtNoeud(L, C, 6, PremInst, N6);
    L =   Cons(Tete(Geometries::Objet(C.ConsultFace(1)).FacesPortees()),
          Cons(Tete(Geometries::Objet(C.ConsultFace(2)).FacesPortees()),
          Cons(Tete(Geometries::Objet(C.ConsultFace(3)).FacesPortees()),
          Cons(Tete(Geometries::Objet(C.ConsultFace(4)).FacesPortees()),
          Cons(Tete(Geometries::Objet(C.ConsultFace(5)).FacesPortees()),
          Cons(Tete(Geometries::Objet(C.ConsultFace(6)).FacesPortees()),
          ConsVide())))));
```

```
// création du graphe conceptuel (ajout des contraintes génériques) lors de la première instanciation
    if (PremInst)
    {
        CGeneriques *CG = CGeneriques::AdresseObjet(C.Generique());
        CG->AjoutContrainte(Distants::Creer(N3,N5,largeur));
        CG->AjoutContrainte(Distants::Creer(N2,N4,longueur));
        CG->AjoutContrainte(Distants::Creer(N1,N6,hauteur));
        CG->AjoutContrainte(Perpendiculaires::Creer(N1,N2));
        CG->AjoutContrainte(Perpendiculaires::Creer(N1,N3));
        CG->AjoutContrainte(Perpendiculaires::Creer(N1,N4));
        CG->AjoutContrainte(Perpendiculaires::Creer(N1,N5));
        CG->AjoutContrainte(Perpendiculaires::Creer(N2,N3));
        CG->AjoutContrainte(Perpendiculaires::Creer(N2,N5));
        CG->AjoutContrainte(Perpendiculaires::Creer(N2,N6));
        CG->AjoutContrainte(Perpendiculaires::Creer(N3,N4));
        CG->AjoutContrainte(Perpendiculaires::Creer(N3,N6));
        CG->AjoutContrainte(Perpendiculaires::Creer(N4,N5));
        CG->AjoutContrainte(Perpendiculaires::Creer(N4,N6));
        CG->AjoutContrainte(Perpendiculaires::Creer(N5,N6));
    }

// création de l'objet dans le B-Rep
    return MDCreerObjet(L);
}
```

La procédure qui suit permet de créer une marche. Sa génératrice est obtenue à partir d'une boîte de laquelle on supprime quatre faces.

```
int CreerMarche(const Identifiants& angle, Boolean PremInst, CPAEs& C)
{
// il est nécessaire, pour utiliser la fonction précédente, de donner trois paramètres, même si dans le cas
// d'une marche, ils ne sont pas utiles, car ils correspondent aux faces supprimées
// le paramètre ValParDef est donc initialisé par une valeur arbitraire
    int bloc   = CreerBoite(ValParDef, ValParDef, ValParDef, PremInst, C);

// destruction des faces inutiles pour la marche
    DetruireFace(Tete(Geometries::Objet(C.ConsultFace(1)).FacesPortees()));
    DetruireFace(Tete(Geometries::Objet(C.ConsultFace(2)).FacesPortees()));
    DetruireFace(Tete(Geometries::Objet(C.ConsultFace(3)).FacesPortees()));
    DetruireFace(Tete(Geometries::Objet(C.ConsultFace(4)).FacesPortees()));

    return bloc;
}
```

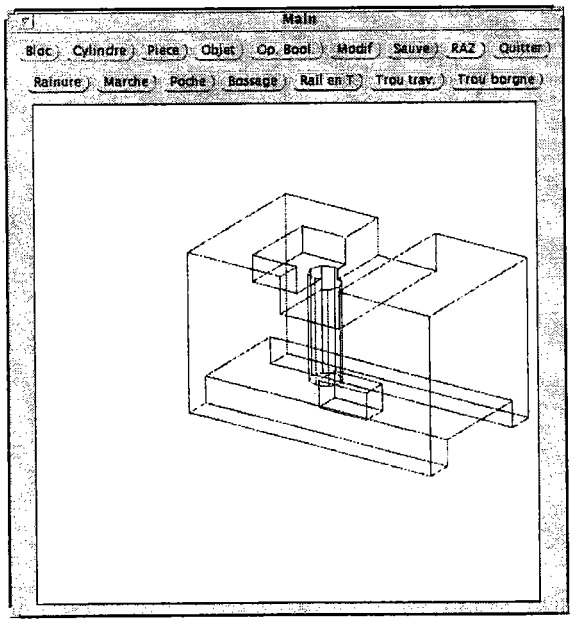
La procédure suivante permet de créer une pièce mécanique, élément de fixation dans un moteur de voiture. La génératrice de cette caractéristique est obtenue en combinant un certain nombre de volumes prédéfinis.

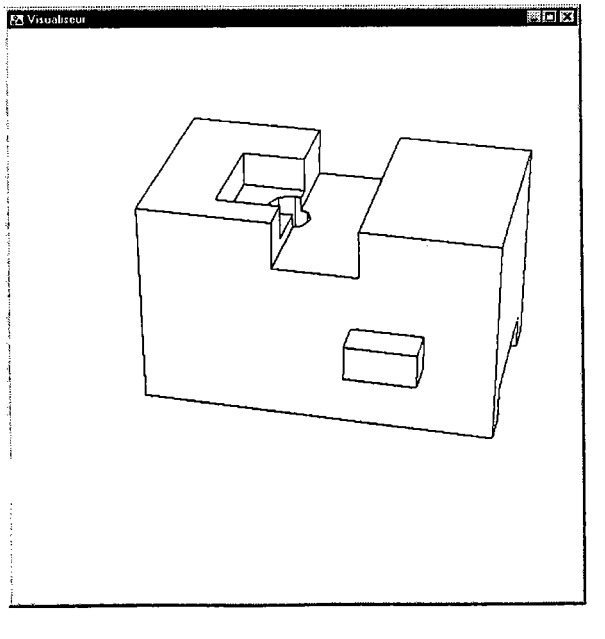
```
int CreerPiece ( const Identifiants& p1, const Identifiants& p2, const Identifiants& p3,
                 const Identifiants& p4, const Identifiants& p5, const Identifiants& p6,
                 const Identifiants& p7, const Identifiants& p8, Boolean PremInst, CPAEs& C)
{
// la création de la pièce est réalisée dans le repère local donnée par (O, i, j, k)
    Vecteurs3D i(1,0,0), j(0,1,0), k(0,0,1), ip, jp;
    Points3D   O(0,0,0);

// les valeurs numériques sont obtenues à partir des paramètres
    double  a  = ParamNumeriques::Objet(p1).Valeur(),
            b  = ParamNumeriques::Objet(p2).Valeur(),
            c  = ParamNumeriques::Objet(p3).Valeur(),
            d  = ParamNumeriques::Objet(p4).Valeur(),
            e  = ParamNumeriques::Objet(p5).Valeur(),
            h1 = ParamNumeriques::Objet(p6).Valeur(),
            h2 = ParamNumeriques::Objet(p7).Valeur(),
            h3 = ParamNumeriques::Objet(p8).Valeur();

// création de volumes prédéfinis et combinaison booléenne de ceux-ci
    int B1 = CreerBloc(a, b, h1, Reperes3D(O, i, j, k), PremInst, C);
    int B2 = CreerBloc(a, b, h1, Reperes3D(Points3D(c,d,0), i, j, k), PremInst, C);
    int D1 = OpDifference(B1, B2, 3);

    ip = Vecteurs3D(a-e,-d,0).VecteurNorme();
    int B3 = CreerBloc(a, b, h1, Reperes3D(Points3D(e,d,), ip, k & ip, k), PremInst, C);
    int D2 = OpDifference(D1, B3, 3);

    jp = Vecteurs3D(-c,b-d,0).VecteurNorme();
    int B4 = CreerBloc(a, b, h1, Reperes3D(Points3D(c,d,0), jp & k, jp, k), PremInst, C);
    int D3 = OpDifference(D2, B4, 3);

    int B5 = CreerBloc(a, b, h1-h2-h3, Reperes3D(Points3D(0,0,h3), i, j, k), PremInst, C);
    int B6 = CreerBloc(e, d, h1, Reperes3D(O, i, j, k), PremInst, C);
    int D4 = OpDifference(B5, B6, 3);

    int D5 = OpDifference(D3, D4, 3);

    int P = MDCreerPoint(c+((e-c)/2), 0, h1/2);
    Vecteur  V(0, 1, 0);
    int Cylindre = MDCreerCylindreTriangule(MDConsPoint(P), V,
                        (e-c)/3, d);
    CreationNoeudsCylindre(Cylindre, C, PremInst);
    int D6 = OpDifference(D5, Cylindre, 3);

    DetruireFace(Tete(Geometries::Objet(C.ConsultFace(5)).FacesPortees()));
    DetruireFace(Tete(Geometries::Objet(C.ConsultFace(8)).FacesPortees()));

    return D6;
}
```

Annexe F - EXEMPLES DE RÉSULTATS D'EXÉCUTION

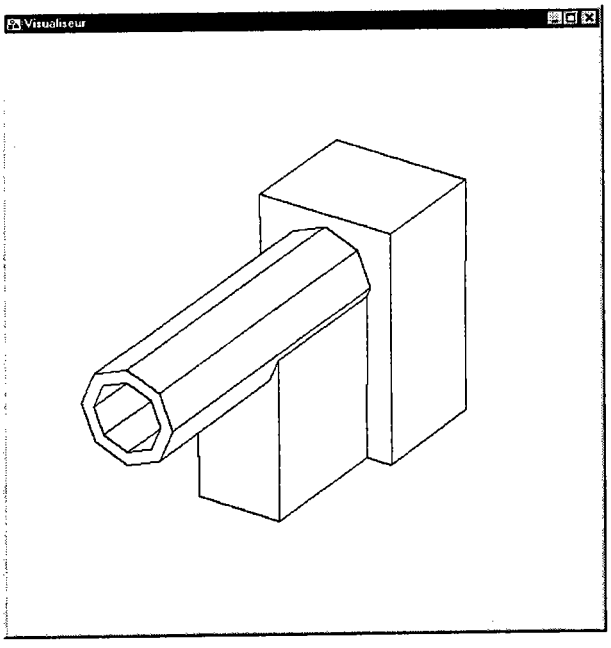
Cette annexe montre quelques objets ayant été créés avec la maquette développée. Le visualiseur intégré à cette maquette ne permet qu'une visualisation au trait, sans élimination des parties cachées. Pour améliorer la qualité des images, nous avons préféré utiliser un autre visualiseur, capable de réaliser un affichage réaliste des modèles obtenus grâce à la maquette et sauvés dans un format particulier. La première image montre l'environnement de la maquette, toutes les autres sont obtenues avec le second visualiseur.

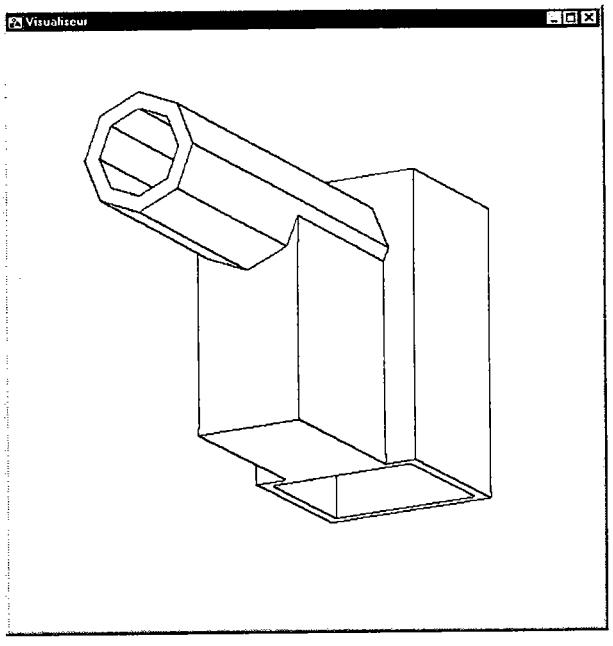
La pièce visualisée sur les images ci-dessous montre quelques unes des caractéristiques de forme qui peuvent être créées grâce à la maquette développée. Elle est obtenue à partir d'une boîte parallélépipédique, sur laquelle sont placés un bossage, une poche, deux rainures et un trou traversant.

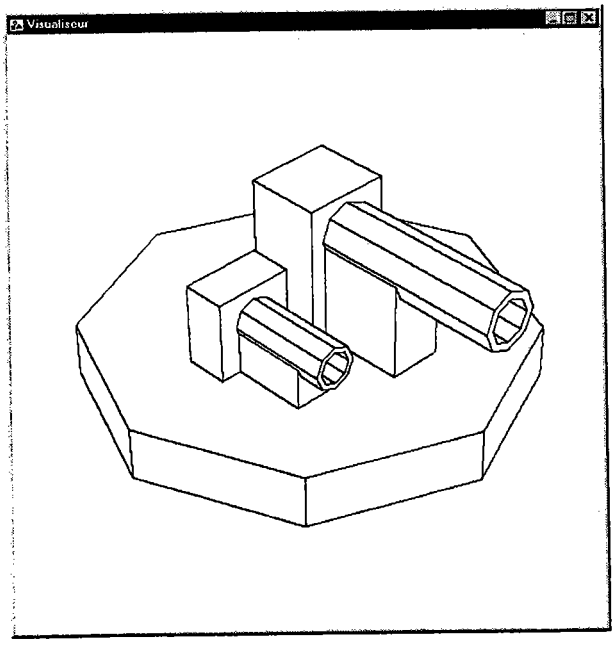




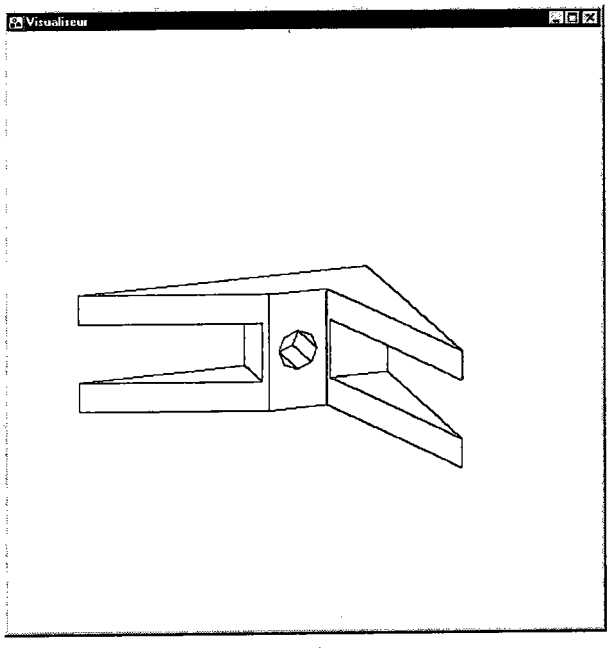
Ci-dessous, une caractéristique de forme moins banale, correspondant à un coude d'admission de liquide LHM dans une citroën XM, vue sous deux angles différents, et deux instanciations de cette caractéristique sur un réservoir de liquide hydraulique, schématisé par un cylindre.

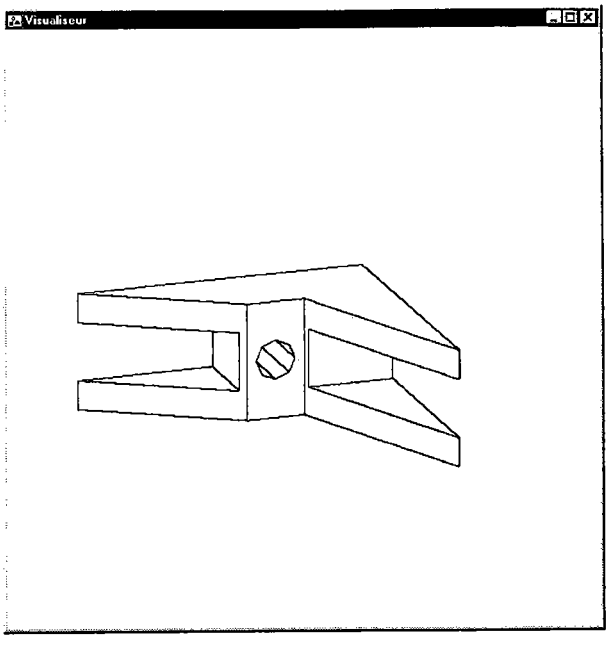




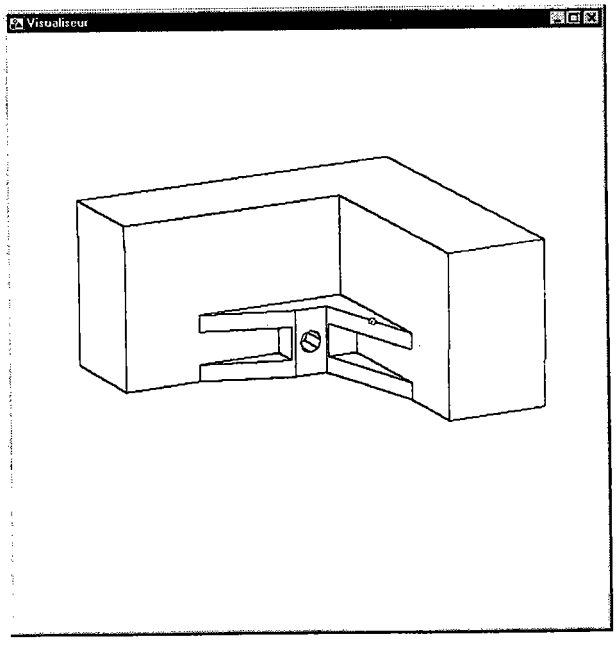


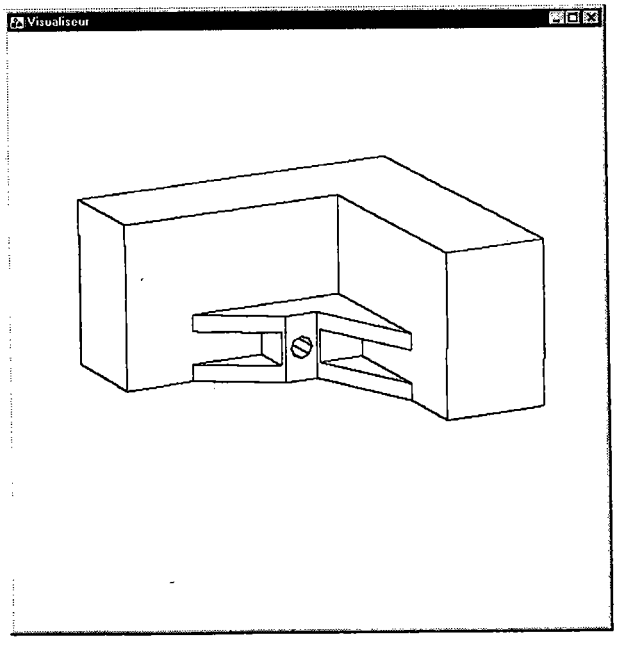
Les deux images ci-dessous présentent une patte de fixation de la pompe à injection dans un moteur, avec des paramètres différents.



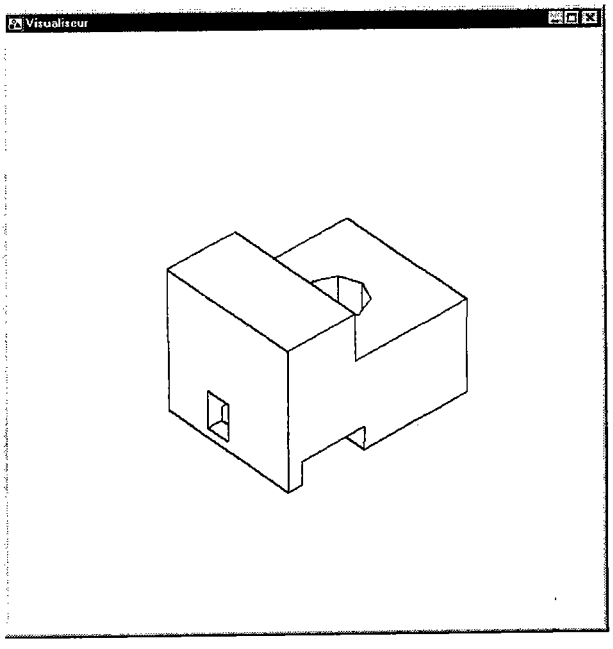


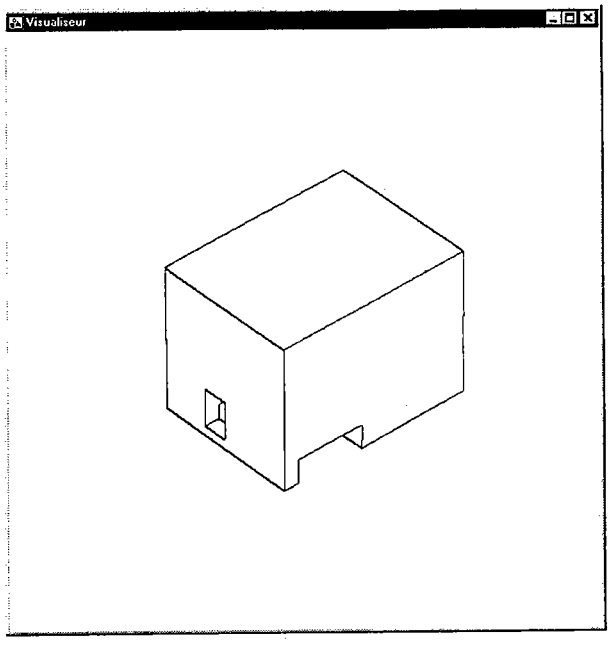
Ci-dessous, la caractéristique précédente, instanciée dans un environnement où le moteur est grossièrement schématisé par l'union de deux boîtes.

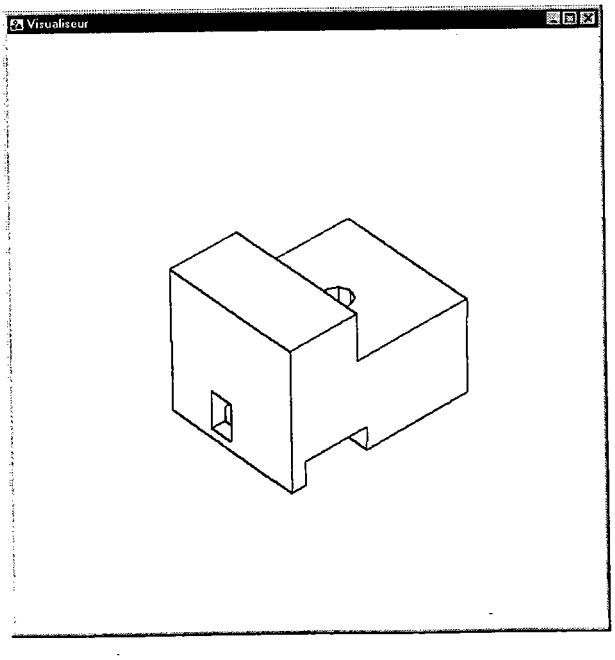




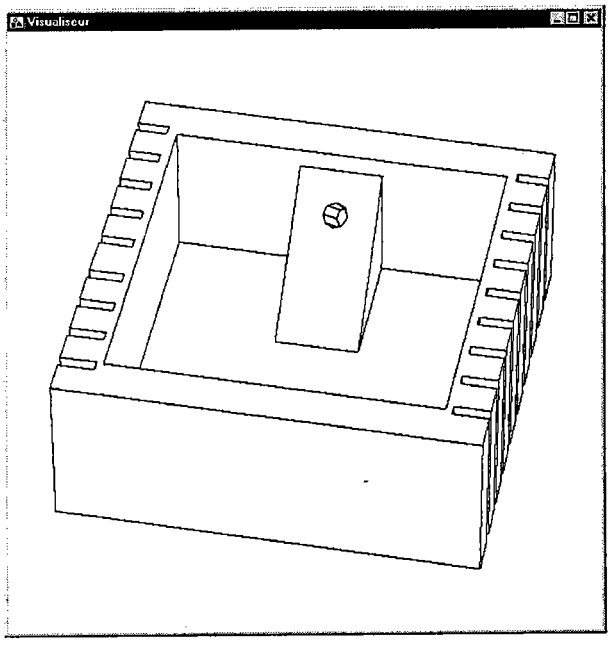
Les trois images qui suivent retracent le scénario d'une modification sur une pièce dans laquelle on a successivement inséré une marche, un perçage, une poche et une rainure. Le rayon du trou est augmenté. Le trou et la marche sont donc désinstanciés (image 2), puis réinstanciés (image 3). La poche et la rainure n'ont pas été affectées.

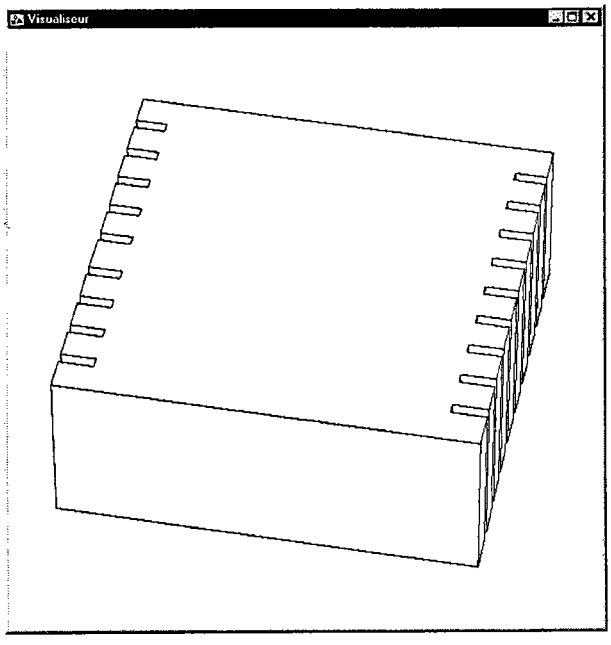


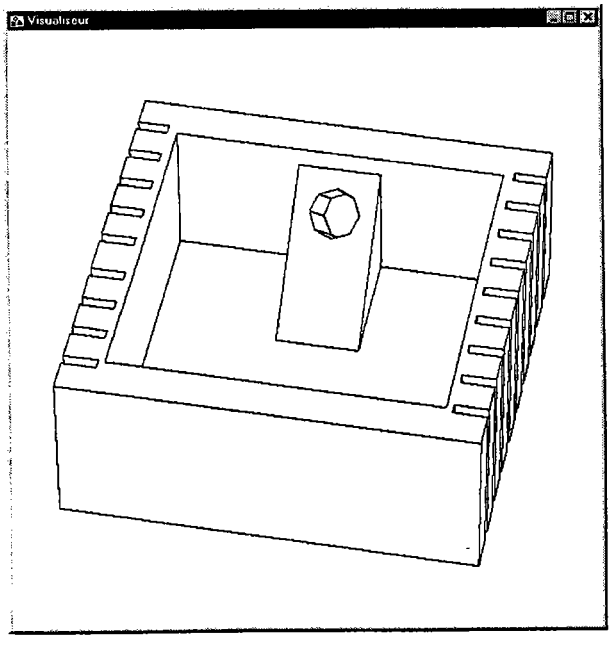




Le scénario ci-dessous montre une pièce composée d'une boîte parallélépipédique, dans laquelle est placée une poche et un certain nombre de rainures sur chacun des côtés. Dans la poche, une nervure a été créée, puis percée. Lors d'une modification du diamètre du trou, il est désinstancié, ainsi que la nervure et la poche, qui interfèrent géométriquement avec le perçage. Par contre, les nombreuses rainures ne sont pas altérées, ce qui représente un gain indéniable en performance (deuxième image). Les trois caractéristiques désinstanciées sont alors réinstanciées, le perçage ayant un nouveau diamètre.







RÉFÉRENCES BIBLIOGRAPHIQUES

[BDS 96] A. BEN AMARA, D. DENEUX, R. SOËNEN, A. DOGUI, Pré-dimensionnement en conception fonctionnelle, *Revue de CFAO et d'informatique graphique*, volume 11, n°1-2/1996, p.133-147

[BiT 93] R. BIDARRA, J.C. TEIXERIA, Intelligent form feature interaction management in cellular modeling scheme, in J. ROSSIGNAC, J. TURNER, G. ALLEN, *Proceedings of Second ACM Symposium on Solid Modeling and Applications*, Montréal, 1993, 19-21 May, ACM Press, p.483-485

[BGS 85] W. BUTTERFIELD, M. GREEN, D. SCOTT, W. STOCKER, *Part features for process planning*, Technical Report R-85-PPP-03, 1985, CAM-I, Inc., Arlington, Texas

[Boo 91] G. BOOCH, *Object Oriented Design with Applications*, The Benjamin/Cummings Publishing Company, Inc., 1991

[CFC 92] E. COCQUEBERT, F. FÉRU, H. CHAOUCH, D. DENEUX, R. SOËNEN, State of the art and evolution in feature-based modeling, *Revue internationale de CFAO et d'infographie*, volume 7, n°2/1992, p169-200

[Che 88] A. CHEVALIER, *Guide du dessinateur industriel*, Hachette Technique, 1988, p.41-70

[ChH 95] X. CHEN, C.M. HOFFMANN, Towards feature attachment, *Computer-Aided Design*, volume 27, n°9, September 1995, p.695-702

[ChL 94] C.L.P. CHEN, S.R. LECLAIR, Integration of design and manufacturing : solving setup generation and feature sequencing using an unsupervised-learning approach, *Computer-Aided Design*, volume 26, n°1, January 1994, p.59-75

[ChS 90] J.C.H. CHUNG, M.D. SCHUSSEL, Technical evaluation of variational and parametric design, *Computers in Engineering*, 1990, p.289-298

[CJC 93] M.A. CHAMBERLAIN, A. JONEJA, T.-C. CHANG, Protrusion-features handling in design and manufacturing planning, *Computer-Aided Design*, volume 25, n°1, January 1993, p.19-28

[CMV 91a] T.M. CHEN, R.A. MILLER, K.R. VEMURI, A framework for feature based part modelling, *ASME Computers in Engineering*, 1991, volume one, p.357 -365

[CMV 91b] T.M. CHEN, R.A. MILLER, K.R. VEMURI, On implementing an integrated design-manufacturability assessment environment, *ASME Computers in Engineering*, 1991, volume one, p.407-413

[Cop 92] J.O. COPLIEN, *Advanced C++ : programming, styles and idioms*, Addison-Wesley Publishing Company

[CuD 88] J.J. CUNNINGHAM, J.R. DIXON, Designing with features : the origin of features, *ASME Computers in Engineering*, 1988, volume one, p.237-243

[DLN 90] J.R. DIXON, E.C. LIBARDI, E.H. NIELSEN, Unresolved research issues in development of design-with-features systems, *Geometric Modeling for Product Engineering*, IFIP 1990, 183-196

[DoW 90] X. DONG, M. WOZNY, Managing feature type dependency in a feature-based modeling system, *ASME Computers in Engineering*, 1990, volume one, p.125-130

[Dun 92] M. DUNN, *Industrial Automation Systems and Integration - Product Data Representation and Exchange - Part 48 : Integrated Generic Resources : Form Feature*, Second edition, 1992, ISO/WD 10303-48

[DZL 93] W. DUAN, J. ZHOU, K. LAI, FSMT : a feature solid-modeling tool for feature-based design and manufacture, *Computer-Aided Design*, volume 25, n°1, January 1993, p.29-38

[FaF 89] B. FALCIDIENO, B. FOSSATI, Representing tolerance information in feature-based solid modelling, *Eurographics'89*, p.463-476

[FeK 95] C.X. FENG, A. KUSIAK, Constraint-based design of parts, *Computer-Aided Design*, volume 27, n°5, May 1995, p.343-352

[FGP 92] B. FALCIDIENO, F. GIANNINI, C. PORZIA, M. SPAGNUOLO, A uniform approach to represent features in different application contexts, *Computers in Industry*, volume 19, 1992, p.175-184

[FHK 96] C.-X. FENG, C.-C. HUANG, A. KUSIAK, P.-G. LI, Representation of functions and features in detail design, *Computer-Aided Design*, volume 28, n°12, December 1996, p.961-971

[FPS 93] Z. FU, A. DE PENNINGTON, A. SAIA, Graph grammar approach to feature representation and transformation, *International Journal of Computer Integrated Manufacturing*, volume 6 (1 & 2), 1993, p.137-151

[Gad 94] R. GADH, Feature mapping and recognition in geometric design generation, in J.J. SHAH, M. MÄNTYLÄ, D. NAU, *Advances in Feature Based Manufacturing*, Elsevier Science Publishers, p.107-128

[Gar 91] Y. GARDAN, *La CFAO : introduction, techniques et mise en œuvre*, Éditions Hermès, France, 1991, 418 pages

[GJL 95] Y. GARDAN, J.P. JUNG, S. LEINEN, B. MARTIN, C. MINICH, C. POINSIGNON, I. STÉMART, *Conception et réalisation d'une maquette illustrant une approche du dialogue, de la gestion des contraintes et de la modélisation en CAO*, Rapport de recherche n°95-01, LRIM, Metz, mars 1995.

[GJL 97a] Y. GARDAN, J.P. JUNG, Y. LANUEL, S. LEINEN, B. MARTIN, C. MINICH, E. PERRIN, C. POINSIGNON, I. STÉMART, Towards adaptable product development, *Proceedings of ICMA'97*, HongKong, Chine, April 1997

[GJL 97b] Y. GARDAN, J.P. JUNG, Y. LANUEL, S. LEINEN, B. MARTIN, C. MINICH, E. PERRIN, C. POINSIGNON, M. SAHNOUNE, F. SCHUTZ, I. STÉMART, A proposal for constraints and features modelling, *Proceedings of ISATA*, Florence, Italy, June 1997

[GMP 95] Y. GARDAN, C. MINICH, C. POINSIGNON, *La modélisation produit : état de l'art*, Rapport de recherche n°95-06, LRIM, Metz, juillet 1995

[GMP 96a] Y. GARDAN, C. MINICH, C. POINSIGNON, IDMME'96 : Propositions pour un modèle produit, *Proceedings of IDMME'96, Tome 2 : First International Conference on Integrated Design and Manufacturing in Mechanical Engineering*, April 15-17, 1996, Nantes, France, p.827-836

[GMP 96b] Y. GARDAN, C. MINICH, C. POINSIGNON, *Spécification d'un modèle produit*, Rapport de recherche n°96-02, LRIM, Metz, mai 1996

[GNR 94] S.K. GUPTA, D.S. NAU, W.C. REGLI, G. ZHANG, A methodology for systematic creation and evaluation of alternative operation plans, in J.J. SHAH, M. MÄNTYLÄ, D. NAU, *Advances in Feature Based Manufacturing*, Elsevier Science Publishers, p.161-184

[GoT 91] A.J.P. GOMES, J.C.G. TEIXEIRA, Form feature modelling in a hybrid CSG/BRep scheme, *Computer & Graphics*, volume 15, n°2, 1991, p.217-229

[GZS 88] D.C. GOSSARD, R.P. ZUFFANTE, M. SAKURAI, Representing dimensions, tolerances and features in MCAE systems, *IEEE Computer Graphics & Applications*, March 1988, p.51-59

[Hog 92] S.G. HOGGAR, *Mathematics for Computer Graphics*, Cambridge University Press

[Hsi 90] D. HSIAO, *Feature mapping and manufacturability evaluation*, Ph.D. dissertation, 1990, Department of Mechanical Engineering, Arizona Sate University

[IMK 94] M. INUI, N. MATSUKI, F. KIMURA, Extended formulation of geometric tolerances based on parametric modifications of surface features, *Proceedings of the IFIP international conference*, Valenciennes 1994, p.673-692

[JRH 92] F. DE JONG, T. REITSEMA, R. HOOGEBOOM, Feature based modelling, a way to improve CAD/CAM process, *Revue internationale de CFAO et d'infographie*, volume 7, n°2/1992, p.209-234

[Jus 92] N.P. JUSTER, Modelling and representation of dimensions and tolerances : a survey, *Computer-Aided Design*, volume 24, n°1, January 1992, p.3-17

[KaN 92] R. KARINTHI, D.S. NAU, An algebraic approach to feature interactions, *IEEE Transaction on Pattern Analysis and Machine Intelligence*, volume 14, n°4, 1992, p.469-484

[KiY 92] K. KITAJIMA, M. YAMAGUCHI, A shell-oriented Boolean set operations algorithm suited for the B-Rep based on boundary edge loops, *Systems and Computers in Japan*, volume 23, n°6, 1992

[Kla 96] R. KLAM, *Proposition et mise en œuvre d'un outil interactif et intuitif de placement d'objets ou de caractéristiques de forme*, Mémoire de DEA, LRIM, Metz, juin 1996

[KOG 96] C. KIM, P.J. O'GRADY, A representation formalism for feature-based design, *Computer-Aided Design*, volume 28, n°6/7, June/July 1996, p.451-460

[KRK 91] F.L. KRAUSE, E. RIEGER, S. KRAMER, PDGL - A language for efficient feature based product gestaltung, *CIRP Annals*, 40, 1, General Assembly 1991

[KrU 92] F.L. KRAUSE, A. ULBRICH, Some application aspects of IMPPACT : an approach to integrated modelling of products and processes, *Revue Internationale de CFAO et d'Infographie*, volume 7, n°2/1992, p.235-265

[LaM 93] T. LAAKKO, M. MÄNTYLÄ, Feature modelling by incremental feature recognition, *Computer-Aided Design*, volume 25, n°8, August 1993, p.479-492

[LaV 97] Y. LANUEL, R. VIVIAN, *Visual features : définition d'une nouvelle approche pour la visualisation*, Rapport de recherche à paraître, LRIM, Metz

[LDS 86] S.C. LUBY, J.R. DIXON, M.K. SIMMONS, Designing with features : creating and using a features data base for evaluation of manufacturability of castings, ASME Computers in Engineering, 1986, p.285-292

[Lei 97] S. LEINEN, *Une nouvelle approche pour la modélisation et la gestion des contraintes en CAO*, Thèse de l'université de Metz, LRIM, Metz, décembre 1997

[LiG 82] R.A. LIGHT, D.C. GOSSARD, Modification of geometric models through variational geometry, *Computer-Aided Design*, volume 14, n°4, July 1982, p.209-214

[Män 90] M. MÄNTYLÄ, A modeling system for top-down design of assembled products, *IBM Journal of Res. and Dev.*, 1990, volume 24, n°5, p.636-659

[MCO 97] F. MANDORLI, U. CUGINI, H. E. OTTO, F. KIMURA, Semantic control in feature-based modeling, *Revue internationale de CFAO et d'informatique graphique*, MICAD 1997, volume 12, n°1-2/1997, p.67-83

[MFG 94] T. DE MARTINO, B. FALCIDIENO, F. GIANNINI, S. HASSINGER, J. OVTCHAROVA, Feature-based modelling by integrating design and recognition approaches, *Computer-Aided Design*, volume 26, n°8, August 1994, p.646-653

[Min 91] C. MINICH, *Contribution à la modélisation d'objets extrudés : modèles théoriques et applications à la CAO*, Thèse de l'université de Metz, LRIM, Metz, février 1991

[Min 96] C. MINICH, Un bilan des techniques d'extraction de caractéristiques de forme, *Revue Internationale de CFAO et d'Informatique Graphique*, volume 11, n°6/1996, p.591-615

[NDZ 91] E.H. NIELSEN, J.R. DIXON, G.E. ZINSMEISTER, Capturing and using designer intent in a design-with-features system, *Third international conference on design theory and methodology*, Miami, September 1991, volume 31, p.95-102

[PaK 96] F. PARIENTÉ, Y.S. KIM, Incremental and localized update of convex decomposition used for form feature recognition, *Computer-Aided Design*, volume 28, n°8, p.589-602

[PCL 90] D.B. PERNG, Z. CHEN, R.K. LI, Automatic 3D machining feature extraction from 3D CSG solid input, *Computer-Aided Design*, volume 22, n°5, June 1990, p.285-295

[Per 95] E. PERRIN, *Une nouvelle approche pour les opérations booléennes : formalisation et mise en œuvre*, Thèse de l'université de Metz, LRIM, Metz, octobre 1995

[Poi 94] C. POINSIGNON, *Proposition et mise en œuvre d'un modeleur basé sur les caractéristiques en CAO*, Mémoire de DEA, LRIM, Metz, septembre 1994

[Pot 95] J.C. POTIER, *Conception sur l'exemple, mise au point et génération de programmes portables de géométrie paramétrée dans le système EBP*, Thèse de doctorat, Poitiers, 1995

[Pra 87] M.J. PRATT, Recent research in form features, in *Advanced Topics in Solid Modeling, ACM Siggraph 87*, Course # 26, Anaheim, CA, July 1987

[Pra 88] M.J. PRATT, Synthesis of an optimal approach to form feature modelling, *ASME Computers in Engineering*, 1988, volume one, p.263-274

[ReC 86] A.A.G. REQUICHA, S.C. CHAN, Representation of geometric features, tolerances, and attributes in solid modelers based on constructive geometry, *IEEE Journal of Robotics and Automation*, volume RA-2, n°3, September 1986, p.156-166

[Req 83] A.A.G. REQUICHA, Toward a theory of geometric tolerancing, *The International Journal of Robotics Research*, volume 2, n°4, Winter 1983, p.45-60

[Req 84] A.A.G. REQUICHA, Representation of tolerances in solid modeling : issues and alternative approaches, in *Solid Modeling by Computers : from Theory to Applications*, J.W. Boyse and M.S. Pickett, Eds. New York : Plenum, 1984, p.3-22

[ReV 89] A.A.G. REQUICHA, J.H. VANDENBRANDE, Form features for mechanical design and manufacturing, *ASME Computers in Engineering*, 1989, volume one, p.47-52

[RLW 91] U. ROY, C.R. LIU, T.C. WOO, Review of dimensioning and tolerancing : representation and processing, *Computer-Aided Design*, volume 23, n°7, September 1991, p.466-483

[RMU 96] M. RANTA, M. MÄNTYLÄ, Y. UMEDA, T. TOMIYAMA, Integration of functional and feature-based product modelling - the IMS / GNOSIS experience, *Computer-Aided Design*, volume 28, n°5, 1996, p.371-381

[RoL 88] U. ROY, C.R. LIU, Feature-based representational scheme of a solid modeler for providing dimensioning and tolerancing information, *Robotics & Computer-Integrated Manufacturing*, volume 4, n°3/4, p.335-345, 1988

[SBH 88] J. SHAH, A. BHATNAGAR, D. HSIAO, Feature mapping and application shell, *ASME Computers in Engineering Conference*, San Fransisco, CA, 1988, July 31 - August 4, volume one, p.489-496

[Sha 91] J.J. SHAH, Assessment of features technology, *Computer-Aided Design*, volume 23, n°5, June 1991, p.331-343

[ShB 86] J. J. SHAH, A. BHATNAGAR, *GT coding scheme for sheet metal features*, Technical Report, 1986, Department of Mechanical Engineering, Arizona State University, Tempe, AZ

[ShL 93] L.C. SHEU, J.T. LIN, Representation scheme for defining and operating form features, *Computer-Aided Design*, volume 25, n°6, June 1993, p.333-347

[ShM 89] J.J. SHAH, D. MILLER, A structure for integrating geometric tolerances with form features and geometric models, *ASME Computers in Engineering*, 1989, volume one, p.395-402

[ShM 95] J.J. SHAH, M. MÄNTYLÄ, *Parametric and feature-based CAD/CAM : Concepts, Techniques, and Applications*, John Wiley & Sons, Inc., 1995

[ShR 88a] J.J. SHAH, M.T. ROGERS, Functional requirements and conceptual design of the feature-based modelling system, *Computer-Aided Engineering Journal*, February 1988, p.9-15

[ShR 88b] J.J. SHAH, M.T. ROGERS, Expert from feature modelling shell, *Computer-Aided Design*, volume 20, n°9, November 1988, p.515-524

[Sow 84] J.F. SOWA, *Conceptual Structures, information processing in mind and machine*, Addison-Wesley Publishing Company, 1984

[SrC 92] A.B. SRIKANTAPPA, R.H. CRAWFORD, Intermediate geometric and interfeature relationships for automatic GT part coding, *ASME Computers in Engineering Conference*, San Fransisco, CA, 1992

[SSJ 94] O.W. SALOMONS, F. VAN SLOOTEN, H.G. JONKER, F.J.A.M. VAN HOUTEN, H.J.J. KALS, Interactive feature definition, *Proceedings of the IFIP international conference*, Valenciennes 1994, p.181-204

[SSK 94] O.W. SALOMONS, F. VAN SLOOTEN, G.W.F. KONING, F.J.A.M. VAN HOUTEN, H.J.J. KALS, Conceptual graphs in CAD, *CIRP Annals* V.43/1, Singapore, August 1994

[SSS 94] J.J. SHAH, Y. SHEN, A. SHIRUR, Determination of machining volumes from extensible sets of design features, in J.J. SHAH, M. MÄNTYLÄ, D. NAU, *Advances in Feature Based Manufacturing*, Elsevier Science Publishers, p.129-157

[TCB 95] S. TICHKIEWITCH, E. CHAPA, P. BELLOY, Un modèle produit multi-vues pour la conception intégrée, *Congrès international de génie industriel de Montréal : "La productivité dans un monde sans frontières"*, 18-20 octobre 1995, p.122-131

[TUY 93] T. TOMIYAMA, Y. UMEDA, H. YOSHIKAWA, A CAD for functional design, *Annals of CIRP'93*, 1993, p.143-146

[VaR 94] J.H. VANDENBRANDE, A.A.G. REQUICHA, Geometric computation for the recognition of spatially interacting machining features, in J.J. SHAH, M. MÄNTYLÄ, D. NAU, *Advances in Feature Based Manufacturing*, Elsevier Science Publishers, p.83-106

[WCJ 93] M.-T. WANG, M.A. CHAMBERLAIN, A. JONEJA, T.-C. CHANG, Manufacturing feature extraction and machined volume decomposition in a computer-integrated feature-based design and manufacturing planning environment, *Computers in Industry*, volume 23, 1993, p.75-86

[WSS 92] C. WEBER, M. SCHULTE, R. STARK, Functional features for design in mechanical engineering, *CARs & FOF, 8th International Conference on CAD/CAM, Robotics and Factories of the Future*, Metz, 1992, Tome 1, p.179-192

[YaW 92] F.C. YANG, M.T. WANG, An object-oriented feature-based computer-aided design system for concurrent engineering, *Energing Technologies and Factory Automation*, Melbourne, 1992 (ETFA'92), p.393-398

INDEX ALPHABÉTIQUE

H

I

L

M

O

P

R

T

U

V

CONTRIBUTION À L'ANALYSE ET À LA RÉALISATION D'UN SYSTÈME DE CAO À BASE DE CARACTÉRISTIQUES DE FORME

Résumé

La modélisation représente une partie très importante des systèmes de Conception et Fabrication Assistées par Ordinateur. Les travaux présentés dans ce mémoire s'inscrivent dans la cadre de la modélisation produit et consistent plus précisément à proposer un système de modélisation basé sur les caractéristiques de forme. Le système proposé remplit deux fonctions principales :

- l'insertion de caractéristiques dans la pièce en cours de conception et leur maintenance pendant toute l'évolution du produit (modification, suppression) ;
- la création par l'opérateur de nouvelles caractéristiques génériques et leur insertion dans la bibliothèque de caractéristiques de forme.

Une structure de données permettant de modéliser les caractéristiques de forme est tout d'abord présentée. Puis une méthode de gestion des modifications dans le modèle de conception, qui évite de réévaluer tout l'historique de construction de la scène, est proposée. Enfin, des outils interactifs et conviviaux permettent à l'utilisateur d'intégrer de nouvelles caractéristiques au logiciel.

Mots clés

Conception assistée par ordinateur, modélisation produit, caractéristiques (*features*), modification incrémentale, intention de conception.